ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 107-112

Rule extraction from support vector machines

Haydemar Núñez[1,3] **Cecilio Angulo**[1,2] **Andreu Català**[1,2]

[1]Dept. of Systems Engineering, Polytechnical University of Catalonia
Avda. Victor Balaguer s/n E-08800 Vilanova i la Geltrú, Spain
{hnunez,cangulo}@esaii.upc.es

[2]LEA-SICA. European Associated Lab. Intelligent Systems
and Advanced Control
Rambla de l'exposiò s/n E-08028 Vilanova i la Geltrú, Spain
andreu.catala@upc.es

[3]Universidad Central de Venezuela. Facultad de Ciencias. Escuela de
Computación. Caracas, Venezuela
hnunez@kuaimare.ciens.ucv.ve

Abstract. Support vector machines (SVMs) are learning systems based on the statistical learning theory, which are exhibiting good generalization ability on real data sets. Nevertheless, a possible limitation of SVM is that they generate black box models. In this work, a procedure for rule extraction from support vector machines is proposed: the SVM+Prototypes method. This method allows to give explanation ability to SVM. Once determined the decision function by means of a SVM, a clustering algorithm is used to determine prototype vectors for each class. These points are combined with the support vectors using geometric methods to define ellipsoids in the input space, which are later transfers to if-then rules. By using the support vectors we can establish the limits of these regions.

1. Introduction

The support vector machine (SVM) is a type of learning machine based on the statistical learning theory, which implements the structural risk minimization inductive principle with the purpose of obtaining a good generalization from limited-size data sets [4,7,10]. Although initially conceived for classification problems of two classes with linearly separable data, new algorithms have already been derived to solve classification problems with non-separable data [5], regression [8] and multi-class problems [2,12].

An important point to stand out is that the support vector machines, like the neural networks, generate black box models in the sense that they do not have the ability to explain, in an understandable form, the process by means of which the exit takes place. In order to bear this limitation, the hypothesis generated by either neural network or SVM could be transferred into a more comprehensible representation; these conversion methods are known as rule extraction algorithms.

In the last years, a proliferation of rule extraction methods from trained neural networks has been observed [1,6,9,11]. Nevertheless, in the case of the SVM, few

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 107-112

research tendencies have been published. In this work, a procedure for the interpretation of the SVM models is proposed: the SVM+Prototypes method. The basic idea is the following one: once determined the decision function by means of SVM, a clustering algorithm is used to determine prototype vectors for each class. These points are combined with the support vectors using geometric methods, to define regions in the input space that can be transferred to if-then rules.

This paper is organized as follows: the foundations of the support vector machines are exposed in the next section. The SVM+prototypes rule extraction method is described in section 3. Then, section 4 describes experimental results of our method applied to several data sets. Finally, we present the conclusions and the future work.

2. Support Vector Machines

Let us consider a binary classification task with the training data set ($\mathbf{x}_i$, y_i), $\mathbf{x}_i \in R^m$, $y_i=\{-1,+1\}$, and let the decision function be f(x)=sign(w·**x**+b). A good generalization is achieved by maximizing the margin between the separating hyperplane (w·**x**+b=0) and the closest data points in the input space. This optimal hyperplane can be determined as follow

minimize: $\langle \mathbf{w}\cdot\mathbf{w} \rangle$
subject to $y_i\,(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 \quad \forall i$

Introducing Lagrange multipliers to solve this problem of convex optimization and making some substitutions, we arrive to the Wolfe dual of the optimization problem:

maximize: $$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,k=1}^{m} \alpha_i y_i \alpha_k y_k (\mathbf{x}_i \cdot \mathbf{x}_k)$$

subject to $$\alpha_i \geq 0 \ \forall i, \quad \sum_{i=1}^{m} \alpha_i y_i = 0$$

The hyperplane decision function can thus be written as:

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{SV} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{x}) + b\right) \qquad (1)$$

In order to expand the method to nonlinear decision functions, the input space projects to another higher-dimensional dot product space F, called feature space, via a nonlinear map ϕ: $R^m \rightarrow F^d$ (d>>m). In this new space the optimal hyperplane is derived. Nevertheless, by using kernel functions which satisfy the Mercer' theorem, it is possible to make all the necessary operations in the input space by using $\phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_j)=K(\mathbf{x}_i,\mathbf{x}_j)$. The decision function is formulated in terms of these kernels:

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \qquad (2)$$

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 107-112

Just a few of the training patterns have a weight α_i non-zero in the equations (1) and (2). These elements lie on the margin and they are known as support vectors (SV). This means that the hypothesis representation generated by the SVM is given solely by the points that are closest to the hyperplane and therefore these are the patterns most difficult to classify.

3. The SVM+Prototypes method for rule extraction

Our proposal takes advantage of the information provided by the support vectors. They are used to determine the boundaries of regions defined in the input space (ellipsoids or hyper-rectangles). These regions are obtained from the combination of prototype vectors and support vectors. The prototype vectors are computed through a clustering algorithm. Each region defines a rule with its corresponding syntax (Figure 1): equation rules, which correspond to mathematical equations of the ellipsoids and interval rules, associated with hyper-rectangles defined from parallel ellipsoids to the coordinate axes.

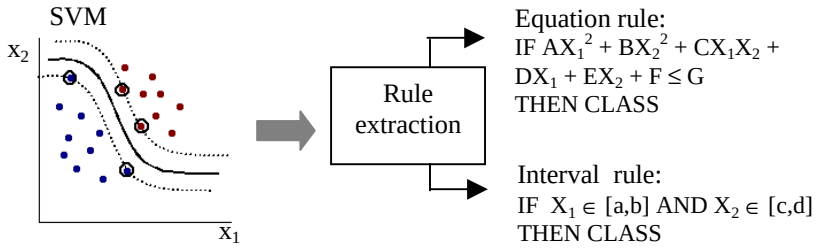

Figure 1. Rules generated by the SVM+prototypes algorithm.

An ellipsoid is defined by the prototype, which will be the centre, and by a support vector within the partition. The chosen support vector will be the farthest to the prototype. The straight line defined by these two points is the first axis of the ellipsoid. By simple geometry, the rest of the axes and the associate vertices are determined. There are three possibilities to define these vertices: with the support vector itself, derived from a support vector or with the farthest point to the prototype. To construct hyper-rectangles a similar procedure is followed. The only difference is that lines parallel to the coordinate axes are used to define the axes of the associated ellipsoid.

In order to define the number of ellipsoids per class, the algorithm follows an incremental scheme. Beginning with a single prototype, the associated ellipsoid is generated. Next, a partition test is applied on this region. If it is negative, the region is transferred to a rule. Otherwise, new regions are generated. This procedure is repeated while a region that fulfils the partition test exists or until the maximum number of iterations is reached. This process allows to control the number of generated rules.

For each iteration, there are m regions with a positive partition test and p regions with a negative partition test. These last ones are transferred to rules. In the next iteration, the data of the m regions are used to determine m+1 new prototypes and to

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 107-112

generate m+1 new ellipsoids. If the maximum number of iterations is reached, all the regions (independently of the results of the partition test) are transferred to rules.

Test conditions attempt to diminish the level of overlapping between regions of different classes by applying several heuristics. The partition of a region is made if the generated prototype belongs to another class, if one of the vertices belong to another class or if a support vector from another class exits within the region.

Figure 2 shows an example of the regions generated by the algorithm (after three iterations). In this case a SVM with a polynomial kernel of degree 2 is trained using randomly-generated separable data (each element is a par $(\mathbf{X},Y)$ with $\mathbf{X} = [x_1,x_2]$, $Y=\{-1,+1\}$). In the Figure the prototype points, the vertices and the support vectors can be observed.

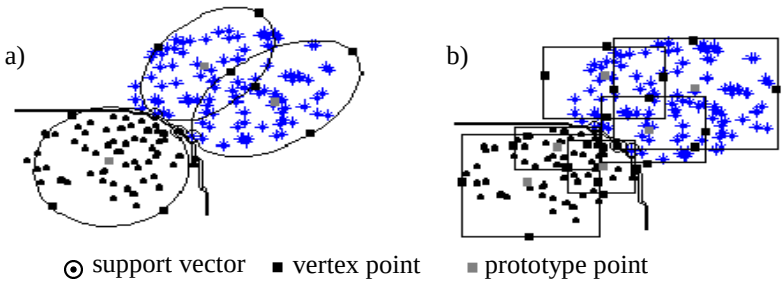

Figure 2. a) Ellipsoids for equation rules. b) Hyper-rectangles for interval rules.

4. Experiments

In order to evaluate the performance of the SVM+Prototypes algorithm, we carried out two kinds of experiments: with artificial datasets and IRIS data set. In both we used vector quantization to generate the prototypes. In the first case, 12 artificial samples were generated randomly (each one constituted by 160 training data and 100 test data). Different overlapping degrees between classes and noise levels were guaranteed for these samples. Afterwards, the decision function was determined by the SVM for each sample using only training data (Figure 3). Finally, the SVM+Prototypes algorithm was applied and rules as following were produced:

IF Condition_1 OR...OR Condition_M THEN Class1 ELSE Class2

Table 1 shows the prediction error obtained by each data set. We observed that the rule error differs from the provided by the SVM at the most in 4%.

In the second experiment, we applied the SVM+Prototypes algorithm to IRIS data set [3]. This process was repeated 30 times, but the data was randomly divided (75% training and 25% test) each time. The prediction error average was 0.03 for SVM, 0.046 for equation rules and 0.047 for interval rules. Bellow, we show the best assembly of interval rules obtained.

R1: IF $X_1 \leq 6.23$ AND $X_2 \geq 2.09$ AND $X_3 \in [1.90,2.03]$ AND $X_4 \leq 0.67$
THEN Iris setosa (error: 0.00)

R2: IF $X_1 \in [4.42,7.13]$ AND $X_2 \in [2.04,3.50]$ AND $X_3 \in [2.89,5.17]$
AND $X_4 \in [0.90,1.79]$ THEN Iris versicolour (error: 0.02)
R1: IF $X_1 \geq 4.50$ AND $X_2 \in [2.34,3.55]$ AND $X_3 \in [4.66,6.51]$ AND $X_4 \geq 1.27$
THEN Iris virginica (error: 0.06)

Where: X_1 = sepal length, X_2 = sepal width, X_3 = petal length, X_4 = petal width.

5. Conclusions and future work

The functions generated by the SVM are constructed in terms of the support vectors. These vectors were used to define the boundaries of the regions defined in the input space. By using support vectors it is possible to build a set of ellipsoids that represents the class with minimum overlapping between classes.

At the moment we are working in the evaluation of different clustering algorithms to improve the generation of prototypes. The selection of good prototypes determines the number and quality of regions. In the future, we will apply the rule extraction algorithm to other public data sets. Also, we want to develop algorithms to simplify rules, with the purpose of producing a compact assembly of rules.

The final objective of this work is the development of a hybrid learning system based on SVM. The idea is that it allows to transform, by means of the rule extraction, the functions produced by SVM and the insertion of the prior domain knowledge.

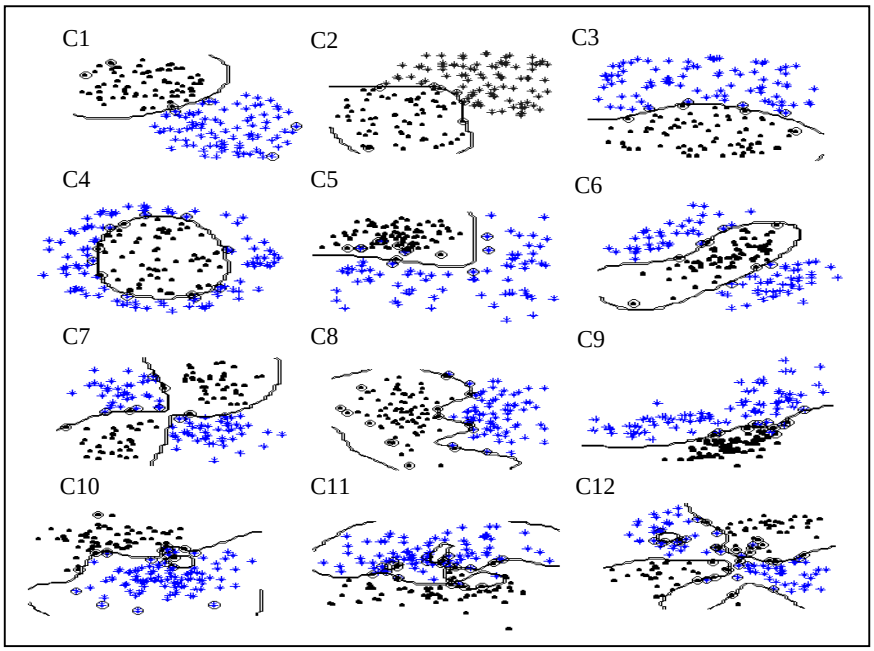

Figure 3. Artificial data sets and decision function generated by SVM

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 107-112

Table1. Results obtained for artificial datasets

	SVM		Equation rules		Interval rules	
Data set	Error	#SV	Error	#C1	Error	#C2
C 1	0.04	7	0.07	1	0.05	1
C 2	0.05	7	0.05	2	0.06	3
C 3	0.03	8	0.04	2	0.06	2
C 4	0.06	12	0.08	5	0.09	6
C 5	0.02	15	0.06	4	0.04	4
C 6	0.02	9	0.03	3	0.05	4
C 7	0.05	10	0.05	2	0.03	2
C 8	0.05	20	0.05	2	0.07	3
C 9	0.03	13	0.04	2	0.02	2
C10	0.11	20	0.11	1	0.13	2
C11	0.16	23	0.16	4	0.19	6
C12	0.16	37	0.15	3	0.15	6

#C1=Nº equation conditions. #C2=Nº interval conditions

References

1. R. Andrews, J. Diederich, A. Tickle: A Survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks. Knowledge-Based Systems, 8(6), 373-389 (1995).
2. C. Angulo, A. Català A: K-SVCR. A Multi-class Support Vector Machines. Proc.of ECML'2000, Lecture Notes in Computer Sciences, 31-38 (2000).
3. C. J. Merz, P. M. Murphy: UCI Repository for Machine Learning Data-Bases.[http://www.ics.uci.edu/~mlearn/MLRepository.html], Department of Information and Computer Science. University of California, Irvine (1996).
4. V. Cherkassky, F. Mulier: Learning from Data. John Wiley & Sons, Inc. (1998).
5. C. Cortes, V. Vapnik: Support-Vector Networks. Machine Learning, 20, 273-297 (1995).
6. M. Craven, J. Shavlik: Using Neural Networks for Data Mining. Future Generation Computer Systems, 13, 211-229 (1997).
7. N. Cristianini, J. Shawe-Taylor: An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press (2000).
8. H. Ducker, C. Burges, L. Kaufman, A. Smola, V. Vapnik: Support Vector Regresion Machines. NIPS, 9, 155-162, MIT Press (1997).
9. A. Tickle. R. Andrews, G. Mostefa, J. Diederich: The Truth will come to light: Directions and Challenges in Extracting the Knowledge Embedded within Trained Artificial Neural Networks. IEEE Transactions on Neural Networks, 9(6), 1057-1068 (1998).
10. V. Vapnik: Statistical Learning Theory. John Wiley&Sons, Inc. (1998).
11. S. Wermter, R. Sun: Hybrid Neural Systems. Springer-Verlag (2000).
12. J. Weston, C. Watkins: Support Vector Machines for Multi-class Pattern Recognition. Proc. of ESANN'99, 219-224 (1999).