UNIVERSITY
OF TRENTO

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.dit.unitn.it

ADAPTIVE MODEL SELECTION FOR DIGITAL LINEAR CLASSIFIERS

Andrea Boni

2002

Technical Report # DIT-02-0034

Also: to be published in Proceedings International Conference on Artificial Neural Networks 2002

Adaptive Model Selection for Digital Linear Classifiers

Andrea Boni

University of Trento
Department of Information and Communication Technologies
Via Sommarive 14,
38100 Povo (Trento), Italy
andrea.boni@ing.unitn.it

Abstract. Adaptive model selection can be defined as the process thanks to which an optimal classifiers h^* is automatically selected from a function class H by using only a given set of examples z. Such a process is particularly critic when the number of examples in z is low, because it is impossible the classical splitting of z in $training + test + validation$. In this work we show that the joined investigation of two bounds of the prediction error of the classifier can be useful to select h^* by using z for both model selection and training. Our learning algorithm is a simple kernel–based Perceptron that can be easily implemented in a counter–based digital hardware. Experiments on two real world data sets show the validity of the proposed method.

1 Introduction

In the framework of classification problems, the machine learning community refers to model selection as the task through which an optimal function class H_k is selected from a given set H with the goal of optimizing the prediction error of the classifier h^* extracted from H_k. Usually such a task involves a tradeoff between the capability of the model to learn the given set of labeled examples $z = \{(x_i, y_i)\}_{i=1}^m$, and a measure of the complexity of the model itself [11]. Each (x_i, y_i) is supposed to be designed independently and identically distributed over an unknown distribution P; we use the following notation: $x_i \in X$, $y_i \in Y$, $z_i = (x_i, y_i) \in X \times Y = Z$; $z \in Z^m$; $X \subset \Re^r, Y = \{-1, 1\}$. In this paper we consider digital learning algorithms $\mathcal{A}$ as operators that map z to a set of parameters A, $\mathcal{A}$:$Z^m \rightarrow A$, $A \subset (N^+)^m$; in particular we consider the Digital Kernel Perceptron with Pocket (DKPP), which leads to the following function class: $H = \{h : \mathcal{K} \times A \times X \rightarrow Y : h(\alpha, x) = \mathrm{sgn}(\sum_{j=1}^m \alpha_j y_j K(x_j, x)), K \in \mathcal{K}, \alpha \in A, x \in X\}$. $K(\cdot, \cdot) : X \times X \rightarrow \Re$ is a kernel function that realizes a dot product in the feature space [1]. The choice of the kernel perceptron (KP) is justified by recent studies that have revalued its role in classification tasks, showing the relation between its generalization ability and the sparsity of the final solution [9]. Furthermore its *simplicity* suggests that a very simple digital hardware can be designed. In this paper we propose an adaptive automatic procedure to select the

best function in H. Our approach is based on recent studies that have proposed uniform (over the distribution P) and data–dependent bounds for the error rate of the KP [9] and for a given learning algorithm [3], respectively. In the next section we briefly introduce the DKPP learning algorithm, while in section 3 we describe uniform and data–depended bounds; based on such bounds we propose a procedure for adaptive model selection. In the end, in section 4 we comment some experiments on two real–world data sets.

2 The DKPP algorithm

It is well known that the KP algorithm belongs to the class of kernel-based approaches, in which a non linear classifier is built after a mapping of the input space to a higher, possibly infinite, feature space via a function Φ. This is a well-known theory that exploits the Reproducing Kernel Hilbert Space (RKHS) framework [1], and recently has been applied with success by the machine learning community. Support Vector Machines (SVMs), designed by Vapnik [11], represent one of the most successful examples of a learning machine based on a kernel method. In practice, the kernel function $K(\cdot,\cdot)$ acts as an operator that hides the non linear map in the feature space, thus avoiding to work directly on ϕ. For classification purposes, the most used kernel functions are: $K(x_i, x_j) = x_i \cdot x_j$ (linear), $K(x_i, x_j) = (1 + x_i \cdot x_j / r)^p$ (polynomial) and $K(x_i, x_j) = \exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right)$ (Gaussian); p and σ^2 are usually indicated as the *hyperparameters* of the problem. The KP has been applied with success for classification and function approximation tasks [6], [7], and its generalization capabilities has been reported for example in [9]. It can be considered as the dual formulation of the Rosenblatt's learning algorithm which, as known, can be applied to any choice of the updating step η. The advantage of the dual formulation consists in the fact that one can exploits the theory of kernels, thus allowing an implicit non linear transformation; furthermore the choice $\eta = 1$ leads to the fact that $\alpha \in A$. Table 1 sketches the flow of our algorithm, called DKPP. We implemented the pocket algorithm [8] in order to accept errors on the training set. This is a crucial point for optimal model selection, as will be clear in the next sections. It is important to point out that the on-going research on kernel-based methods suggests that many classifiers perform comparably, provided that a good kernel has been chosen [9]; therefore classifiers like the DKPP, are very appealing and show their superiority when targeting VLSI implementations, as simple counter–based architectures can be designed. Figure 1 shows such an architecture, where $q_{ij} = y_i y_j K(x_i, x_j)$ and MSB_i indicates the Most Significant Bit of the accumulator, in a 2's complement coding. In this paper we explore the way of automatic tuning the model of the DKPP.

3 Uniform and data–depended bounds for the DKP

In general, the error rate (or prediction error), of a classifier can be defined as the probability for (x, y) drawn randomly over P that h is wrong, that is $h(\alpha, x) \neq y$:

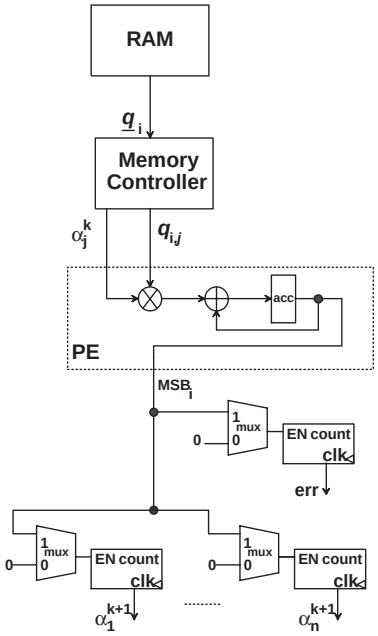

Fig. 1. A counter–based architecture for the DKPP.

$er_P\left(h\left(\alpha_{k,z},x\right),y\right)=\Pr\left\{(x,y)\in Z:h\left(\alpha_{k,z},x\right)\neq y\right\}$ [2], where $\alpha_{k,z}=\mathcal{A}_k\left(z\right)$ is the vector obtained by $\mathcal{A}_k$ after the observation of z and over the function class H_k. It is well known from the Probably Approximately Correct (PAC) learning framework that the behavior of er_P can be represented in terms of a bound which holds with a given probability; in practice, usually one asserts that the following inequality:

$$er_P\left(h\left(\alpha_{k,z},x\right),y\right)\leq \hat{er}_P\left(h_{k,z},z\right)+\varepsilon\left(m,d_k,\delta\right) \tag{1}$$

holds with a given probability $1-\delta$; $\hat{er}_P(h_{k,z},z)$ is the error rate of $h_{k,z}$ on z (also known as empirical error rate), that is:
$\hat{er}_P\left(h_{k,z},z\right)=\frac{1}{m}\left|i:1\leq i\leq m,h\left(\alpha_{k,z},x_i\right)\neq y_i\right|$ and d_k is a measure of the complexity of the function class H_k (for example the Vapnik–Chervonenkis dimension [11])[1]. Exploiting the well-known compression scheme by Warmuth et. al. [5], a uniform bound for the DKPP has been recently proposed [9]; according to such a theorem, with probability at least $1-\delta$, the prediction error of the DKPP, as long as $\hat{er}_P=0$, is:

$$er_z\left(h_{k,z}\right)\leq\frac{1}{m-d_k}\left(\ln\left[\binom{m}{d_k}\right]+\ln m+\ln\frac{1}{\delta}\right) \tag{2}$$

[1] For simplicity, during the text we will use the notation $er_P(h_{k,z})=er_P\left(h\left(\alpha_{k,z},x\right),y\right)$ and $\hat{er}_P(h_{k,z})=\hat{er}_P\left(h_{k,z},z\right)$, to indicate the prediction error and the empirical error rate of $h_{k,z}$; furthermore, given a set S with $|S|$ we intend its cardinality.

where d_k is the number of non zero alphas. We refer to eq. (2) as B_1. In the agnostic case ($\hat{er}_P \neq 0$), one can use a more general result which permits the calculus of ε in a data–dependent way, thanks to the introduction of a measure, called penalized complexity, that can be directly estimated on the basis of the given training set z [3]:

$$er_P\left(h_{k,z}\right) \leq \hat{er}_P\left(h_{k,z}\right) + d_k + \sqrt{\frac{9}{2m}\ln\frac{1}{\delta}} \tag{3}$$

where, in this case, $d_k = \max_{h \in H_k}\left(\hat{er}_P^{(1)}(h) - \hat{er}_P^{(2)}(h)\right)$, with:
$\hat{er}_P^{(1)}(h) = \frac{2}{m}\left|i : 1 \leq i \leq m/2 \leq m, h\left(\alpha, x_i\right) \neq y_i\right|$,
$\hat{er}_P^{(2)}(h) = \frac{2}{m}\left|i : m/2+1 \leq i \leq m, h\left(\alpha, x_i\right) \neq y_i\right|$. We refer to (3) as B_2. The measure d_k, can be easily computed by applying the DKPP on a new trining set z' obtained by splitting z in two halves and by flipping the labels of the second half [3]. In practice, it is easy to see that:

$$d_k = 1 - 2\hat{er}_P\left(h_{k,z'}, z'\right) \tag{4}$$

The main goal to design an *optimal* learning systems, is to bound the right side of equation (1), in order to bound the prediction error. This leads to the Structural Risk Minimization (SRM) principle; according to SRM one looks for a function h having lower empirical error over a fixed function class H_k, characterized by the complexity d. From a formal point of view, one should build several function classes in increasing size, that is increasing complexity ($H_1 \subset \cdots \subset H_k \subset \cdots$), and then pick a function h which has small training error and comes from a classes H_k having lower complexity (that is lower ε according to (1)). Our idea for adaptive model selection consists in building several H_k and then in measuring their *richness* on the basis of $B_{1,2}$. In practice we choose the function $h^* = \arg\min_k\left[\hat{er}_P\left(h_{k,z}\right) + \varepsilon\left(m, d_k, \delta\right)\right]$. The general procedure is sketched in table 2. Note that, theoretically, one could only use B_2. Actually, it is known that several function classes, such as the Gaussian one, have typically an high complexity, and the corresponding measure d_k, found according to the maximal discrepancy estimate, does not give any useful information on the problem ($d_k = 1$); on the other hand, in these cases we have $\hat{er}_P = 0$, thus allowing the use of B_1. Our experiments show that the joined investigation of $B_{1,2}$ can assure an optimal model selection for different real–world problems.

4 Experiments

We test our approach on two well–known datasets from [4, 10], that is sonar (SNR) and pima indian diabetes (PIMA), respectively. The results are summarized in table 3. Both data sets are two–class problems with $r = 60$ and $r = 8$ input features respectively, whilst the number of training samples are $m = 104$ and $m = 576$; TS indicates the number number of errors on 104 and 192 validation patterns respectively. We applied the model selection method to select

Table 1. The DKPP algorithm.

Step	Description
1.	set $\alpha = 0$, $\alpha_{opt} = 0$, $nerr = m$
2.	Repeat until no mistakes occur or a max number of steps has been reached
3.	for $i = 1$ to m do
4.	compute $\mathrm{MSB}_i = y_i \left(\mathrm{sgn} \left(\sum_j \alpha_j y_j K(x_i, x_j) \right) \right)$
5.	if $\mathrm{MSB}_i < 0$ then
6.	$\alpha_i = \alpha_i + 1$
7.	compute $err = \lvert i : y_i \left(\mathrm{sgn} \left(\sum_j \alpha_j y_j K(x_i, x_j) \right) \right) < 0 \rvert$
8.	if $err < nerr$ then $\alpha_{opt} = \alpha$, $nerr = err$
9.	end for
10.	$\alpha_{k,z} = \alpha_{opt}$

Table 2. A procedure for adaptive model selection.

Step	Description
1.	set $k = 1$
2.	Select a kernel function (es: linear, polynomial or gaussian)
3.	Select a value for the hyperparameter (build H_k)
4.	Apply DKKP and find $\alpha_{k,z} = \mathcal{A}_k(z)$, d_k
5.	Compute $B_{k,\{1,2\}} = \acute{e}r_P\left(h_{k,z}\right) + \varepsilon\left(m, d_k, \delta\right)$
6.	$k = k + 1$
7.	Choose $h^* = \arg\min_k \mathrm{B}_{k,\{1,2\}}$

Table 3. Experiments for the sonar and pima indian diabetes datasets.

	SNR	$\sigma^2 = 0.1$	$\sigma^2 = 0.5$	$\sigma^2 = 1.0$	$p = 2$	$p = 3$	$p = 4$
$\mathrm{B}_1(TS)$		1.86(11)	1.82(7)	1.96(10)	6.58(19)	4.8(15)	4.8(13)
$\mathrm{B}_2(TS)$		–	–	–	–	–	–
	PIMA	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$	$\sigma^2 = 0.5$	$p = 2$	$p = 3$	$p = 4$
$\mathrm{B}_1(TS)$		1.41(53)	1.6(50)	2.16(60)	–	–	–
$\mathrm{B}_2(TS)$		–	–	–	0.5(41)	0.51(40)	0.6(43)

one among two kernel functions (Gaussian and polynomial) and the associated hyperparameter (σ^2 or p). As expected, there is a substantial agreement between the bounds $B_{1,2}$ and TS; as reported in table 3, our criteria chooses the best model, that is a Gaussian kernel with $\sigma^2 = 0.5$ for SNR and a polynomial with $p = 2$ or $p = 3$ for PIMA. As a final remark note that with '–' we mean that corresponding bound does not give any useful information for model selection. For example, in the case of B_2, it means that $d_k = 1$ for every value of the hyperparameter, while in the case of B_1 it simply means that we cannot apply equation (2) as we are in presence of an agnostic case.

5 Conclusion

In this paper we have propose an efficient method for adaptive model selection. We used a digital kernel based perceptron with pocket, but our approach could be applied to other learning systems as well, as long as good bounds of the prediction error are provided; in particular, in this work we have investigated the use of two different kinds of bounds: a uniform bound, that can be applied to systems characterized by a high level of complexity, and a data–dependent bound. Experiments on two real world problems have demonstrated the validity of the method.

References

1. Aizerman, M. A., Braverman, E. M. and Rozonoer, L. I.: Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning. Automation and Remote Control, **25** (1964) 821-837.
2. Bartlett P.: Neural Networks Learning: Theoretical Foundations, Cambridge University Press, 1999.
3. Bartlett, P.L., Boucheron, S., and Lugosi, G.: Model Selection and Error Estimation. Machine Learning, **48** (2002), 85–113.
4. Blake, C., Keogh, E., and Merz, C.J.: UCI Repository of Machine Learning Databases, *http://www.ics.uci.edu/ mlearn/MLRepository.html*.
5. Floyd, S. and Warmuth, M.: Sample compression, learnability and the Vapnik–Chervonenkis dimension. Machine Learning, **21** (1995) 269–304.
6. Freund Y. and Shapire, R.E.: Large Margin Classification Using the Perceptron Algorithm. Machine Learning, **37** (1999) 277-296.
7. Friess, T.T. and Harrison, R.F.: A Kernel-Based Adaline for Function Approximation. Intelligent Data Analysis, **3** (1999) 307-313.
8. Gallant, S.I: Perceptron–Based Learning Algorithms. IEEE Transaction on Neural Networks, **1** (1990) 179–191.
9. Herbrich, R., Graepel, T., and Williamson, R.: From Margin to Sparsity. NIPS 13, 2001.
10. J.M. Torres Moreno, M.B. Gordon. Characterization of the Sonar Signals Benchmark. Neural Processing Letters, **7** (1998) 1–4.
11. Vapnik, V.N.: The Nature of Statistical Learning Theory. John Wiley & Sons, NY, USA, 2nd edition, 1999.