New Results on Error Correcting Output Codes of Kernel Machines

Andrea Passerini, Massimiliano Pontil, and Paolo Frasconi, *Member, IEEE*

***Abstract*—We study the problem of multiclass classification within the framework of error correcting output codes (ECOC) using margin-based binary classifiers. Specifically, we address two important open problems in this context: decoding and model selection. The decoding problem concerns how to map the outputs of the classifiers into class codewords. In this paper we introduce a new decoding function that combines the margins through an estimate of their class conditional probabilities. Concerning model selection, we present new theoretical results bounding the leave-one-out (LOO) error of ECOC of kernel machines, which can be used to tune kernel hyperparameters. We report experiments using support vector machines as the base binary classifiers, showing the advantage of the proposed decoding function over other functions of the margin commonly used in practice. Moreover, our empirical evaluations on model selection indicate that the bound leads to good estimates of kernel parameters.**

***Index Terms*—Error correcting output codes (ECOC), machine learning, statistical learning theory, support vector machines.**

I. Introduction

Many machine learning algorithms are intrinsically conceived for binary classification. However, in general, real world learning problems require that inputs are mapped into one of several possible categories. The extension of a binary algorithm to its multiclass counterpart is not always possible or easy to conceive (examples where this is possible are decision trees or prototypes methods such as k-nearest neighbors). An alternative consists in *reducing* a multiclass problem into several binary subproblems. A general reduction scheme is the information theoretic method based on error correcting output codes (ECOC), introduced by Dietterich and Bakiri [16] and more recently extended in [2]. The simplest coding strategy, sometimes called "one-hot" or "one-versus-all," consists in defining as many dichotomies of the instance space as the number of classes, where each class is considered as "positive" in one and only one dichotomy. Typically, the binary classifiers are trained independently but a few recent works [12], [23] considered also the case where classifiers are trained simultaneously. We focus on the former approach although some of our results may be extended to the latter one.

Dichotomies can be learned in different ways. In this paper, we are interested in the case of margin-based binary classifiers, as induced by a fairly large class of algorithms that include support vector machines (SVMs) [11], [33], but also classic methods such as the perceptron [30] and its variants [20]. They all learn a real-valued function $f(x)$ of an instance x, called the margin of x, and then take the sign of $f(x)$ to obtain classification. The theory developed by Vapnik [33] shows that when f belongs to a reproducing kernel Hilbert space $\mathcal{H}$, generalization is related to the norm of f in $\mathcal{H}$ or, equivalently, to the margin of f which can be defined as the inverse of its norm.[1] Therefore, it may be expected that methods such as SVMs, that attempt to maximize the margin, will achieve good generalization. This setting is general enough to accommodate nonlinear separation and nonvector data, provided that a suitable kernel function is used to define the inner product in $\mathcal{H}$, see [14], [18], [31], [33], [34].

When using margin-based classifiers to implement a set of dichotomies for multiclass problems, the input instance is first mapped to a real vector of margins formed by the outputs of the binary classifiers. A target class is then computed from this vector by means of a decoding function [16]. In this setting, we focus on two fundamental and complementary aspects of multiclassification, namely (1) which strategy should be used to "decode" the real vector of margins to obtain classification, and (2) how to study the generalization error of ECOC and use the results to estimate kernel hyperparameters.

Concerning the first aspect, early works assumed that the output of each binary classifier was a boolean variable, and the decoding strategy was based on the Hamming distance [16]. However, in the case that the binary learners are margin-based classifiers, Allwein *et al.* [2] showed the advantage of using a loss-based function of the margin. In this paper, we suggest a different approach which is based on decoding via conditional probabilities of the outputs of the classifiers. The advantages offered by our approach are twofold. First, the use of conditional probabilities allows to combine the margins of each classifier in a principled way. Second, the decoding function is itself a class conditional probability which can give an estimate of multiclassification confidence. We report experiments using support vector machines as the base binary classifiers, showing the advantage of the proposed decoding function over other functions of the margin commonly used in practice.

Concerning the second aspect of multiclassification with margin-based classifiers, we begin by observing that the kernel function typically depends on hyperparameters that are treated as constants by optimization approaches like SVMs. However, since they constitute additional degrees of freedom, their

Manuscript received October 25, 2002; revised May 14, 2003. A shorter version of this paper appeared in ECAI 2002.

A. Passerini and P. Frasconi are with the Department of Systems and Computer Science, University of Florence, Firenze, Italy (e-mail: passerini@dsi.unifi.it; paolo@dsi.unifi.it).

M. Pontil is with the Department of Computer Science, University College London, London WC1E, U.K. (e-mail: m.pontil@cs.ucl.ac.uk).

Digital Object Identifier 10.1109/TNN.2003.820841

[1]In the case of linearly separable data, the margin of f is equal to the minimum of $|f(x)|/\|f\|$ on the training set instances, where $\|f\|$ is the norm of f in the Hilbert space $\mathcal{H}$.

choice should be controlled in order to prevent overfitting. Determining hyperparameters is often distinguished from the estimation of the parameters that are optimized by a learning algorithm. The problem is also known as *model selection* in statistics and machine learning, where several early criteria have been proposed (see, e.g., [1], [13], [32]). Model selection usually consists in determining the value of a very small set of hyperparameters. In this case, it can be carried out by calling as a subroutine a learning algorithm that receives hyperparameters as constant input arguments. Recent methods for tuning several hyperparameters simultaneously include gradient descent [5] and sensitivity analysis [10]. The former method consist in choosing a differentiable model selection criterion and searching a global optimum in the joint space of parameters and hyperparameters. The latter works by iteratively minimizing an estimate of the generalization error of the support vector machine. The method proposed in this paper is based on a general bound on the leave-one-out (LOO) error in the case of ECOC of kernel machines. The bound can be directly used for estimating the optimal value of a small set of kernel parameters. The novelty of this analysis is that it allows multiclass parameters optimization even though the binary classifiers are trained independently. We report experiments showing that the bound leads to good estimates of kernel parameters.

The paper is organized as follows. In Section II we shortly review the theory of ECOC and the associated loss-based decoding methods. In Section III we introduce a new decoding function based on conditional probabilities and give a theoretical justification of its validity. In Section IV we extend the bound on the LOO error to the case of multiclassification. Finally, in Section V we empirically validate the usefulness of the theoretical results presented in the paper.

II. Background on ECOC

ECOC work in two steps: training and classification. During the first step, S binary classifiers are trained on S dichotomies of the instance space, formed by joining non overlapping subsets of classes. Assuming Q classes, let us introduce a "coding matrix" $\mathbf{M} \in \{-1, 0, 1\}^{Q \times S}$ which specifies a relation between classes and dichotomies. $m_{qs} = 1 (m_{qs} = -1)$ means that points belonging to class q are used as positive (negative) examples to train the sth classifier f_s. When $m_{qs} = 0$, points in class q are not used to train the sth classifier. Thus each class q is encoded by the qth row of matrix $\mathbf{M}$ which we denoted by $\mathbf{m}_q$. During prediction a new input x is classified by computing the vector formed by the outputs of the classifiers, $\mathbf{f}(x) = (f_1(x), \ldots, f_S(x))$ and choosing the class whose corresponding row is closest to $\mathbf{f}(x)$. In so doing, classification can be seen as a decoding operation and the class of input x is computed as

$$\arg \min_{q=1}^{Q} d\left(\mathbf{m}_q, \mathbf{f}(x)\right)$$

where d is the decoding function. In [16], the entries of matrix $\mathbf{M}$ were restricted to take only binary values and the d was chosen to be the Hamming distance

$$d(\mathbf{m}_q, \mathbf{f}) = \sum_{s=1}^{S} \frac{|m_{qs} - \mathrm{sign}(f_s)|}{2}. \tag{1}$$

When the binary learners are margin-based classifiers, [2] showed the advantage of using a loss-based function of the margin

$$d_L(\mathbf{m}_q, \mathbf{f}) = \sum_{s=1}^{S} L(m_{qs} f_s)$$

where L is a loss function. L is typically a nondecreasing function of the margin and, thus, weights the confidence of each classifier according to the margin. The simplest loss function one can use is the linear loss for which $L(m_{qs} f_s) = -m_{qs} f_s$. Several other choices are possible, although no formal results exist that suggest an optimal choice.

It is worthwhile noting that the ECOC framework includes two multiclass classification approaches often used in practice: one-versus-all and all-pairs. In the former approach there is one classifier per class, which separates it from all the others. A new input is assigned to the class whose associated classifier has the maximum output. In the ECOC framework one-versus-all is equivalent to linear decoding with a $Q \times Q$ coding matrix whose entries are always -1 except diagonal entries which are equal to 1. In the latter approach, also known as *pairwise coupling* [21] or *round robin classification* [22], there are $Q(Q-1)/2$ classifiers, each separating a pair of classes. Classification is decided by majority voting. This scheme is equivalent to Hamming decoding with the appropriate coding matrix.

When all binary classifiers are computed by the same learning algorithm, Allwein *et al.* [2] proposed to set L to be the same loss function used by that algorithm. For example, in the case of SVMs, this corresponds to the soft-margin loss function $L(m_{qs} f_s) = |1 - m_{qs} f_s|_+$, where $|x|_+ = x$ if $x > 0$ and zero otherwise (see Section IV-A). In the next section we suggest a different approach which is based on decoding via conditional probabilities of the outputs of the classifiers.

III. Decoding Functions Based on Conditional Probabilities

As aforementioned, a loss function of the margin may have some advantages over the standard Hamming distance because it can encode the confidence of each classifier in the ECOC. This confidence is, however, a relative quantity, i.e., the range of the values of the margin may vary with the classifier used. Thus, just using a linear loss function may introduce some bias in the final classification in the sense that classifiers with a larger output range will receive a higher weight. Not surprisingly, we will see in the experiments in Section V that the Hamming decoding usually works better than the linear one in the case of pairwise schemes. A straightforward normalization in some interval, e.g., $[-1, 1]$, can also introduce bias since it does not fully take into account the margin distribution. A more principled approach is to estimate the conditional probability of each class q given the input x. Given the S trained classifiers, we assume that all the information about x that is relevant for determining the class is contained in the margin vector $\mathbf{f}(x)$ (or $\mathbf{f}$ for short), i.e., $P(Y = q|x) = P(Y = q|\mathbf{f})$. Let us now introduce the set of all possible codewords $\mathbf{o}_k$, $k = 1, \ldots, 2^S$ and let $\mathbf{O}$ be a random vector of binary variables. A realization of $\mathbf{O}$ will be a codeword. For simplicity, we shall use the symbols -1 and $+1$ to denote codebits. The probability of Y given the margin vector can thus be rewritten by marginalizing out codewords and decomposing using the chain rule

$$P(Y=q|\mathbf{f}) = \sum_{k=1}^{2^S} P(Y=q|\mathbf{O}=\mathbf{o}_k,\mathbf{f})P(\mathbf{O}=\mathbf{o}_k|\mathbf{f}).$$

The above model can be simplified by assuming the class to be independent of $\mathbf{f}$ given the codeword $\mathbf{o}_k$. This assumption essentially means that $\mathbf{f}$ has a direct causal impact on $\mathbf{O}$, and in turn $\mathbf{O}$ has a direct causal impact on Y. As a result

$$P(Y=q|\mathbf{f}) = \sum_{k=1}^{2^S} P(Y=q|\mathbf{O}=\mathbf{o}_k)P(\mathbf{O}=\mathbf{o}_k|\mathbf{f}).$$

We choose a simple model for the probability of class q given the codeword $\mathbf{o}_k$ by looking at the corresponding row $\mathbf{m}_q$ in the coding matrix. A zero entry in the row is treated as "don't care," i.e., replacing it with a a value of 1 or -1 results in an equally correct codeword for the class. Thus each class q has a number of valid codes $\mathcal{C}_q$ given by all possible substitutions of 1 or -1 in the zero entries of $\mathbf{m}_q$. Invalid codes $\bar{\mathcal{C}}$ are those which are not valid for any class $q \in Q$. We then define

$$P(Y=q|\mathbf{O}=\mathbf{o}_k) = \begin{cases} 1, & \text{if } \mathbf{o}_k \in \mathcal{C}_q \\ 0, & \text{if } \mathbf{o}_k \in \mathcal{C}_{q'} \text{ with } q' \neq q \\ \frac{1}{Q}, & \text{otherwise (i.e., } \mathbf{o}_k \in \bar{\mathcal{C}} \text{ is not a valid class code).} \end{cases}$$

Under this model

$$P(Y=q|\mathbf{f}) = \sum_{\mathbf{o}_k \in \mathcal{C}_q} P(\mathbf{O}=\mathbf{o}_k|\mathbf{f}) + \alpha$$

where

$$\alpha = \frac{1}{Q}\sum_{\mathbf{o}_k \in \bar{\mathcal{C}}} P(\mathbf{O}=\mathbf{o}_k|\mathbf{f})$$

collects the probability mass dispersed on the invalid codes. We further assume that each individual codebit O_s is conditionally independent of the others given $\mathbf{f}$, and that it is also independent of the other outputs $f_{s'}$ given f_s (in other words, we are assuming that the only cause for O_s is f_s). Our conditional independence assumptions can be graphically described by the Bayesian network depicted in Fig. 1. As a result, we can write the conditional probability of the class q as

$$P(Y=q|\mathbf{f}) = \sum_{\mathbf{o}_k \in \mathcal{C}_q} \prod_{s=1}^{S} P(O_s = o_{ks}|f_s) + \alpha. \quad (2)$$

We further note that the probability of a bit corresponding to a zero value in the coding matrix is independent of the output of the classifier (it is a "don't care" bit). Moreover, it should be equally distributed between the possible realizations $\{-1, 1\}$. All valid codes $\mathbf{o}_k \in \mathcal{C}_q$ for a given class q have then the same probability

$$\prod_{s=1}^{S} P(O_s = o_{ks}|f_s) = \frac{1}{2^Z} \prod_{s \in S: m_{qs} \neq 0} P(O_s = m_{qs}|f_s)$$

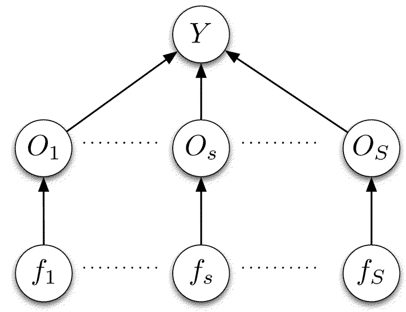


Fig. 1. Bayesian network describing the probabilistic relationships amongst margins, codewords, and class.

where Z is the number of zero entries in the row corresponding to the class. By noting that there are exactly 2^Z of such valid codes, we can simplify (2) to

$$P(Y=q|\mathbf{f}) = \prod_{s \in S: m_{qs} \neq 0} P(O_s = m_{qs}|f_s) + \alpha. \quad (3)$$

In this case, the decoding function will be

$$d(\mathbf{m}_q, \mathbf{f}) = -\log P(Y=q|\mathbf{f}). \quad (4)$$

The problem boils down to estimating the individual conditional probabilities in (3), a problem that has been addressed also in [26], [28]. Our solution consists of fitting the following set of parametric models

$$P(O_s = m_{qs}|f_s) = \frac{1}{1+\exp\{m_{qs}(A_s f_s + B_s)\}}$$

where A_s and B_s are adjustable real parameters that reflect the slope and the offset of the cumulative distribution of the margins. A_s and B_s can be estimated independently by maximizing the following set of Bernoulli log-likelihoods

$$\mathcal{L}_s = \sum_{i: m_{y_i s} \neq 0} \log\left\{\frac{1}{1+\exp\{m_{y_i s}(A_s f_s(x_i) + B_s)\}}\right\}. \quad (5)$$

The index i in (5) runs over the training examples (x_i, y_i). It must be observed that fitting the sigmoid parameters using the same examples used to train the margin classifiers would unavoidably lead to poor estimates since the distribution of $f_s(x_i)$ is very different for training and for testing instances (for example, in the case of separable SVMs, all the support vectors contribute a margin that is exactly $+1$ or -1). To address this, in our experiments we used a threefold cross-validation procedure to fit A_s and B_s, as suggested in [28].

We remark that an additional advantage of the proposed decoding algorithm is that the multiclass classifier outputs a conditional probability rather than a mere class decision.

IV. ECOC of Kernel Machines

In this section, we study ECOC schemes which use kernel machines as the underline binary classifier. Our main result is a bound on the LOO error of the ECOC with a general decoding function. We first recall the main features of kernel machines for binary classification.

A. Background on Kernel Machines

Let $D_\ell = \{(x_i, y_i)_{i=1}^\ell \in \{X \times \{-1,1\}\}^\ell$ be a training set and $V : \mathbb{R} \to \mathbb{R}$ a loss function. Kernel machines [14], [18], [31], [33] are the minimizers of functionals of the form

$$F[f; D_\ell] = \frac{1}{\ell}\sum_{i=1}^{\ell} V\left(y_i f(x_i)\right) + \lambda \|f\|_K^2 \tag{6}$$

where λ is a positive constant named regularization parameter. The minimization of functional in (6) is done in a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ defined by a symmetric and positive definite kernel $K : X \times X \to \mathbb{R}$, and $\|f\|_K^2$ is the norm of a function $f : X \to \mathbb{R}$ belonging to $\mathcal{H}$. This norm is a measure of smoothness, e.g., a Sobolev norm [15]. Thus, the regularization parameter $\lambda > 0$ trades off between small empirical error and smoothness of the solution. A correct choice of λ prevents from overfitting. In fact, this is theoretically justified by means of VC-theory [4], [18], [33]. For more information on RKHS see [3], [15], [19], [27], [34].

Assuming V is convex, the minimizer of functional in (6) is unique and has the form[2]

$$f(x) = \sum_{i=1}^{\ell} \alpha_i y_i K(x_i, x). \tag{7}$$

The coefficients α_i are computed by solving an optimization problem whose form is determined by the loss function V. For example, in SVMs [11], V is the soft-margin loss, $V(yf(x)) = |1 - yf(x)|_+$. In this case, the α_i are the solution of a quadratic programming problem with constraints $\alpha_i \in [0, 1/2\ell\lambda]$—see, e.g., [18]. A peculiar property of an SVM is that, usually, only few data points have nonzero coefficients α_i. These points are named support vectors. For more information on SVMs, see, e.g., [14], [31].

B. Bounds on the LOO Error

We now present our bound on the LOO error of ECOC of kernel machines.

We define the multiclass margin [2] of point $(x, y) \in X \times \{1, \ldots, Q\}$ to be

$$g(x,y) = d_L\left(\mathbf{m}_p, \mathbf{f}(x)\right) - d_L\left(\mathbf{m}_y, \mathbf{f}(x)\right)$$

with

$$p = \arg\min_{q \neq y} d_L\left(\mathbf{m}_q, \mathbf{f}(x)\right).$$

When $Q = 2$ and L is the linear loss, $g(x,y)$ reduces to twice the definition of margin for binary classification.[3] When $g(x,y)$ is negative, point x is misclassified. Thus, the empirical misclassification error can be written as

$$\frac{1}{\ell}\sum_{i=1}^{\ell} \theta\left(-g(x_i, y_i)\right)$$

where $\theta(\cdot)$ is the Heavyside function: $\theta(x) = 1$ if $x > 0$ and zero otherwise.

[2]We assume that the bias term is incorporated in the kernel K.

[3]Assuming $S = 2$ and $\mathbf{m}_1 = -\mathbf{m}_2 = \{1, -1\}$.

The LOO error is defined by

$$\frac{1}{\ell}\sum_{i=1}^{\ell} \theta\left(-g^i(x_i, y_i)\right)$$

where we have denoted by $g^i(x_i, y_i)$ the margin of example x_i when the ECOC is trained on the data set $D_\ell \setminus \{(x_i, y_i)\}$. The LOO error is an interesting quantity to look at when we want to find the optimum hyperparameters of a learning machine, as it is an almost unbiased estimator for the test error (see, e.g., [33]).[4]

Unfortunately, computing the LOO error is time demanding when ℓ is large. This becomes practically impossible in the case that we need to know the LOO error for several values of the parameters of the machine used. In the case of binary SVMs, bounds on the LOO error were studied in [10]—see also [17] and references therein.

In the following theorem we give a bound on the LOO error of ECOC of kernel machines. An interesting feature is that the bound only depends on the solution of the machines trained on the full data set (so training the machines once will suffice). Below we denote by f_s the s-machine, $f_s(x) = \sum_{i=1}^{\ell} \alpha_i^s m_{y_i s} K^s(x_i, x)$, and let $G_{ij}^s = K^s(x_i, x_j)$. We first present the result in the case of linear decoding.

Theorem 4.1: Suppose the linear decoding function $d_L(\mathbf{m}_q, \mathbf{f}) = -\mathbf{m}_q \cdot \mathbf{f}$ is used, where $\cdot$ denotes the inner product. Then, the LOO error of the ECOC of kernel machines is bounded by

$$\frac{1}{\ell}\sum_{i=1}^{\ell} \theta\left(-g(x_i, y_i) + \max_{q \neq y_i} U_q(x_i)\right) \tag{8}$$

where we have defined the function

$$U_q(x_i) = (\mathbf{m}_q - \mathbf{m}_p) \cdot \mathbf{f}(x_i) + \sum_{s=1}^{S} m_{y_i s}\left(m_{y_i s} - m_{qs}\right) \alpha_i^s G_{ii}^s \tag{9}$$

with $p = \arg\max_{q \neq y_i} \mathbf{m}_q \cdot \mathbf{f}(x_i)$.

The proof is postponed to the Appendix. The theorem says that point x_i is counted as a LOO error when its multiclass margin is smaller than $\max_{q \neq y_i} U_q(x_i)$. This function is always larger or equal than the positive value

$$\sum_{s=1}^{S} m_{y_i s}(m_{y_i s} - m_{ps}) \alpha_i^s G_{ii}^s.$$

Roughly speaking, this value is controlled by two factors: the parameters α_i^s, $s = 1, \ldots, S$ (where each parameter indicates if point x_i is a support vector for the sth kernel machine) and the Hamming distance between the correct codeword $\mathbf{m}_{y_i}$ and the closest codeword to it $\mathbf{m}_p$.

Theorem 4.1 also enlightens some interesting properties of the ECOC of kernel machines which we briefly summarized in the following.

[4]We remark that using the LOO error to select the model parameters presents an important problem, namely that the variance of this estimator may be large, see, e.g. [9]. More generally, a k-fold cross validation estimator may have large variance when k is small, see [24]. On the contrary there is little bias in the estimator. So, ideally, k should be selected in order to minimize the sum of the bias and variance, which is highly time consuming. Fortunately, in many practical situations, as also our experiments below indicate, using the LOO error estimator this is not much of a problem.

- **Relation between the regularization parameter and LOO error**
  Assume that, for every $s = 1, \ldots, S$, $\alpha_i^s \in [0, C]$ with $C = 1/(2\ell\lambda)$ (see, e.g., [18]). In this case
  $$\max_{q \neq y_i} U_q(x_i) \leq (S-1)C\kappa$$
  where $\kappa = \max_s \max_i G_{ii}^s$. Thus, the smaller C is, the closer the LOO error to the empirical error will be. In particular, if $C\kappa \ll 1$ the LOO would typically not deviate much from the empirical error.
- **Stability of the ECOC schemes**
  One-versus-all schemes are more stable than other ECOC schemes, meaning that their multiclass margin is less affected by removing one point in that case. In fact, note that in the one-versus-all scheme each pair of rows has only two different elements, so when one point is removed, the bound in Theorem 4.1 implies that the margin will not change of more than $2C$. For pairwise schemes, instead, the worst change is $(Q-1)C$. For dense codes the situation is even worse: the worst case is $(S-1)C$. This observation provides some insights on why the simple one-versus-all SVMs works well in practice.
- **One-versus-all schemes**
  For one-versus-all schemes we easily see that
  $$U_q(x_i) = 2\left(f_q(x_i) - f_p(x_i)\right) + 2\left(\alpha_i^{y_i} + \alpha_i^q\right).$$
  This has a simple interpretation: when x_i is removed from the training set, its margin is bounded by the margin obtained if the classifier of that point, f_{y_i}, was penalized by $\alpha_i^{y_i}$ while the remaining classifiers f_q, $q \neq y_i$, increased their margin of α_i^q.

Finally, we note that the bound in (8) is close the LOO error when the parameter C used to train the kernel machine is "small," i.e., when $C\kappa < 1$ for every $s = \{1, \ldots, S\}$. Improving the bound when this condition does not hold is an open problem.

Theorem 4.1 can be extended to deal with other decoding functions provided that they are monotonic nonincreasing. This is formalized in the next corollary, whose proof is in the Appendix.

Corollary 4.1: Suppose the loss function L is monotonic non increasing. Then, the LOO error of the ECOC of kernel machines is bounded by

$$\frac{1}{\ell}\sum_{i=1}^{\ell}\theta\Bigg(L\left(m_{y_i s} f_s(x_i) - \alpha_i^s G_{ii}^s m_{y_i s}^2\right. - \min_{q \neq y_i}\sum_{s=1}^{S} L\left(m_{qs} f_s(x_i) - \alpha_i^s G_{ii}^s m_{y_i s} m_{qs}\right)\Bigg)\Bigg). \tag{10}$$

Note that the corollary applies to all decoding functions used in the paper.

V. Experiments

The proposed methods are validated on ten data sets from the UCI repository [6]. Their characteristics are shortly summarized in Table I. Continuous attributes were linearly normalized between zero and one, while categorical attributes where "one-hot" encoded, i.e., if there are D categories, the dth category is represented by a D-dimensional binary vector having the dth coordinate equal to 1 and all remaining coordinates equal to zero.

TABLE I
CHARACTERISTICS OF THE DATA SETS USED

Name	Classes	Train	Test	Inputs
Anneal	5	898	-	38
Ecoli	8	336	-	7
Glass	6	214	-	9
Letter	26	15000	5000	16
Optdigits	10	3823	1797	64
Pendigits	10	7494	3498	16
Satimage	6	4435	2000	36
Segment	7	1540	770	19
Soybean	19	683	-	35
Yeast	10	1484	-	8

A. Comparison Between Different Decoding Functions

We trained ECOC using SVMs as the base binary classifier,[5] with a fixed value for the regularization parameter given by the inverse of the training set average of $K(x,x)$. In our experiments, we compared our decoding strategy to Hamming and other common loss-based decoding schemes (linear, and the soft-margin loss used to train SVMs) for three different types of ECOC schemes: one-versus-all, all-pairs, and dense matrices consisting of $3Q$ columns of $\{-1, 1\}$ entries.[6] SVMs were trained on a Gaussian kernel $K(x,t) = \exp\{-\gamma\|x-t\|^2\}$. In order to avoid the possibility that a fortuitous choice of the parameter γ could affect our results we carried out an extensive series of experiments where we compared the test error of the four decoding schemes considered for 11 different values of γ.

Results are summarized in Figs. 2 and 3. For data sets with less than 2000 instances (Fig. 2) we estimated prediction accuracy by a 20-fold cross-validation procedure. For the larger data sets (Fig. 3), we used the original split defined in the UCI repository except in the case of letter, where we used the split of 15 000–5000.

Our likelihood decoding works better for all ECOC schemes and for most values of γ, and is often less sensitive to the choice of the kernel hyperparameter.

Another interesting observation is that the Hamming distance works well in the case of pairwise classification, while it performs poorly with one-versus-all classifiers. Both results are not surprising: the Hamming distance corresponds to the majority vote, which is known to work well for pairwise classifiers [21] but does not make much sense for one-versus-all because in this case ties may occur often.

The behavior of all curves shows that tuning kernel parameters may significantly improve performance. We also note that a simple encoding scheme such as one-versus-all performs well with respect to more complex codes. This seems to be due to

[5]Our experiments were carried out using SVM^{Light} [35].

[6]Dense matrices were generated using a variant of the BCH algorithm [7] realized by Dietterich [16].

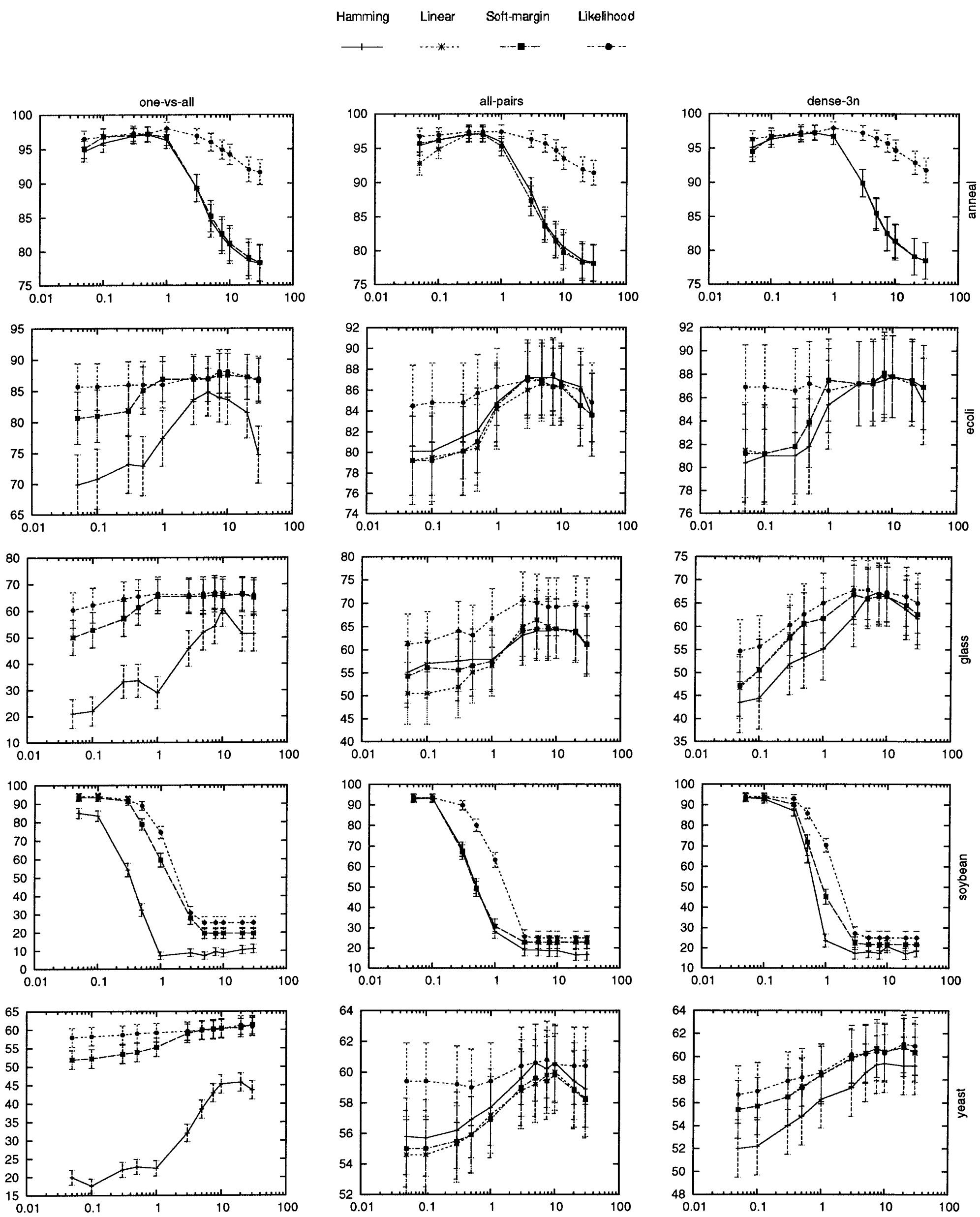


Fig. 2. Test error plotted against kernel hyperparameter γ. Data sets anneal, ecoli, glass, soybean, yeast.

the fact that binary SVM trained with Gaussian kernel provide complex decision functions, and one should not use a complex EOOC unless some prior knowledge indicates to do so. Similar results were observed for text classification [29].

B. Model Selection Experiments

We now show experiments where we use the bound presented in Section IV-B to select optimal kernel parameters. We focused on the datasets with more than 2000 instances, and searched for the best value of the γ hyperparameter of the Gaussian kernel. To simplify the problem we searched for a common value for all binary classifiers among a set of possible values. Plots in Fig. 4 show the test error and our LOO estimate for different values of γ for the three ECOC schemes discussed in the previous section. Notice that the minimum of the LOO estimate is very close to the minimum of the test error, although we often observed a slight bias toward smaller values of the variance.

VI. CONCLUSION

We studied ECOC constructed on margin-based binary classifiers under two complementary perspectives: the use of conditional probabilities for building a decoding function, and the use of a theoretically estimated bound on the LOO error for optimizing kernel parameters. Our experiments show that transforming margins into conditional probabilities helps recalibrating the outputs of the classifiers, thus improving the overall multiclass classification accuracy in comparison to other loss-based decoding schemes. At the same time, kernel parameters can be effectively adjusted by means of our LOO error bound. This further improves classification accuracy.

The probabilistic decoding method developed here assumes a fixed coding matrix for mapping codewords to classes. This choice also fixes the conditional probability distribution of the class given the codeword, although in a more general setting this conditional probability could be left unspecified and learned from data.

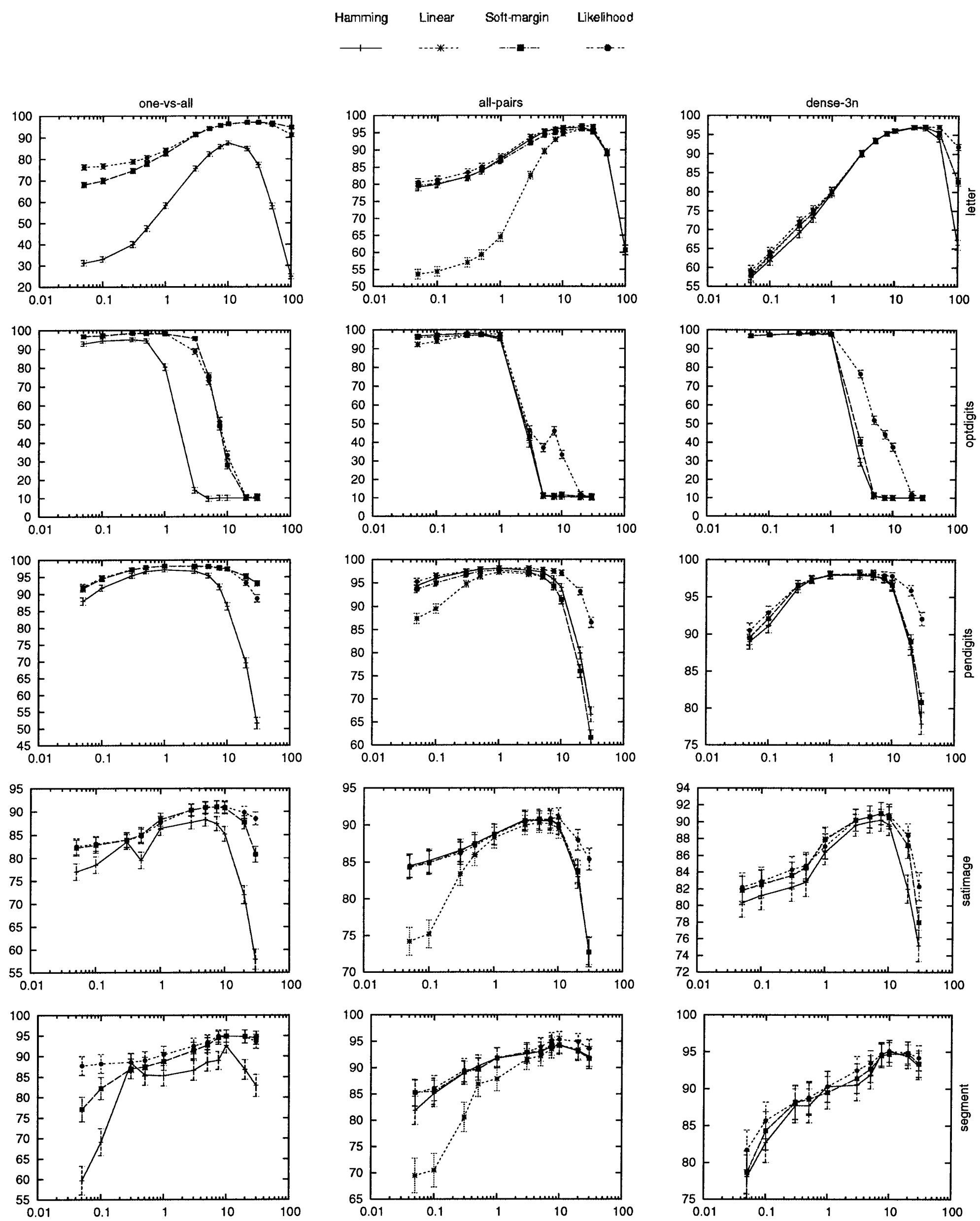


Fig. 3. Test error plotted against kernel hyperparameter γ. Data sets letter, optdigits, pendigits, satimage, segment.

The LOO bound presented in this paper could be smoothed into a differentiable function, enabling the application to the optimization of several hyperparameters simultaneously. An interesting future study in this sense is to use the derived LOO bound to perform feature selection. From a theoretical viewpoint it will be also interesting to study generalization error bounds of the ECOC of kernel machines. It should be possible to use our result within the framework of stability and generalization introduced in [8].

APPENDIX

We present here the proofs of the results presented in Section IV. To this end, we first need the following lemma.

Lemma 1.1: Let f be the kernel machine as defined in (7) obtained by solving (6). Let f^i be the solution of (6) found when the data point (x_i, y_i) is removed from the training set. We have

$$y_i f(x_i) - \alpha_i G_{ii} \leq y_i f^i(x_i) \leq y_i f(x_i). \tag{11}$$

Proof: The left-hand side (LHS) of (11) was proved in [25]. Note that, if x_i is not a support vector, $\alpha_i = 0$ and $f = f^i$, so both inequalities are trivial in this case. Thus suppose that x_i is a support vector. To prove the right-hand side (RHS) inequality we observe that $F[f; D_\ell] \leq F[f^i; D_\ell]$, and $-F[f; D_\ell^i] \leq -F[f^i; D_\ell^i]$. By combining the two bounds and using the definition of H we obtain that

$$V(y_i f(x_i)) \leq V(y_i f^i(x_i)).$$

Then, the result follows from the fact that V is monotonic. □

Proof of Theorem 4.1: Our goal is to bound $g^i(x_i, y_i)$—the multiclass margin of point i when this is removed from the training set—in terms of the α_i parameters obtained by training once the machines on the full training set. We have

$$g^i(x_i, y_i) = \mathbf{m}_{y_i} \cdot \mathbf{f}^i(x_i) - \mathbf{m}_{p^i} \cdot \mathbf{f}^i(x_i)$$

where we have defined

$$p^i = \arg\max_{q \neq y_i} \mathbf{m}_q \cdot \mathbf{f}^i(x_i).$$

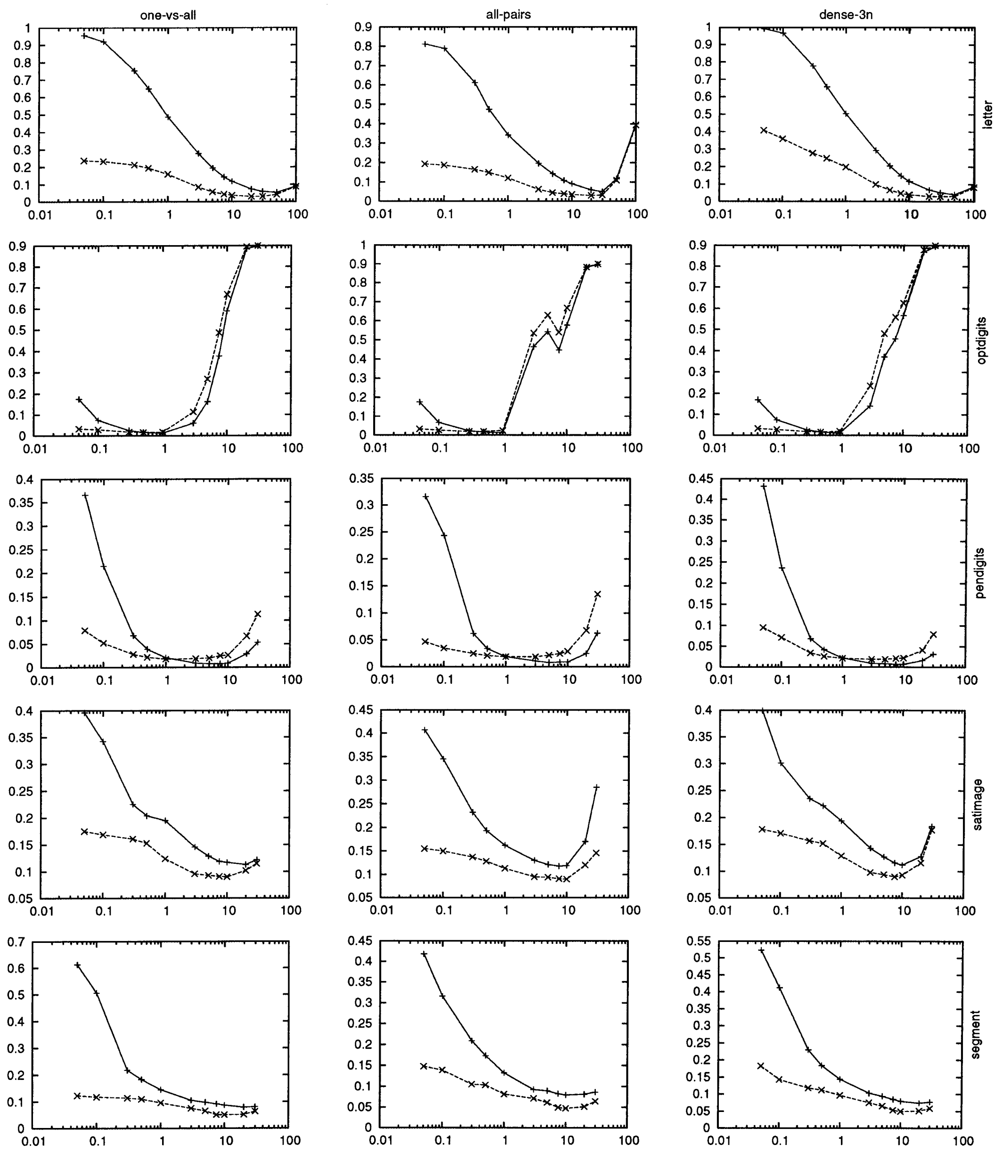


Fig. 4. Empirical comparison between test error (dashed line) and the LOO (solid line) bound of Corollary 4.1. The likelihood decoding function is used in all the experiments.

By applying Lemma 1.1 simultaneously to each kernel machine used in the ECOC procedure, Inequality (11) can be rewritten as

$$f_s^i(x_i) = f_s(x_i) - \lambda_s m_{y_i s}, \quad s \in \{1, \ldots, S\}$$

where λ_s is a parameter in $[0, \alpha_i^s G_{ii}^s]$. Using the above equation, we have

$$\begin{aligned} g^i(x_i, y_i) &= \sum_{s=1}^{S} \left(m_{y_i s} - m_{p^i s}\right) f^i(x_i) \\ &= \sum_{s=1}^{S} \left[\left(m_{y_i s} - m_{p^i s}\right) f_s(x_i) \right. \\ &\qquad \left. - m_{y_i s}\left(m_{y_i s} - m_{p^i s}\right) \lambda_s\right] \\ &\geq \sum_{s=1}^{S} \left[\left(m_{y_i s} - m_{p^i s}\right) f_s(x_i) \right. \\ &\qquad \left. - m_{y_i s}\left(m_{y_i s} - m_{p^i s}\right) \alpha_i^s G_{ii}^s\right]. \end{aligned}$$

Last inequality follows from the observation that $m_{y_i s}(m_{y_i s} - m_{p^i s})$ is always nonnegative. From the same inequality, we have

$$\begin{aligned} g^i(x_i, y_i) - g(x_i, y_i) \geq \sum_{s=1}^{S} &\left[\left(m_{ps} - m_{p^i s}\right) f_s(x_i) \right. \\ &\left. - m_{y_i s}\left(m_{y_i s} - m_{p^i s}\right) \alpha_i^s G_{ii}^s\right] \end{aligned}$$

from which the result follows. □

Proof of Corollary 4.1: Following the main argument in the proof of Theorem 4.1, the multiclass margin of point i when this is removed from the training set is bounded as

$$\begin{aligned} g^i(x_i, y_i) \geq \min_{q \neq y_i} \Bigg\{ \min_{\lambda} \sum_{s=1}^{S} &L\left(m_{qs} f_s(x_i) - \lambda_s m_{y_i s} m_{qs}\right) \\ &- L\left(m_{y_i s} f_s(x_i) - \lambda_s m_{y_i s}^2\right) \Bigg\} \end{aligned}$$

where $\min_\lambda$ is a shorthand for the minimum with respect to $\lambda_s \in [0, \alpha_i^s G_{ii}^s]$, for $s = 1, \ldots, S$. This minimum may be difficult to compute. However, is it easy to verify that when the loss function L is monotonic nonincreasing, the minimum is always achieved at the right border

$$g^i(x_i, y_i) \geq \min_{q \neq y_i} \sum_{s=1}^{S} L\left(m_{qs} f_s(x_i) - \alpha_i^s G_{ii}^s m_{y_i s} m_{qs}\right) - L\left(m_{y_i s} f_s(x_i) - \alpha_i^s G_{ii}^s m_{y_i s}^2\right).$$

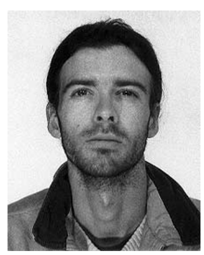
This concludes the proof. □

ACKNOWLEDGMENT

We wish to thank T. Dietterich for providing us with the BCH code.

REFERENCES

[1] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Proc. 2nd Int. Symp. Information Theory*, 1973, pp. 267–281.
[2] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: a unifying approach for margin classifiers," in *Proc. 17th Int. Conf. Machine Learning*. San Francisco, CA: Morgan Kaufmann, 2000, pp. 9–16.
[3] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 686, pp. 337–404, 1950.
[4] P. Bartlett and J. Shawe-Taylor, "Generalization performance of support vector machine and other patern classifiers," in *Advances in Kernel Methods—Support Vector Learning*, B. Scholkopf and C. Burges, Eds. Cambridge, MA: MIT Press, 1998.
[5] Y. Bengio, "Gradient-based optimization of hyper-parameters," *Neural Computation*, vol. 12, no. 8, pp. 1889–1900, 2000.
[6] C. L. Blake and C. J. Merz, UCI Repository of Machine Learning Databases, 1998.
[7] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Inform. Contr.*, vol. 3, pp. 68–79, Mar. 1960.
[8] O. Bousquet and A. Elisseeff, "Stability and generalization," *J. Machine Learning Res.*, vol. 2, pp. 499–526, 2002.
[9] P. Burman, "A comparative study of ordinary cross validation, 5-fold cross validation, and the repeated learning testing methods," *Biometrica*, vol. 76, no. 3, pp. 503–514, 1989.
[10] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing kernel parameters for support vector machines," *Machine Learning*, vol. 46, no. 1–3, pp. 131–159, 2002.
[11] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 1–25, 1995.
[12] K. Crammer and Y. Singer, "On the learnability and design of output codes for multiclass problems," in *Computat. Learning Theory*, 2000, pp. 35–46.
[13] P. Craven and G. Wahba, "Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross validation," *Numer. Math*, vol. 31, pp. 377–403, 1979.
[14] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*. Cambridge, U.K.: Cambridge Univ. Press, 2000.
[15] F. Cucker and S. Smale, "On the mathematical foundations of learning," *Bulletin (New Series) of the Amer. Math.Soc.*, vol. 39, no. 1, pp. 1–49, 2001.
[16] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *J. Artific. Intell. Res.*, vol. 2, pp. 263–286, 1995.
[17] A. Elisseeff and M. Pontil *et al.*, "Leave-one-out error and stability of learning algorithms with applications," in *NATO-ASI Series on Learning Theory and Practice*, J. Suykens *et al.*, Eds: IOS Press, 2002.
[18] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," *Adv. Computat. Math.*, vol. 13, pp. 1–50, 2000.
[19] C. H. FitzGerald, C. A. Micchelli, and A. Pinkus, "Functions that preserves families of positive definite functions," *Linear Alg. its Applicat.*, vol. 221, pp. 83–102, 1995.
[20] Y. Freund and R. E. Schapire, "Large margin classification using the perceptron algorithm," in *Computat. Learning Theory*, 1998, pp. 209–217.
[21] J. H. Friedman, "Another approach to polychotomous classification," Dept. Statistics, Stanford Univ., Tech. Rep., 1996.
[22] J. Fürnkranz, "Round robin classification," *J. Machine Learning Research*, vol. 2, pp. 721–747, 2002.
[23] Y. Guermeur, A. Elisseeff, and H. Paugam-Mousy, "A new multi-class SVM based on a uniform convergence result," in *Proc. IEEE IJCNN—Int. Joint Conf. Neural Networks*, 2000.
[24] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning: data mining, inference, and prediciton," in *Springer Series in Statistics*. New York: Springer-Verlag, 2002.
[25] T. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," in *Proc. Neural Inform ation Processing Conf.*, 1998.
[26] J. Kwok, "Moderating the outputs of support vector machine classifiers," *IEEE Trans. Neural Networks*, vol. 10, pp. 1018–1031, 1999.
[27] C. A. Micchelli, "Interpolation of scattered data: distance matrices and conditionally positive definite functions," *Construct. Approximat.*, vol. 2, pp. 11–22, 1986.
[28] J. Platt, "Probabilistic outputs for support vector machines and comparison to regularized likelihood methods," in *Advances in Large Margin Classiers*, A. Smola, P. Bartlett, B. Scholkopf, and D. Schurmans, Eds. Cambridge, MA: MIT Press, 1999.
[29] J. D. M. Rennie and R. Rifkin, "Improving multiclass text classification with the support vector machine," MIT, Tech. Rep. 2001-026, 2001.
[30] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain," *Psycholog. Rev.*, vol. 65, pp. 386–408, 1958.
[31] B. Schoelkopf and A. J. Smola, *Learning With Kernels*. Cambridge, MA: MIT Press, 2002.
[32] V. N. Vapnik, *Estimation of Dependences Based on Empirical Data*. Berlin, Germany: Springer-Verlag, 1982.
[33] ——, *Statistical Learning Theory*. New York: Wiley, 1998.
[34] G. Wahba, "Splines models for observational data," in *Series in Applied Mathematics*. Philadelphia, PA: SIAM, 1990, vol. 59.
[35] T. Joachims, "Making large-scale (SVM) learning practical," in *Advances in Kernel Methods—Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. Cambridge, MA: MIT Press, 1998, ch. 11, pp. 169–185.



Andrea Passerini received the M.Sc. degree in computer engineering in 2000 from the University of Florence, Florence, Italy.

He is currently a Ph.D. student working with the Machine Learning and Neural Networks group, University of Florence. His research interests include web research and focused crawling, kernel methods, support vector machines and multiclass classification, machine learning techniques applied to bioinformatics, especially protein structure prediction and gene expression analysis.

Massimiliano Pontil received the Bachelor and Ph.D. degrees in physics from the University of Genova, Genova, Italy, in 1994 and 1999, respectively.

He is currently a Lecturer with the Department of Computer Science and an Honorary Lecturer in the Gatsby Unit at the University College London. His research interests involve learning theory and its relation to pattern recognition, statistics, and approximation theory. Previously, he has been a visiting student with the Massachusetts Institute of Technology (MIT), Cambridge, in 1998, a Postdoctoral Fellow in 1999 and 2000 with the Center for Biological and Computational Learning at MIT, a Research Fellow at City University of Hong Kong in 2001 and 2002, where he is still affiliated, and a Research Associate with the Department of Information Engineering of the University of Siena in 2001 and 2002. He has also spent some time as a visiting researcher at the RIKEN Brain Science Institute, Tokyo, and AT&T Labs.. He has published more than 50 papers in international journals and conferences on different aspects of learning theory, machine learning, and computer vision.

Paolo Frasconi (S'91–M'96) received the M.Sc. degree in electronic engineering in 1990, and the Ph.D. degree in computer science in 1994, both from the University of Florence, Italy.

He is presently an Associate Professor of Computer Science with the University of Florence. He previously held positions with the University of Cagliari, Italy, the University of Wollongong, Australia, and the Massachusetts Institute of Technology, Cambridge. His current research interests are in the area of machine learning using connectionist models and belief networks, with particular emphasis on problems involving learning about sequential and structured information. Application fields of his interest include bioinformatics, natural language processing, pattern recognition, and document processing.

Dr. Frasconi serves as an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS, the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and the ACM Transactions on Internet Technology. In 2001, he codirected the NATO Advanced Studies Institute "Artificial Intelligence and Heuristic Methods for Bioinformatics." He is a member of the ACM, the IAPR, and the AI*IA.