Machine Learning manuscript No.
(will be inserted by the editor)

Pruning of Error Correcting Output Codes by Optimization of Accuracy-Diversity Trade off

Süreyya Özöğür Akyüz · Terry Windeatt · Raymond Smith

Received: date / Accepted: date

Abstract Ensemble learning is a method of combining learners to obtain more reliable and accurate predictions in supervised and unsupervised learning. However the ensemble sizes are sometimes unnecessarily large which causes extra memory usage, computational costs and decrease effectiveness. To overcome such side effects, pruning algorithms have been developed which are combinatorial but finding the exact subset of ensembles is computationally infeasible. Different types of heuristic algorithms are developed to obtain an approximate solution but they are lacking of a theoretical guarantee. Error Correcting Output Code (ECOC) is one of the well known ensemble techniques for multiclass classification which combines the outputs of binary base learners to predict the classes for multiclass data. In this paper, we propose a novel approach for pruning ECOC matrix by utilizing accuracy and diversity information simultaneously. All existing pruning methods need the size of the ensemble as a parameter, so the performance of the pruning methods depends on the size of ensemble. Our unparametrized pruning method is novel as being independent of the size of ensemble. Experimental results show that our pruning method is mostly better than other existing approaches.

S. Özöğür-Akyüz
University of Surrey, Center for Vision, Speech and Signal Processing, GU2 7XH, Surrey, UK.
Bahçeşehir University, Department of Mathematics and Computer Science, Istanbul, Turkey
E-mail: s.akyuz@surrey.ac.uk, sureyya.akyuz@bahcesehir.edu.tr

T. Windeatt
University of Surrey, Center for Vision, Speech and Signal Processing, GU2 7XH, Surrey, UK E-mail: T.windeatt@surrey.ac.uk

R. Smith
University of Surrey, Center for Vision, Speech and Signal Processing, GU2 7XH, Surrey, UK E-mail: Raymond.Smith@surrey.ac.uk

Keywords Ensemble learning, · Ensemble Pruning, · Error Correcting Output Codes, · DC programming, · Support Vector Machines, · Integer Programming

1 Introduction

It is widely accepted that ensemble classifiers are more accurate than a single classifier [47]. A number of effective ensemble methods have been developed over the previous decades, e.g. bagging [6] and boosting [15]. There is no integrated theory behind ensemble methods and several authors stated that there is no available consistent and theoretically sound explanations for ensemble classifiers [24]. Despite these negative reflections, at least three theories are able to explain the effectiveness of ensemble learners. The first theory is based on large margin classifiers [30]. It is shown that ensemble classifiers enlarge the margins and improve the capability of generalization performance of Output coding [1] which comes from Vapnik's Statistical Learning Theory [40]. The second theoretical argument is on bias-variance decomposition of error which states that ensemble classifiers reduce the variance or both bias and variance [7, 23, 35]. The last general theory is constructed upon a set theoretical point of view to remove all algorithmic details of classifiers and training procedures in which the classifiers are considered as sets of points [21, 22].

The effectiveness of ensemble methods relies on the diversity of the classifiers and accurate learning models. Despite their effectiveness, they may require extensive memory to save all the learning models and it can be time consuming to get a prediction on unlabeled test data. For small data sets, these two costs can be negligible, but they may be prohibitive when ensemble methods are applied to large scale data sets. Furthermore, the size of ensemble may lead to inefficiency and selecting subsets of classifiers can improve the generalization performance [47].

Several pruning algorithms have been developed such as ranking classifiers according to their individual performance and picking the best one. The major problem in selecting the best subset for the ensemble occurs when optimizing some criteria of subsets since it turns out to be a greedy search algorithm which has no theoretical or empirical quality guarantee [47]. In [10], a novel approach is proposed to investigate a given base classifier's effectiveness by measuring its accuracy k times with respect to each individual class of a k class problem, averaging the results. The proposed measure is used to prune the ensemble by using ordered aggregation and referred to ACEC. Since pruning is performed by ordered aggregation, as the size of ECOC matrix increase, running time of ACEC increases. Like in the other ordered aggregation methods, pruning size should be given as an input which leads the problem parameter dependent in [10]. Evolutionary pruning methods for ensemble selection are developed in [20]. Unlike the heuristic methods we mentioned above, ensemble pruning may be formulated as quadratic integer programming problem and solved by semi-definite programming which searches for optimal diversity-accuracy trade off [47]. This new approach differs from previous ensemble methods as it consists of discrete variables and can be considered as the discrete version of weighted ensemble optimization. The optimization problem contains a parameter k which is the ensemble size of the selected subset. Since it must be chosen beforehand, the solution, here, also depends on this parameter and hence the accuracy of the ensemble.

Similar optimization-based approach is proposed in [44] which considers accuracy and diversity measures as in [47] by a continuous optimization model with sparsity learning. The objective function in [44] is defined by a loss function and is chosen specifically to be the least square function. Weights for each classifier are forced to be sparse by L_1 norm regularization within the constraints to determine subset of an ensemble and weights of classifiers in the ensemble are found out by minimizing the error rate by least square function defined in the objective function. Diversity is taken into account with an additional diversity function defined by Yule's Q statistics within the constraints of the optimization problem. Since the model in [44] is defined by least squares function which contains true label vectors, it cannot be applied to ECOC framework directly. Because columns of ECOC matrix provide different problems with different relabellings. Yin et al. (2014) then further improved their work by penalizing sparsity and diversity constraints within objective function in [45]. In [45], error is minimized by the same idea by least square function.

In this paper, unlike pruning methods in [10] and [47], pruning size is not required as an input in the problem. Our first contribution is the reformulation of the quadratic integer formulation of [47] as an unconstrained problem by approximating cardinality constraint which refers to the ensemble size defined by zero norm approximation. Therefore, the problem becomes not only parametric-free of ensemble size but also it results in a non convex continuous optimization problem. The non convex problem here is reformulated by difference of convex functions as in [38] and solved by nonlinear programming solver function **fiminunc** in MATLAB's optimization toolbox [5]. In this paper, the overall ensemble pruning problem is adapted to ECOC which is the second novel contribution of this study. The performance of our proposed approach *ECOC with UP* is compared with well known methods Reduced Error Pruning, Random Guessing, Kappa Pruning and recent approaches ACEC in [10], SDP in [47].

Unlike with the methods mentioned above, subclass technique (subECOC) is developed in [3] on the ECOC framework where by splitting the initial classes of the problem, larger but easier problem to solve ECOC configurations. The multi-class problem's decomposition is performed by discriminant tree approach.

Our proposed method has common goals and optimization structure with the method in [44] and [45] with its diversity and sparsity notions. However, expression of accuracy, diversity and sparsity differs by their respective objective function and constraints. Furthermore it is developed to prune ECOC matrix. One of the important aspect of our algorithm is that it can be applied to ECOC framework but it is not possible to adapt objective function in [44] and [45] to ECOC framework because of the loss function term. It should be noted that ECOC columns represent different labelings which constitute different binary problem and hence do not agree with the loss term in [44] and [45].

The proposed framework is novel, since optimising the accuracy/diversity trade-off for ECOC through pruning by using optimization has not previously been attempted. The closest approach is [47], but that is restricted to two-class problems, and is a discrete optimization. Furthermore our approach automatically determines the optimal pruning rate as part of the optimisation, and therefore does not need to be supplied as a parameter in advance. In [10], ECOC is pruned by ordered aggregation which achieves high accuracy but on the other hand it slows down the running time as the number of examples and the number of class increase. The proposed approach in this work differs from the method in [10] since prunning is modeled by continuous optimization framework which gives exact number of classifiers in the subensemble unlike by ordered aggregation in [10].

The rest of the paper is organized as follows. In section 2, ECOC and accuracy/diversity trade-off will be reviewed. Section 3 describes our method with the mathematical formulation and the solution technique. Section 4 is devoted to the experiments and concludes with a discussion in Section 5.

2 Error Correcting Output Codes

The Error Correcting Output Codes (ECOC) [11] framework is a general method for multiclass classification by embedding of binary classifiers. Given a set of k classes, ECOC generates a coding matrix M of size $k \times n$ in which each row corresponds to a codeword per class, i.e., i^{th} row refers to the codeword of length n for i^{th} class. Each codeword consists of $\{-1, +1\}$ binary entries. In terms of learning, M is constructed by taking into account n binary problems where each one corresponds to a column of M. Each of these binary problems (or dichotomizers) splits the multiclass problem into two class coded by -1 or $+1$ (or 0 if the class is not considered) in M. Then at the decoding step, applying each trained classifier will give a binary output on the test set which will form a codeword for the test point. The class of the test point is determined by finding the minimal distance between the codeword of the test point and codeword of classes in M. The data point is assigned to the closest codeword in M.There are different decoding strategies in the literature, for a closer look please see [13]. In this paper, we will use Hamming distance[1] as a distance measure between codewords. In Figure 1, an example of ECOC coding/decoding is given.

The ECOC framework is independent of base classifiers and has been shown to reduce bias and variance produced by the learning algorithms as mentioned

[1] Hamming distance between codeword c_1 and c_2 is the number of places where c_1 and c_2 are different

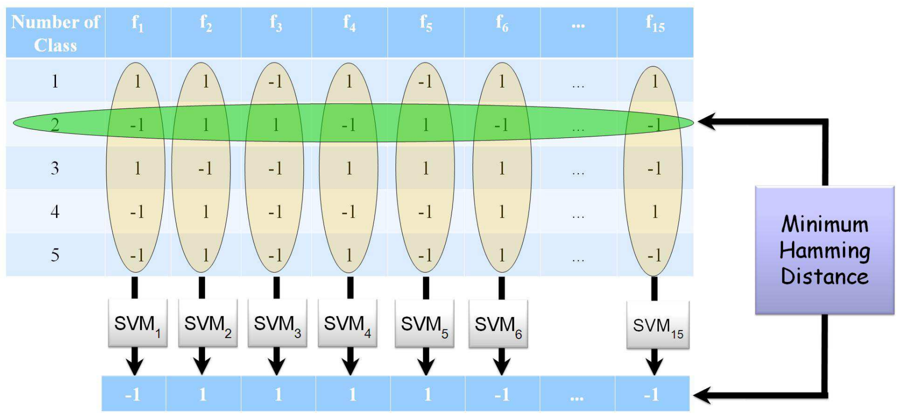

Fig. 1 An example of ECOC framework with Support Vector Machine (SVM) base classifiers.

in [23]. Because of these reasons, ECOC has been widely used for the multiclass classification problems.

2.1 Accuracy Diversity Trade off

Diversity has long been recognised as a necessary condition for improving ensemble performance, since if base classifiers are highly correlated, it is not possible to gain much by combining them. The individual classifiers may be very accurate, and indeed the more accurate they become, the less diversity exists between them [42]. This dilemma has become known as the accuracy/diversity trade-off. There have been many attempts to define diversity [25], but the consensus is that no matter how it is defined, there does not exist any measure that can by itself predict generalisation error of an ensemble. Since there is a lack of a general theory on how diversity impacts ensemble performance, experimental studies continue to be an important contribution to discovering whether a relationship exists and if so whether it can be quantified and understood.

For solving multi-class problems using ECOC, considerations of the accuracy/diversity trade-off are even more complex. The original motivation for ECOC was based on error-correcting theory, which assumes that errors are independent. However, when applied to multi-class machine learning problems, the error correlation depends on the data set, base classifier as well as the code matrix. In the original ECOC approach [11], heuristics were employed to maximise the distance between the columns to reduce error correlation. There have been other attempts to address error correlation. Hadamard matrices maximise distance between rows and columns and were used as ECOC code matrix in [16], in which an upper bound on probability of error correlation was derived as a function of minimum distance between code words. In [1] it was shown that a high minimum distance between any pair implies a reduced upper bound on the generalisation error, and in [17] it was shown for a random matrix that if the code is equi-distant and long enough, then decision-making is bayes-optimal. More recent approaches aim to introduce problem-dependent designs to address error correlation [13].

Based on previous research, it is possible to summarise the main considerations in designing ECOC matrices

– minimum Hamming Distance between rows (error-correcting capability)

– variation of Hamming Distance between rows (effectiveness of decoding)

– number of columns (repetition of different parts of sub-problems)

– Hamming Distance between columns and complement of columns (independence of base classifiers).

All these constraints make optimal design of coding and decoding strategies a complex problem. Previous studies have attempted to address some of these constraints experimentally. An accuracy/diversity study was carried out in [42] using a random code matrix with near equal split of classes (approximately equal number of 1's in each column), as proposed in [34]. It is proved in [43] that optimal performance is attained if codes are equi-distant, and an experimental comparison is made of random, equi-distant and non-equidistant code matrices. However, the proposed approach in this paper is the first to incorporate pruning into ECOC and address the optimisation of the accuracy/diversity trade-off in a principled fashion.

3 Pruning Methods

In this section, we introduce a family of pruning methods based on ordered aggregation. The order of aggregation is determined according to the different measures such as voting, accuracy and diversity in which the initial ensemble starts with the optimal measures and iteratively adds new candidate classifiers based on diversity/accuracy of the ensemble on the training set $Z_{training}$.

As discussed in Section 2.1, a single measure of accuracy or diversity alone is not a sufficient criterion to prune the ensembles. Both accuracy and the diversity must be considered together for selecting the classifiers in the ensemble. Some rules that change the order of aggregation are Reduced Error Pruning Method (REP) [26], Kappa Pruning (KP) [26], Complementarity Measure [27], Margin Distance Minimization (MDSQ) [27], Orientation Ordering [28] and Boosting Based Pruning [29]. In this section, we will give the idea of the first two methods, namely REP and KP and continue with our method developed for ECOC.

3.1 Reduced Error Pruning (REP)

This method is first introduced in [26]. The result of combining the predictions of the classifiers in an ensemble $E_T = \{h_t(x)\}_{t=1}^{T}$ using equally weighted voting is

$$H_{E_T}(x) = \arg\max_{y} \sum_{t=1}^{T} I\left(h_t(x) = y\right),\ y \in Y$$

where h_t is the hypothesis and Y is the set of labels. The first classifier in the ensemble is the one having the minimum classification error. The algorithm searches for the second classifier which makes the classification error of the new ensemble minimum. After it finds the new ensemble, it iteratively adds the rest of the classifiers one by one and incorporates the one which gives the lowest ensemble error for the new ensemble until it reaches the desired ensemble size. The subensemble S_u is constructed by adding to S_{u-1}, the classifier

$$s_u = \arg\max_{k} \sum_{x,y \in Z_{training}} I\left(H_{S_{u-1} \cup h_k}(x) = y\right), k \in E_T \setminus S_{u-1},$$

where k runs over the all classifiers which haven't been selected up to that iteration and $y \in Y$ and I is the indicator function.

In this paper, we adapted REP algorithm to ECOC by the same manner. SVM with Gaussian kernel is used as a base classifier for the ECOC learners. Each base classifier is determined by 5 fold cross validation and REP is applied on 10 different random folds in which each fold has its own ensemble. The size of the subensemble is chosen to be the same size as the subensemble of the pruned ECOC matrix proposed in this study.

3.2 Kappa Pruning (KP)

This method is based on selecting the most diverse classifiers by using κ statistics [25]. As in REP, Kappa Pruning iteratively adds a new classifier to the ensemble which gives the minimum pairwise κ measure. It starts with the pair of classifiers which have the minimum pairwise κ diversity. Then, it adds the classifier which makes the mean of the pairwise diversities minimum in the ensemble. Likewise in REP, here the formula differs only with kappa measure

$$s_u = \arg\max_{k} \kappa_{Z_{training}}\left(h_k, H_{S_{u-1}}\right), k \in E_T \setminus S_{u-1},$$

where κ is the pairwise diversity measure given by $\kappa = \frac{N^{00}}{N^{11}+N^{10}+N^{01}+N^{00}}$ in [25]. Here N^{ab} is the number of elements where $y_i = a$ or $y_j = b$. In this study, KP is adapted to ECOC by using exactly the same logic. As with REP, SVM is used as base classifier for ECOC and CV is applied as in Section 3. The method is tested on 10 different random folds in which each fold has its own ensemble.

3.3 Pruning of ECOC by using Optimization Model

It is known that as the number of classes increases, the number of base classifiers in the ECOC matrix also increase for exhaustive search. Let us remember that if k is the number of classes, the number of columns in ECOC can be at most $\frac{2^k}{2} - 1$. As k increases the number of base classifiers in the ensemble increases exponentially. Hence the running time and the efficiency of the algorithm decreases. In this study we propose a pruning method to select the best accurate and diverse classifiers from exhaustive ECOC coding by incorporating the diversity measure of [47]. In order to get good mathematical formulation of trade off, error structure of the problem is represented by a linear combination of base classifiers' errors and diversities from the error analysis in [47]. It is claimed that if the strength and diversity measurements for a classification ensemble can be found, a linear combination of them should serve as a good approximation of the overall ensemble error. Minimizing the approximate ensemble error function will be the objective of mathematical programming. In [47], the error of each base classifier is reported in the matrix P on the training set as follows:

$P_{i,j} = 0$, if the j^{th} classifier is correct on data point i,

$P_{i,j} = 1$, otherwise.

A matrix G is defined to be the error matrix by $G = P^T P$ since the diagonal term G_{ii} will be total error that classifier i makes, and the off diagonals G_{ij} will be the common errors that classifier i and j make so that off diagonals correspond to the measure for diversity. The matrix G is then normalized to put all the elements in the same scale as

$$\tilde{G}_{ii} = \frac{G_{ii}}{N},$$

where N is the number of training points and

$$\tilde{G}_{ij,i\neq j} = \frac{1}{2}\left(\frac{G_{ij}}{G_{ii}} + \frac{G_{ij}}{G_{jj}}\right). \tag{1}$$

We note that pruning of the ECOC matrix in this study differs from choosing the best columns of ECOC matrix since pruning here is based on both the observed error rates and diversity measures on the training sets. As discussed above, $\tilde{G}_{ii}$ represents the total error made by classifier i and $\tilde{G}_{ij}$ measures the common errors made by pairwise classifiers i and j. Note that the new matrix $\tilde{G}$ is symmetric by taking the average of $\frac{G_{ij}}{G_{ii}}$ and $\frac{G_{ij}}{G_{jj}}$. Furthermore, $\sum_i \tilde{G}_{ii}$ measures the overall strength of the ensemble and $\sum_{ij,i\neq j} \tilde{G}_{ij}$ measures the diversity. The mathematical programming is formulated by quadratic integer problem in [47] where a fixed subset of classifiers is searched as a constraint by the following problem

$$\min_x x^T \tilde{G} x \tag{2}$$

$$\text{subject to } \sum_i x_i = k,$$
$$x_i \in \{0, 1\}.$$

Here, x_i represents whether the i^{th} classifier is in the ensemble or not. If $x_i = 1$, then it means i^{th} classifier is chosen to be in the ensemble, if $x_i = 0$ then it is not considered in the ensemble and k is the length of the ensemble and should be chosen as a parameter beforehand. The problem (2) is NP hard in general but it is approximated as a max-cut problem in [47] and solved by semi-definite programming (SDP). The overall efficiency of the problem depends on the ensemble size k since it is the parameter given before solving the problem and the solution changes accordingly.

3.3.1 Unparametrized Pruning (UP)

In this study, we get rid of the parameter of ensemble size simply by adding penalization term to the objection function with a regularization constant ρ. Note that the constraint in equation (2) can be written as $\|x\|_0 = k$. Here zero norm is defined by the number of non zero elements which leads the sparsity in the model. Instead of determining the pruning rate k in (2), finding the indices of non zero entires of x corresponds to the pruning rate which refers to the number of classifiers in the subensemble. Furthermore, by this way and with the help of sparsity, we introduced the relaxation of the binary vector to the real vector, i.e. $x \in \mathbf{R}^n$. Then the equation (2) becomes an unconstrained problem which is the regularized version of the problem (2).

$$\min_{x \in R^n} x^T \tilde{G} x + \rho \|x\|_0 \tag{3}$$

The first step to solve the continuous optimization problem (3) is to approximate the cardinality constraint, i.e. $\|x\|_0$. One can approximate it by $\|x\|_1$ which is the usual heuristic. We approximated it as the negative log-likelihood of a Student t-distribution, which is a tighter approximation than $\|x\|_1$ and has been used in many different contexts [8,14,38,41]. There are other approximations to $\|x\|_0$, e.g., $\sum_{i=1}^n (1 - e^{-\alpha |x_i|})$ where $\alpha > 0$ [4].

If we define the approximation of the zero norm as

$$\|x\|_0 := \sum_{i=1}^n 1_{x_i \neq 0} = \lim_{\epsilon \to 0} \sum_{i=1}^n \frac{\log(1 + |x_i|/\epsilon)}{\log(1 + 1/\epsilon)},$$

then the problem (3) becomes

$$\min_{x \in R^n} x^T \tilde{G} x + \rho \lim_{\epsilon \to 0} \sum_{i=1}^n \frac{\log(1 + |x_i|/\epsilon)}{\log(1 + 1/\epsilon)} \tag{4}$$

Let us denote the set of symmetric matrices by $\mathbf{S}^n$ and denote positive semidefinite and positive definite matrix by $\mathbf{S}^+$ and $\mathbf{S}^{++}$ respectively. Observe that the matrix $\tilde{G}$ is a symmetric matrix since $\tilde{G}_{ij} = \tilde{G}_{ji}$, and the indices ij and ji refer to the common errors between i^{th} and j^{th} classifier. Note that the problem (4) is a non convex unconstrained problem since the matrix $\tilde{G} \notin \mathbf{S}^{+}$or $\mathbf{R}^{++}$. Then we can not use the well known property of convex optimization (see Theorem below [31]) which states the following

Theorem 1 *[31] Let x^* be the feasible point of the problem*

$$\min_x f(x), \tag{5}$$

where $f(x)$ is convex differentiable function. If $\nabla(f(x^*)) = 0$ then x^* is the global minimum of (5).

We will model the problem (4) by difference of convex (DC) functions since the objective function has the structure of DC. Note that the recent formulation (4) is a continuous optimization problem unlike the problem (2) which has a combinatorial term. Before giving DC formulation of the problem, lets define DC program as below:

Definition 1 Let Ω be the convex set in R^n and $f : \Omega \to R$ be a real valued function. Then, f is a DC function on Ω if there exist two *convex* functions $g, h : \Omega \to R$ such that $f(x) = g(x) - h(x), x \in \Omega$. Optimization problems of the form

$$\min_{x\in\Omega} f_0(x) \tag{6}$$

$$\text{s.t.} f_i(x) \le 0,\ i = 1, \ldots, m \tag{7}$$

where $f_i(x) = g_i(x) - h_i(x),\ i = 0, 1, \ldots, m$ is called DC programming.

In order to formulate (4) as DC program, let us choose $\tau \in R$ such that $\tilde{G}+\tau I \in S^+$. If $\tilde{G} \in S^+$, such τ exists trivially, choose $\tau > 0$. If $\tilde{G}$ is indefinite, choosing $\tau > -\lambda_{min}(\tilde{G})$ where λ_{min} is the smallest eigenvalue of $\tilde{G}$ ensures that $\tilde{G} + \tau I \in S^+$. The similar approximation is performed for different concepts such as solving generalized eigenvalue problem in [38]. Therefore, if we choose $\tau > \max(0, -\lambda_{min})$, then we will have positive semi definite matrix for any $\tilde{G} \in S^n$. Then the problem (4) can be written as

$$\min_x x^T \left(\tilde{G} + \tau I\right) x - \tau \|x\|_2^2 + \rho \lim_{\epsilon\to 0} \sum_{i=1}^{n} \frac{\log\left(1 + |x_i|/\epsilon\right)}{\log\left(1 + 1/\epsilon\right)},$$

where $\|.\|_2$ is referred to Euclidean norm. The above problem can be approximated further by neglecting the term lim and choosing $\epsilon > 0$. Hence the following convex unconstrained problem is obtained

$$\min_x x^T \left(\tilde{G} + \tau I\right) x - \tau \|x\|_2^2 + \rho \sum_{i=1}^{n} \frac{\log\left(1 + |x_i|/\epsilon\right)}{\log\left(1 + 1/\epsilon\right)}. \tag{8}$$

With this new formulation, the first model introduced by [47] is made independent of the size of the subensemble k by penalizing the constraint in model (2) with a regularization constant τ. The size of subensemble k changes the accuracy of the model since the subensemble having small k can lack important classifiers or having too large k can include redundant classifiers. Testing for all k and choosing the best k by exhaustive search on the training set will make the algorithm too slow, especially as the number of classes increases. In the new formulation proposed in model (8), new parameters τ and ρ are introduced where the first one is approximated by the minimum eigenvalue of the matrix $\tilde{G}$ and the latter one is by well known statistical method cross validation. Here we choose the number folds as 5. The algorithm of the proposed method in this study is given by Algorithm 1 and is referred to "ECOC with UP".

Algorithm 1 ***ECOC with UP***

```
Input: a $k$ class problem
Base classifier
ECOC matrix $M_{k\times n}$
Output: error rates of Test Set
1: Partition data $X$ into Training Set $X_{tr}$ and Test Set $X_{test}$
2: Run base classifier for each column of $M_{k\times n}$ on $X_{tr}$
3: Compute matrix $\tilde{G}$ defined by equation (1) on $X_{tr}$
4: Compute a solution $x$ of
       $$\min_x x^T \left(\tilde{G} + \tau I\right) x - \tau \|x\|_2^2 + \rho \sum_{i=1}^{n} \frac{\log(1 + |x_i|/\epsilon)}{\log(1 + 1/\epsilon)}. \quad (9)$$
5: Find all $i's$ such that $x_i \geq 0.5$, $i = 1 \dots m$ to be the indices of new classifiers in subset
6: Construct new ECOC matrix $\tilde{M}_{k\times m}$ by choosing columns from indices in Step 5.
7: Run ECOC framework with $\tilde{M}$ on test set $X_{test}$
```

4 Experiments

We implemented our novel pruning method by MATLAB **fminunc** function from optimisation toolbox which solves the unconstrained optimization problem [5]. The new parameters τ and ρ introduced in equation (8) are calculated by $\tau = \lambda_{min}(\tilde{G})$ and by 5 fold cross validation respectively. First we derived the matrix $\tilde{G}$ from the standard ECOC algorithm on the training set and then performed pruning by solving the optimization problem (8). It has been theoretically and experimentally proven that the randomly generated long or equi-distant code matrices give close to optimum performance when used with strong base classifiers [18,43]. Thus, in this study coding of the ECOC matrix is performed by random generation such that each column of ECOC matrix has approximately equal number of -1s and +1s. The solution set of the problem (8) constitutes the reduced number of base learners for ECOC. We used

UCI machine learning repository data sets of ecoli, glass, dermatology, yeast, wine and facial expression classification data [2]. As a base learner we used SVM with Gaussian kernels. Kernel parameters and regularisation constant of SVM are chosen by 10 fold cross validation. In [47], pruning size is set beforehand but in our formulation, it is part of the optimisation (8). Since the solution of the problem (8) consists of continuous values, we approximated the solution to binary values by well known heuristic *rounding* in combinatorial optimization. From the experiments we observe that the solution converges if a threshold is placed on the real output x of (8). The threshold is taken as $x > 0.5$ as indicated in Step 5 in Algorithm 1 where the i^{th} classifier is included to the subensemble if $x_i > 0.5$.

In Table 1, 2 and 3, error rates and running time in seconds (given in parenthesis) are compared with different pruning methods REP and KP, ACEC in [10], SDP in [47] that are adapted to ECOC and further random guessing results are reported. For statistical significance, student t-test is performed to asses whether the error rate is in the confidence interval on 10 test folds and it is found that the error rates reported in Table 1, 2 and 3 are in confidence interval which are referred to CI. The statistical significance is found to be $p > 0.99$ with a confidence level of 95% and $H = 0$ which indicates that we should reject the null hypothesis. Our null hypthesis is "Error rates are random for 10 test folds". The subensemle size k is determined from our pruning method Algorithm 1 by Step 5 where k refers to the number of non zero elements of vector x in equation (4) and it is fixed for the other pruning methods in order to make a meaningful comparison. The method "SDP" on which our algorithm is based on is parametric on the subensemble size k. The error rate highly depends on k. If k is too small it may give high error rate and if k is too big it may contain redundant classifiers in the subensemble which also affects error rate. So it is important to determine the best k. Finding k heuristically will be time consuming since it is necessary to try all k values, i.e., $k = 1, 2, \ldots n$. Thus comparison of error rates of "ECOC with UP" with the method SDP is not strictly fair since k in SDP is found in "ECOC with UP" beforehand. Likewise, comparison with ACEC is not strictly fair since it is an ordered aggregated method and cpu time of ACEC is higher than all methods compared in this study. Thus for large data sets, ACEC is not an efficient method although it gives higher accuracy.

On the other hand, k is replaced with a new regularization parameter ρ introduced within penalization term of our approach "ECOC with UP" is determined by cross validation for values $\rho = [1 \quad 10 \quad 100 \quad 500 \quad 1000]$ on training set. Furthermore, it is important to observe that running time for cross validation to determine ρ is less than carrying out heuristically to find k.

We tested our algorithm on different size of pool of classifiers, i.e., different size of ECOC matrices, such as 50, 100, 150 base classifiers on 5 data sets from UCI machine learning repository [2] and Facial Expression Classification

[2] Cohn Kanade Database, http://vasc.ri.cmu.edu/idb/html/face/facial_expression/

(FAC) data [19] which will be explained in next Section 4.1 . These results show that ACEC performs better in terms of error rate but it is very slow in running time since it is based on ordered aggregation. If we compare our approach "ECOC with UP" with other methods except ACEC, it performs mostly better than other pruning methods. Note that, SDP and ACEC are not applicable for large scale data as it can be seen on facial expression data "FAC" on each table. Observe that when any other pruning method except "ECOC with UP" gives better results, it has always very slow running time. For instance, In Table 1 for FAC data, Full ECOC and Random Guessing give better error rate but the running time is greater than with "ECOC with UP". KP is still better than our method but the running time increases significantly. In Table 3, for the wine data, even though REP is significantly better than all, the running time is twice as long as "ECOC with UP" method which has the second best error rate. Likewise, for the glass data in Table 3, "ECOC with UP" has the second best result with a lower running time than Full ECOC, REP and KP. Especially, REP and KP can be very time consuming because of the combinatorial structure of the algorithm, even though they give better results in some cases, e.g., Table 3. As explained in Section 3.1 and Section 3.2, both of the algorithms go through all combinations to find the minimum ensemble error which makes these algorithms very slow. It should be also noted that pruning size must be determined as an input variable for all methods that we compared in this section.

4.1 Facial Expression Classification

Automatic facial expression recognition has applications in areas such as human-computer interaction, human emotion analysis, biometric authentication and fatigue detection. However, the emotions characterised in these different applications can be quite varied. While it is possible to learn emotions directly from features derived from raw images, an attractive and more generic approach is to decompose the face into discrete facial features. A commonly used method is the facial-action coding system (FACS) [12,39], in which facial features are characterised as one of 44 types known as action units (AUs). The advantage of FACS is that facial expressions of interest (e.g. emotions or fatigue) can be learned by looking for particular groups of AUs so that the *interpretation* can be de-coupled from their *detection*. Fig. 2 shows example AUs from the eye region.

It is possible to define a series of two-class problems, in which a classifier is trained to differentiate each AU from all other AUs (one versus rest). However, the presence of one AU may affect the strengths of other AUs, in other words not all AUs are linearly independent. In this paper, AU detection is posed as a single multiclass problem, in which groups of AUs are assigned a single class. Therefore a single AU may appear in more than one group.

In order to detect AUs, the images first need to be pre-processed. Multi-resolution local binary patterns (MLBP) are used for feature extraction [37,

Data	k	ACEC	SDP	**ECOC with UP**	Full ECOC	REP	Random Guessing	KP
ecoli	28	**0.1613**±0.0923(1820*s*)	0.1876±0.0780(169s)	0.1841±0.0718(250s)	0.2019 ±0.0633(692s)	0.2084±0.0838 (1099s)	0.2317±0.0662(403s)	0.1928±0.0723 (902s)
		CI =[0.0952 0.2273]	CI=[0.0966,0.2786]	CI=[0.1004,0.2679]	CI=[0.1280,0.2757]	CI=[0.1106,0.3062]	CI=[0.1544,0.3089]	CI=[0.1083,0.2772]
glass	22	**0.2503**±0.1027(936s)	0.3829±0.1231(79s)	0.3448±0.0983 (224s)	0.4720±0.1604(204s)	0.3432±0.1209(626s)	0.3781±0.1281(89s)	0.4592±0.1615(511s)
		CI=[0.1768 0.3238]	CI=[0.2392,0.5265]	CI=[0.2300,0.4595]	CI=[0.2848,0.6592]	CI=[0.2021,0.4843]	CI=[0.2286,0.5275]	CI=[0.2707,0.6477]
derm	37	0.0275±0.0259(4075s)	0.0332±0.0254(70s)	0.0338±0.0326(591s)	0.0247±0.0195(608s)	0.0280±0.0270(1778s)	**0.0247**±0.0195(443s)	0.0247±0.0273(1897s)
		CI=[0.0090 0.0460]	CI = [0.0036 0.0629]	CI=[-0.0042,0.0718]	CI=[0.0019, 0.0474]	CI=[-0.0034,0.0595]	CI=[0.0019,0.0474]	CI=[-0.0072,0.0565]
yeast	28	**0.4657**±0.0425(33961s)	0.5197±0.0374(3708s)	0.4853±0.0574(3671s)	0.5062±0.0520(7840s)	0.5162±0.1035(18560s)	0.5136±0.0474(4013s)	0.5082±0.0631(19665s)
		CI= [0.4353 0.4961]	CI=[0.4760,0.5634]	CI=[0.4183,0.5523]	CI=[0.4455,0.5670]	CI=[0.3955,0.6370]	CI=[0.4583,0.5690]	CI=[0.4345,0.5818
wine	32	0.0569±0.0558(737s)	0.0628±0.0590(81s)	0.0609±0.0485(302s)	0.0668±0.0436(140s)	**0.0374**±0.0430(341s)	0.0668±0.0436(88s)	0.0668±0.0436(455s)
		CI=[0.0170 0.0968]	CI=[-0.0060,0.1317]	CI=[0.0044,0.1175]	CI=[0.0159,0.1177]	CI=[-0.0127,0.0875]	CI=[0.0159, 0.1177]	CI=[0.0159, 0.1177]
FAC	40	NA	0.4538±0.0616(9367s)	0.4538±0.0530(6099s)	**0.4250**±0.0466(14164s)	0.5038±0.0652(8070s)	**0.4250**±0.0457(11044s)	0.4346±0.0464(18299s)
		NA	CI=[0.3819, 0.5257]	CI=[0.3920, 0.5157]	CI = [0.3706,0.4794]	CI=[0.4277, 0.5800]	CI=[0.3716,0.4784]	CI=[0.3805, 0.4887]

Table 1 Mean error values and running times of 10 fold cross validation with 50 base classifiers, here k is the ensemble size after pruning.

Data	k	ACEC	SDP	**ECOC with UP**	Full ECOC	REP	Random Guessing	KP
ecoli	75	**0.1492**±0.0811(3584s)	0.2809±0.0832(150s)	0.1517±0.0335(217s)	0.1867±0.0758(724s)	0.1765±0.0761(1262s)	0.1837±0.0711(523s)	0.1893±0.0889(2082s)
		CI=[0.0911 0.2072]	CI=[0.2213 0.3404]	CI=[0.1101,0.1934]	CI=[0.1325,0.2409]	CI=[0.1220,0.2309]	CI=[0.1328,0.2345]	CI=[0.1257, 2528]
glass	27	**0.2455**±0.0859(1627s)	0.4720±0.1604(70s)	0.3558±0.1015(389s)	0.4720±0.1604(371s)	0.3663±0.1305(1361s)	0.3686±0.1314(106s)	0.4257±0.1458(1118s)
		CI =[0.1841 0.3070]	CI=[0.3572,0.5868]	CI=[0.2832,0.4284]	CI=[0.3572,0.5868]	CI=[0.2730,0.4596]	CI=[0.2746,0.4625]	CI=[0.3215,0.5300]
derm	23	**0.0275**±0.0259(6094s)	0.0275±0.0259(173s)	**0.0275**±0.0259(1391s)	0.0304±0.0275(1107s)	0.1157±0.0675(1118s)	**0.0275**±0.0259(125s)	**0.0275**±0.0259(3058s)
		CI=[0.0090 0.0460]	CI=[0.0090,0.0460]	CI=[0.0090,0.0460]	CI=[0.0107,0.0501]	CI=[0.0674,0.1640]	CI=[0.0090,0.0460]	CI=[0.0090,0.0460]
yeast	82	**0.4306**±0.0558(75645s)	0.4799±0.0493(12079s)	0.4730±0.0424(3601s)	0.4853±0.0515(8200s)	0.4961±0.0507(15462s)	0.4772±0.0531(1123s)	0.4745±0.0535(4310s)
		CI=[0.3907 0.4705]	CI=[0.4447,0.5152]	CI=[0.4204,0.5256]	CI=[0.4484,0.5221]	CI=[0.4598,0.5323]	CI=[0.4392,0.5152]	CI=[0.4362,0.5128]
wine	47	0.0511±0.0435(1307s)	0.0628±0.0441(116s)	0.0609±0.0485(431s)	0.0628±0.0441(290s)	**0.0433**±0.0413(1012s)	0.0628±0.0441(217s)	0.0746±0.0690(397s)
		CI=[0.0199 0.0822]	CI=[0.0313,0.0943]	CI=[0.0262,0.0956]	CI=[0.0313,0.0943]	CI=[0.0138,0.0728]	CI=[0.0313,0.0943]	CI=[0.0252,0.1239]
FACE	86	NA	0.4231±0.0435(19129s)	0.4596±0.0540(13752s)	0.4462±0.0413(27389s)	0.4558±0.0552(17828s)	0.4365±0.0544(24139s)	**0.4308**±0.0427(37900s)
		NA	CI=[0.3919,0.4542]	CI=[0.4210,0.4982]	CI=[0.4166,0.4757]	CI=[0.4163,0.4953]	CI=[0.3976,0.4755]	CI=[0.4002,0.4613]

Table 2 Mean error values and running times of 10 fold cross validation with 100 base classifiers, here k is the ensemble size after pruning.

Data	k	ACEC	SDP	**ECOC with UP**	Full ECOC	REP	Random Guessing	KP
ecoli	88	**0.1487**±0.0920(6397s)	NA	0.1751±0.0663(1431s)	0.1893±0.0842(2465s)	0.1902±0.0794(6373s)	0.1893±0.0725(1558s)	0.2044±0.0739(5686s)
		CI=[0.0829 0.2145]	NA	CI=[0.1276,0.2225]	CI=[0.1290,0.2495]	CI=[0.1334,0.2470]	CI=[0.1374,0.2411]	CI=[0.1516,0.2573]
glass	62	**0.2408**±0.0817(2721s)	0.4720±0.1604(150s)	0.3876±0.1547(1132s)	0.4720±0.1604(589s)	0.3495±0.1278(2809s)	0.4720±0.1604(**246s**)	0.4625±0.1501(2157s)
		CI=[0.1823 0.2992]	CI=[0.3572,0.5868]	CI=[0.2770,0.4983]	CI=[0.3572,0.5868]	CI=[0.2581,0.4409]	CI=[0.3572,0.5868]	CI=[0.3551,0.5698]
derm	119	**0.0275**±0.0259(14293s)	**0.0275**±0.0292(563s)	0.0309±0.0212(3704s)	0.0304±0.0275(2056s)	0.0280±0.0302(8264s)	0.0304±0.0275(1596s)	0.0304±0.0275(1077s)
		CI=[0.0090 0.0460]	CI=[0.0066,0.0484]	CI=[0.0157,0.0461]	CI=[0.0107,0.0501]	CI=[0.0065,0.0496]	CI=[0.0107,0.0501]	CI=[0.0107, 0.0501]
yeast	77	**0.4340**±0.0573(93663s)	0.4779±0.0495(9371s)	0.4825±0.0423(9011s)	0.4745±0.0448(20945s)	0.4731±0.0565(68825s)	0.4813±0.0457(11267s)	0.4738±0.0496(60365s)
		CI=[0.3931 0.4750]	CI=[0.4424,0.5133]	CI=[0.4523,0.5128]	CI=[0.4424,0.5066]	CI=[0.4327,0.5135]	CI=[0.4486,0.5139]	CI=[0.4383,0.5093]
wine	97	0.0569±0.0558(2246s)	NA	0.0551±0.0442(1326s)	0.0668±0.0436(570s)	**0.0374**±0.0430(2471s)	0.0609±0.0398(411s)	0.0628±0.0441(7942s)
		CI=[0.0170 0.0968]	NA	CI=[0.0234,0.0867]	CI=[0.0356,0.0980]	CI=[0.0067,0.0681]	CI=[0.0325,0.0894]	CI=[0.0313,0.0943]
FAC	110	NA	NA	0.4577±0.0487(20039s)	0.4462±0.0460(42134s)	0.4885±0.0596(29218s)	0.4500±0.0490(30758s)	**0.4404**±0.0448(58412s)
		NA	NA	CI=[0.4229,0.4925]	CI=[0.4132,0.4791]	CI=[0.4458,0.5311]	CI=[0.4149,0.4850]	CI=[0.4083,0.4725]

Table 3 Mean error values and running times of 10 fold cross validation with 150 base classifiers, here k is the ensemble size after pruning.

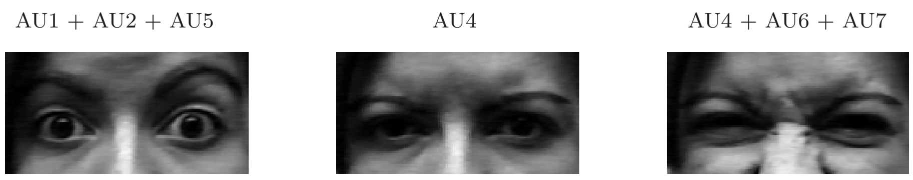

Fig. 2 Some example AUs and AU groups from the region around the eyes. AU1 = inner brow raised, AU2 = outer brow raised, AU4 = brows lowered and drawn together, AU5 = upper eyelids raised, AU6 = cheeks raised, AU7 = lower eyelids raised. The images are shown after manual eye location, cropping, scaling and histogram equalisation.

Class number	1	2	3	4	5	6	7	8	9	10	11	12
AUs present	None	1,2	1,2,5	4	6	1,4	1,4,7	4,7	4,6,7	6,7	1	1,2,4
No. examples	152	23	62	26	66	20	11	48	22	13	7	6

Table 4 Classes of action unit groups used in the experiments.

32], and the fast correlation-based filter (FCBF) algorithm [46] is employed for feature selection. Further details of the pre-processing and normalisation procedures may be found in [36].

Results are presented for the Cohn-Kanade face expression database [19], which contains frontal video clips of posed expression sequences from 97 university students. The last image has available ground truth in the form of a manual AU coding by human experts. We focused on detecting AUs from the upper face region as shown in Fig. 2. In order to avoid defining classes with very few training patterns, AU groups with three or fewer examples were ignored. This led to 456 images available for training and these were distributed across the 12 classes shown in Table 4.

We applied the same procedure in ECOC and in pruning, as described in Section 3. ECOC is performed with 200 base classifiers, for which we used SVM with Gaussian kernel [9]. Each run was based on a different randomly chosen stratified training set with a 90/10 training/test split. The ECOC code matrices were randomly generated with balanced numbers of 1s and -1s in each column, as proposed by [33]. Experimental results of FAC data are compared with the proposed pruning algorithm in Tables 1,2 and 3, They show that appropriate subset of base learners gives approximately same error rate, so that fewer base learners leads to less training time, which is proportional to number of base learners.

5 Discussion and Conclusion

In this study, we proposed a faster and more efficient pruning method for ECOC by optimizing the accuracy and diversity simultaneously in the proposed cost function. Our new algorithm prunes the set of base classifiers by solving a continuous optimization unlike in [47]. One of the important aspects of the proposed method here is that the size of the pruned set comes directly from the optimization problem. The unconstrained optimization formulation given in equation (4) does not need the ensemble size and finds the optimum subensemble on the training set. In [47] and [10], the pre-defined pruning size determines the error rate, while here it is determined by searching optimality conditions. For higher number of classes and higher number of base classifiers, the pruning will lead to a more efficient solution for multiclass classification. Different size of ECOC matrices are tested for 5 different pruning method and for full ECOC matrix without pruning. For most of the data ECOC with UP reduces the error in a smaller running time as shown in Table 1, 2 and 3. It should be clarified that as decoding, we used Hamming distance and as a future work, we will apply other decoding methods proposed in Section 2.2.

References

1. Allwein, E.L., Schapire, R.E., Singer, Y.: Reducing multiclass to binary: a unifying approach for margin classifiers (2000)
2. Blake, C., Merz, C.: UCI repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences (1998). Http://www.ics.uci.edu/∼mlearn/MLRepository.html
3. Bouzas, D., Arvanitopoulos, N., Tefas, A.: Optimizing subclass discriminant error correcting output codes using particle swarm optimization. In: 21st International Conference on Artificial Neural Networks, ICAAN 2011,Lecture Notes in Computer Science, pp. 79–86. Springer-Verlag, Berlin, Heidelberg (2011)
4. Bradley, P.S., Mangasarian, O.L.: Feature selection via concave minimization and support vector machines. In: 15th international conf. on machine learning, San Francisco: Kaufmann, pp. 82–90 (1998)
5. Branch, M.A., Grace, A.: Optimization Toolbox User's Guide, Version 2. 24 Prime Park Way, Natick, MA 01760-1500 (2002)
6. Breiman, L.: Bagging predictors. Machine Learning **24**(2), 123–140 (1996)
7. Breiman, L.: Arcing classifiers. The Annals of Statistics **26**, 801–849 (1998)
8. Candes, E.J., Wakin, M., Boyd, S.: Enhancing sparsity by reweighted l_1 minimization. The Journal of Fourier Analysis and Applications **14**, 877–905 (2007)
9. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2**, 27:1–27:27 (2011). Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm
10. C.Zor, Windeatt, T., Kittler, J.: Ecoc matrix pruning using accuracy information. In: Multiple Classifier Systems, 11th International Workshop, MCS 2013, Nanjing, China, May 15-17, 2013. Proceedings, vol. 7872, pp. 386–397. Springer Berlin Heidelberg (2014)
11. Dietterich, T., Bakiri, G.: Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research **2**, 263–282 (1995)
12. Ekman, P., Friesen, W.: The Facial Action Coding System: A Technique for The Measurement of Facial Movement. Consulting Psychologists Press, San Francisco (1978)
13. Escalera, S., Pujol, O., Radeva, P.: On the decoding process in ternary error-correcting output codes. IEEE Transactions in Pattern Analysis and Machine Intelligence **32**, 120–134 (2010)
14. Fazel, M., Hindi, H., Boyd, S.: Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In: American control conference, Denver, Colorad (2003)
15. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 148–156 (1996). URL citeseer.nj.nec.com/freund96experiments.html

16. Guruswami, V., Sahai, A.: Multi-class learning, boosting, and error-correcting codes. In: Proc twelth Conf on Computational Learning Theory (1999)
17. James, G.M.: Majority vote classifiers: Theory and applications. Ph.D. thesis, Dept. of Statistics, Univ. of Stanford, Calif. (1998)
18. James, G.M., Hastie, T.: Error coding and PICT's. ASA,Student paper competitions for statistical students (1997). Http://www-stat.stanford.edu/ gareth/
19. Kanade, T., Cohn, J., Tian, Y.: Comprehensive database for facial expression analysis. In: Proc. 4th Int. Conf. Automatic Face and Gesture Recognition, pp. 46–53 (2000)
20. Kim, Y., Street, N.W., Menczer, F.: Meta-evolutionary ensembles. In: In IEEE International Joint Conference on Neural Networks, p. 27912796 (2002)
21. Kleinberg, E.M.: An overtraining-resistant stochastic modeling method for pattern recognition. Annals of Statistics **4**, 2319–2349 (1996)
22. Kleinberg, E.M.: A mathematically rigorous foundation for supervised learning. In: J. Kittler and F. Roli, editors, Multiple Classifier Systems. First International Workshop, MCS 2000, Cagliari, Italy, vol. 1857 of Lecture Notes in Computer Science, pp. 67–76. Springer-Verlag (2000)
23. Kong, E., Dietterich, T.: Error - correcting output coding correct bias and variance. In: In The XII International Conference on Machine Learning, pp. 313–321. San Francisco, CA. Morgan Kauffman (1995)
24. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience, New York (2004)
25. Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles. Machine Learning **51**, 181–207 (2003)
26. Margineantu, D., Dietterich, T.: Pruning adaptive boosting. In: 14th Intl Conf. Machine Learning, pp. 211–218 (1997)
27. Martinez-Mungoz, G., Suarez, A.: Aggregation ordering in bagging. In: Proc. IASTED Intl Conf. Artificial Intelligence and Applications, pp. 258–263 (2004)
28. Martinez-Mungoz, G., Suarez, A.: Pruning in ordered bagging ensembles. In: 23rd Intl Conf. Machine Learning, pp. 609–6163 (2006)
29. Martinez-Mungoz, G., Suarez, A.: Using boosting to prune bagging ensemble. Pattern Recognition Letters **28**, 156–165 (2007)
30. Mason, L., Bartlett, P., Baxter, J.: Improved generalization through explicit optimization of margins. Machine Learning **38**, 243–255 (2000)
31. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer Verlag (2000)
32. Raja, Y., Gong, S.: Sparse multiscale local binary patterns. In: British Machine Vision Conference, vol. 2, pp. 799–808. Edinburgh (2006)
33. Schapire, R.: Using output codes to boost multi-class learning problems. In: Proc. 14th Int. Conf. Machine Learning, pp. 313–321 (1997)
34. Schapire, R.: Using output codes to boost multiclass learning problems. In: 14th International Conf. on Machine Learning, pp. 313–321. Morgan Kaufman (1997)
35. Schapire, R.E.: A brief introduction to boosting. In: 16th International Joint Conference on Artificial Intelligence, In Thomas Dean, editor, pp. 1401–1406. Morgan Kauffman (1999)
36. Smith, R., Windeatt, T.: Facial action unit recognition using filtered local binary pattern features with bootstrapped and weighted ecoc classifiersstudies in computational intelligence. Computational Intelligence **373**, 1–20 (2011)
37. Smith, R.S., Windeatt, T.: Facial expression detection using filtered local binary pattern features with ecoc classifiers and platt scaling. Journal of Machine Learning Research - Proceedings Track **11**, 111–118 (2010)
38. Sriperumbudur, B.K., Torres, D.A., Lanckriet, G.R.G.: A majorization-minimization approach to the sparse generalized eigenvalue problem. Machine Learning (2011). DOI 10.1007/s10994-010-5226-3
39. Tian, Y.I., Kanade, T., Cohn, J.: Recognizing action units for facial expression analysis. IEEE Trans. on PAMI **23**, 97–115 (2001)
40. Vapnik, V.N.: Statistical Learning Theory. Wiley, New York (1998)
41. Weston, J., Elisseeff, A., Schoelkopf, B., Tipping, M.: Use of the zero-norm with linear models and kernel methods. Journal of Machine Learning Research **3**, 1439 – 1461 (2003)

42. Windeatt, T.: Accuracy/ diversity and ensemble classifier design. IEEE Trans Neural Networks **17**, 1194– 1211 (2006)
43. Windeatt, T., Ghaderi, R.: Coding and decoding strategies for multi-class learning problems. Information Fusion **4**, 11–24 (2003)
44. Yin, X., Huang, K., Hao, H., Iqbal, K., Wang, Z.: A novel classifier ensemble method with sparsity and diversity. Neurocomputing **134**, 214–221 (2014)
45. Yin, X., Huang, K., Yang, C., Hao, H.: Convex ensemble learning with sparsity and diversity. Information Fusion **20**, 49–59 (2014)
46. Yu, L., Liu, H.: Efficient feature selection via analysis of relevance and redundancy. Journal of Machine Learning Research **5**, 1205–1224 (2004)
47. Zhang, Y., Burer, S., Street, W.N.: Ensemble pruning via semi-definite programming. Journal of Machine Learning Research **7**, 1315 – 1338 (2006)