Weighted Decoding ECOC for Facial Action Unit Classification

Terry Windeatt

Abstract .

There are two approaches to automating the task of facial expression recognition, the first concentrating on what meaning is conveyed by facial expression and the second on categorising deformation and motion into visual classes. The latter approach has the advantage that the interpretation of facial expression is decoupled from individual actions as in FACS (Facial Action Coding System). In this chapter, upper face action units (*aus*) are classified using an ensemble of MLP base classifiers with feature ranking based on PCA components. When posed as a multi-class problem using Error-Correcting-Output-Coding (ECOC), experimental results on Cohn-Kanade database demonstrate that error rates comparable to two-class problems (one-versus-rest) may be obtained. The ECOC coding and decoding strategies are discussed in detail, and a novel weighted decoding approach is shown to outperform conventional ECOC decoding. Furthermore, base classifiers are tuned using the ensemble Out-of-Bootstrap estimate, for which purpose, ECOC decoding is modified. The error rates obtained for six upper face *aus* around the eyes are believed to be among the best for this database.

Key words: MCS, ECOC, FACS

1 Introduction

The topic of this chapter concerns solving a supervised learning problem in face expression recognition recognition using a combination of neural network classifiers. In the case of face recognition, pattern features consist of real numbers representing different aspects of facial features, as described in

CVSSP, University of Surrey, University, Guildford, Surrey, UK GU2 7XH
t.windeatt@surrey.ac.uk

Section 4. In order to design the learning system we follow the well established technique of dividing the example patterns into two sets, a training set to design the classifier and a test set, which is subsequently used to predict the performance when previously unseen examples are applied.

Multiple Classifier Systems have become an established method for improving generalisation performance over a single classifier, and the relevant aspects are discussed in Section 2. The single classifier performance can be quite sensitive to classifier parameters, and it has previously been shown [17] that an ensemble is less sensitve to base classifier complexity. However, even though an ensemble is less likely to over-fit, there is still the difficulty of tuning individual classifier parameters with respect to ensemble performance. Multi-layer perceptrons (MLP) make powerful classifiers that may provide superior performance compared with other classifiers, but are often criticized for the number of free parameters. The common approach to adjusting parameters is to further divide the training set into two to produce a validation set. When the number of examples is in short supply, cross-fold validation may be used. For example, in n-fold cross-validation, the set is randomly split into n equal parts with (n-1) parts used for training and one part used as a validation set to tune parameters. Training is repeated n times with a different partition each time, and the results averaged. However, it is known that these approaches to validation are either inappropriate or very time-consuming. Ideally all the training set should be used for training, so that there is no need for validation. However, this requires that over-fitting be detected by looking at performance on only the training set, which is a difficult problem. In this chapter the OOB estimate (Section 2), is used to determine optimal parameters from the training set.

The problem of face expression recognition is difficult because facial expression depends on age, ethnicity, gender, occlusions as well as pose and lighting variation [6]. Facial action unit (au) classification is an approach to face expression recognition that decouples the recognition of expression from individual actions. In FACS (facial action coding system) [1] the problem is decomposed into forty-four facial action units, that includes six upper face *aus* around the eyes. This approach has the potential of being applied to a much richer set of applications than an approach that targets facial expression directly. However, the coding process requires skilled practitioners and is time-consuming so that typically there are a limited number of training patterns.

There are various approaches to determining features for discriminating between *aus*. Originally, features were based on geometric measurements of the face that were involved in the au of interest [1]. For example, features were extracted based upon whether the eyes were open or closed, the degree of eye opening, and the location and radius of the iris. More recently, holistic approaches based on PCA, Gabor [2] and Haar wavelets represent a more general approach to extracting features [3], and have been shown to give comparable results. The difficulty with these latter approaches is the large

number of features. When combined with the limited number of patterns, this can lead to the small sample-size problem, that is when the number of patterns is less than or comparable to the number of features. A method of eliminating irrelevant features is therefore required [4] [5]. In this chapter the Out-of-Bag error estimate is used to optimise the number of features.

In previous work [6] [9] five feature ranking schemes were compared using Gabor features in an MLP ensemble. The schemes were Recursive Feature Elimination (RFE) [11] (Section 4) combined with MLP weights and noisy bootstrap, boosting (single feature selected each round), one-dimensional class-separability measure and Sequential Floating Forward Search (SFFS). A full description of these feature selection techniques may be found in [6]. MLP weights combined with RFE, Section 5, perform well for feature selection, even though it is known that MLP weights are not good at selecting most relevant features [7]. It was shown that ensemble performance is relatively insensitive to the feature-ranking method with simple one-dimensional performing at least as well as multi-dimensional schemes. This was a somewhat surprising conclusion, since it is known that sophisticated multi-dimensional schemes out-perform one-dimensional schemes for single classifiers [11]. It was also shown that the ensemble using PCA features with its own inherent ranking outperformed Gabor.

Error-Correcting Output Coding (ECOC) is a well-established method [12] [13] for solving multi-class problems by decomposition into complementary two-class problems, and is fully discussed in Section 3. However, the idea behind ECOC is quite simple and so we introduce the main concept here. ECOC is a two-stage process, coding followed by decoding. The coding step is defined by the binary $k \times b$ code word matrix C that has one row (codeword) for each of k classes, with each column defining one of b sub-problems that use a different labeling. Assuming each element of C is a binary variable z a training pattern with target class ω_l for $l = 1...k$ is re-labeled as class Ω_1 if $C_{ij} = z$ and as class Ω_2 if $C_{ij} = \overline{z}$. The two super-classes Ω_1 and Ω_2 represent, for each column, a different decomposition of the original problem. For example, if a column of C is given by $[01001]^T$, this would naturally be interpreted as patterns from class 2 and 5 being assigned to Ω_1 with remaining patterns assigned to Ω_2. This is in contrast to the conventional One-versus-rest code, which can be defined by the diagonal $k \times k$ code matrix. In the decoding step, an unknown pattern is classified according to closest codeword.

In this chapter, features based on Principal Components Analysis (PCA Section 4) are used with Error-Correcting Output Coding (ECOC) and a weighted decoding strategy based on bootstrapping individual base classifiers is proposed. The principle behind weighted decoding is to reward classifiers that perform well. The weights in this study are fixed in the sense that none change as a function of the particular pattern being classified. Sometimes this is referred to as implicit data-dependence or constant weighting. It is generally recognized that a weighed combination may in principle be superior, but it is not easy to estimate the weights.

Although this chapter employs MLP ensembles, the techniques for OOB, feature selection and ECOC weighted decoding are suitable for any base classifier. The chapter is organised as follows. Section 2 discusses ensemble techniques and Bootstrapping, Section 3 the ECOC method including weighted decoding, Section 4 describes the database and design decisions for *au* classification, and Section 5 compares 2-class classification with weighted and conventional ECOC decoding.

2 Ensembles and Bootstrapping

For some classification problems, both two class and multiclass, it is known that the lowest error rate is not always reliably achieved by trying to design a single best classifier. An alternative approach is to employ a set of relatively simple sub-optimal classifiers and to determine a combining strategy that pools together the results. Although various systems of multiple classifiers have been proposed, most use similar constituent classifiers, which are often called base classifiers. A necessary condition for improvement by combining is that the results of the base classifiers are not too well correlated, as discussed in [18]. There are some popular approaches for reducing correlation that are based on perturbing feature sets, perturbing training sets or injecting randomness [19]. For example two well-known training set perturbation methods are Bagging [20] and Boosting [21]. All these perturbation techniques have in common that each base classifier handles the same problem in the sense that the class labelling is identical. There is another type of correlation reduction technique, aimed solely at multiclass problems, that perturbs class labels. In a method like Error Correcting Output Coding (ECOC) each base classifier solves a sub-problem that uses a different class labelling. Techniques like binary decision clustering [22] and pairwise coupling [23] may also be considered in this category.

The architecture envisaged is a simple MCS framework in which there are parallel MLP base classifiers, as shown in figure 1. For realistic problems, slow convergence and lack of guarantee of global minima are drawbacks of MLP training [26]. An MLP Ensemble offers a way of solving some of these problems [24]. The rationale is that it may be easier to optimise the design of a combination of relatively simple MLP classifiers than to optimise the design of a single complex MLP classifier. An MLP with random starting weights is a suitable base classifier since randomisation is known to be beneficial in the MCS context. Problems of local minima and computational slowness may be alleviated by the MCS approach of pooling together the decisions obtained from locally optimal classifiers. However, there is still the problem of tuning base classifiers.

Although it is known that diversity among base classifiers is a necessary condition for improvement in ensemble performance, there is no general agree-

ment about how to quantify the notion of diversity among a set of classifiers. Experimental evidence in [27] casts doubt on the usefulness of diversity measures for predicting ensemble accuracy. Diversity measures can be categorised into pair-wise and non-pair-wise, but to apply pair-wise measures to finding overall diversity it is necessary to average over the classifier set. These pair-wise diversity measures are normally computed between pairs of classifiers and take no account explicitly of the target labels. As explained in [28], the accuracy-diversity dilemma arises because when base classifiers become very accurate their diversity must decrease, so that it is expected that there will be a trade-off. A class separability measure that combines accuracy and diversity for two-class problems is described in [17]. For two-class problems, over-fitting may be detected by observing the class separability measure computed on the training set as it varies with base classifier complexity. In this chapter a modifed version of the class separability measure is proposed in Section 3.4 for the weighted decoding strategy.

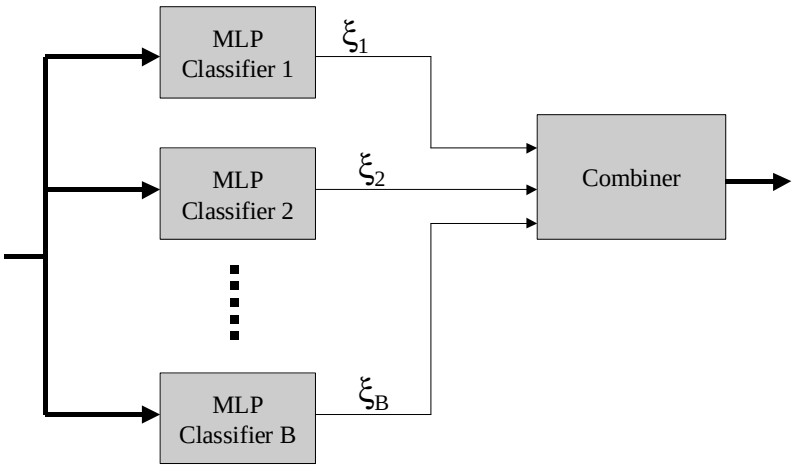

Fig. 1 Ensemble MLP Architecture

Bootstrapping is an ensemble technique which implies that if μ training patterns are randomly sampled with replacement, $(1\text{-}1/\mu))\mu \cong 37\%$ are removed with remaining patterns occurring one or more times. An advantage of Bootstrapping is that the Out-of-Bootstrap (OOB) error estimate may be used to tune base classifier parameters, and furthermore, the OOB is a good estimator of when to stop eliminating features [10]. Normally, deciding when to stop eliminating irrelevant features is difficult and requires a validation set or cross-validation techniques. The base classifier OOB estimate uses the patterns left out of training, and should be distinguished from the ensemble

OOB. For the ensemble OOB, all training patterns contribute to the estimate, but the only participating classifiers for each pattern are those that have not been used with that pattern for training (that is, approximately thirty-seven percent of classifiers). Note that OOB gives a biased estimate of the absolute value of generalisation error [29], but for tuning purposes the estimate of the absolute value is not important. The ensemble OOB estimate is incorporated into the ECOC decoding strategy in Section 3.2.

3 Error-Correcting Output Coding ECOC

There are several reasons for decomposing the original multiclass problem into separate and complementary two-class problems. Firstly, some accurate and efficient two-class classifiers do not naturally scale up to multiclass. Attention can then be focused on developing an effective technique for the two-class case, without having to consider explicitly the design and automation of the multiclass classifier. It is also hoped that the parameters of a simple classifier run several times are easier to determine than a complex classifier run once and may facilitate more efficient solutions. Finally, solving different 2-class sub-problems, perhaps repeatedly with random perturbation, may help to reduce error in the original problem.

It needs to be remembered however, that even if ECOC successfully produces accrate and diverse classifiers there is still the need to choose or design a suitable combining strategy. Bagging and Boosting originally used respectively the majority and weighted vote, which are both hard-level combining strategies. By hard-level we mean that a single-hypothesis decision is taken for each base classifier, in contrast with soft-level which implies a measure of confidence associated with the decision. The ECOC method was originally motivated by error-correcting principles, as discussed in Section 3.1 and used a Hamming Distance-based hard-level combining strategy. When it could be shown that ECOC produced reliable probability estimates [25], the decision-making strategy was changed to soft-level (L^1 norm equation (2)).

3.1 Motivation

First let us motivate the need for a suitable output coding by discussing the case of Multi-layer Perceptron (MLP) network. A single multiple output MLP can handle a multiclass problem directly. The standard technique is to use a k-dimensional binary target vector that represents each one of k classes using a single binary value at the corresponding position, for example $[0, ...0, 1, 0, ...0]$ which is sometimes referred to as one-per-class (OPC) encoding. The reason that a single multiclass MLP is not a suitable candidate for

use as a base classifier is that all nodes share in the same training, so errors are far from independent and there is not much benefit to be gained from combining. However a 2-class MLP is a suitable base classifier, and independence among classifiers is achieved by the problem decomposition defined by the coding method, as well as by injection of randomness through the starting weights. Of course, no guarantee can be given that a single MLP with superior performance will not be found, but the assumption is that even if one exists its parameters would be more difficult to determine.

An alternative to OPC is distributed output coding [15], in which k binary vectors are assigned to the k classes on the basis of meaningful features corresponding to each bit position. For this to provide a suitable decomposition some domain knowledge is required so that each classifier output can be interpreted as a binary feature which indicates the presence or otherwise of a useful feature of the problem at hand. The vectors are treated as code words so that a test pattern is assigned to the class that is closest to the corresponding code word. It is this method of assigning, which is analogous to the assignment stage of error-correcting coding, that provides the motivation for employing ECOC in classification.

The first stage of the ECOC method, as described in section 3.2, gives a strategy to decompose a multiclass problem into complementary two-class sub-problems. The second stage of the ECOC method is the decoding step, which was originally based on error-correcting principles under the assumption that the learning task can be modelled as a communication problem, in which class information is transmitted over a channel [16]. In this model, errors introduced into the process arise from various sources including the learning algorithm, features and finite training sample. The motivation for encoding multiple classifiers using an error-correcting code with Hamming Distance-based decoding was to provide error insensitivity with respect to individual classification errors. From the transmission channel viewpoint, we would expect that the one-per-class and distributed output coding matrices would not perform as well as the ECOC matrix, because of inferior error-correcting capability.

3.2 ECOC algorithm and OOB estimate

In the ECOC method, a $k \times b$ binary code word matrix C has one row (code word) for each of k classes, with each column defining one of b sub-problems that use a different labelling. Specifically, for the jth sub-problem, a training pattern with target class w_i $(i = 1...k)$ is re-labelled either as class Ω_1 or as class Ω_2 depending on the value of C_{ij} (typically zero or one). One way of looking at the re-labelling is to consider that for each column the k classes are arranged into two super-classes Ω_1 and Ω_2.

A test pattern is applied to the b trained classifiers forming vector

$$\mathbf{y} = [y_1, y_2, ...y_b]^T \tag{1}$$

in which y_j is the real-valued output of jth base classifier.

The distance between output vector and code word for each class is given by

$$L_i^1 = \sum_{j=1}^{b} |C_{ij} - y_j| \tag{2}$$

Equation (2) represents the L^1 norm or Minkowski distance, but if y_j in equ. 2 is taken as binary decision, this reduces to Hamming Distance. The decoding rule is to assign a test pattern to the class corresponding to closest code word $ArgMin_i(L_i^1)$.

A diagrammatic representation of the decoding step for a three class problem is given in figure 2, in which the test pattern is assigned to the code word that has minimum Hamming Distance compared with ECOC ensemble outputs.

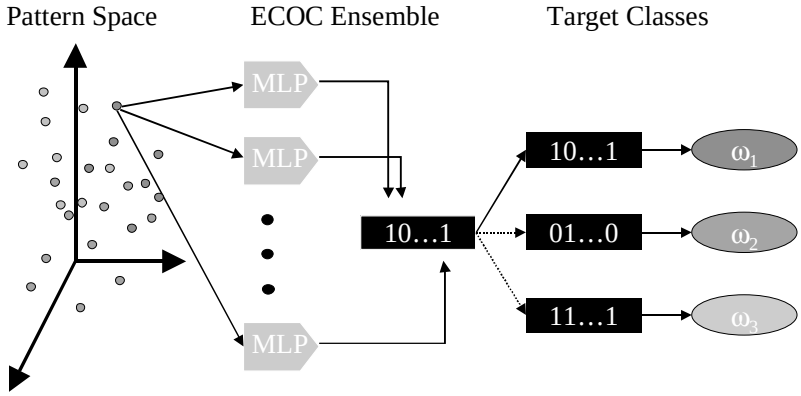

Fig. 2 Representation of the Hamming-based decoding step for a three class problem

To obtain the ensemble OOB estimate, the *pth* pattern is classified using only those classifiers that are in the set OOB_m, defined as the set of classifiers for which the *pth* pattern is out-of-bootstrap. For the OOB estimate, the summation in equ. 2 is therefore modified to

$$L_i^1 = \sum_{j \in OOB_m} |C_{ij} - y_j| \tag{3}$$

In other words it is necessary, for each pattern, to remember which classifier used that pattern for training. In the decoding step the columns of ECOC matrix C are removed if they correspond to classifiers that used the pth pattern for training. Therefore, on average, the column size of the C is about one third of the total number of classifiers.

3.3 Coding Strategies and Errors

When the ECOC technique was first developed it was believed that the ECOC code matrix should be designed to have certain properties to enable it to generalise well [12]. Various coding strategies have been proposed, but most ECOC code matrices that have been investigated previously are binary and problem-independent, that is pre-designed. Random codes have received much attention, and were first mentioned in [16] as performing well in comparison with error-correcting codes. In [12] random, exhaustive, hill-climbing search and BCH coding methods were used to produce ECOC code matrices for different column lengths. Random codes were investigated in [31] for combining Boosting with ECOC, and it was shown that a random code with a near equal column split of labels was theoretically better. Random codes were also shown in [30] to give Bayesian performance if pairs of code words were equidistant, and it was claimed that a long enough random code would not be outperformed by a pre-defined code. In [32] a random assignment of class to codeword was suggested in order to reduce sensitivity to code word selection.

According to error-correcting theory, an ECOC matrix designed to have d bits error-correcting capability will have a minimum Hamming Distance $2d + 1$ between any pair of code words. Assuming each bit is transmitted independently, it is then possible to correct a received pattern having d or fewer bits in error, by assigning the pattern to the code word closest in Hamming distance. While in practice errors are not independent, the experimental evidence is that application of the ECOC method does lead to reduced test error rate. From the perspective of error-correcting theory, it is therefore desirable to use a matrix C containing code words having high minimum Hamming Distance between any pair. Besides the intuitive reason based on error-correcting theory, this distance property has been confirmed from other perspectives. In [33] it was shown that a high minimum distance between any pair implies a reduced upper bound on the generalisation error, and in [30] it was shown for a random matrix that if the code is equidistant, then decision-making is optimum.

Maximising Hamming Distance between any pair of code words is intended to remove individual classification errors on the re-labelled training sets, but even if classifiers are perfect (Bayesian) there will still be errors due to decoding. The decoding errors can be categorised into those due to inability of

sub-problems to represent the main problem, and those due to the distance-based decision rule. Sub-problems are more independent and likely to benefit from combining if Hamming distance between columns is maximised, remembering that a column and its complement represent identical classification problems [12]. The distance-based effect on decoding error can be understood by analysing the relationship between decoding strategy and Bayes decision rule. Consider that the decomposition of a multiclass classification problem into binary sub-problems in ECOC can be interpreted as a transformation between spaces from the original output $\mathbf{q}$ to $\mathbf{p}$, given in matrix form by

$$\mathbf{p} = C^T \mathbf{q} \tag{4}$$

where $\mathbf{q}$ are individual class probabilities
Using the distance-based decision rule from (equ. (2)) and equ. (4)

$$L_i^1 = \sum_{j=1}^{b} |(\sum_{l=1}^{k} q_l C_{lj}) - C_{ij}| \tag{5}$$

and knowing that $\sum_{l=1}^{k} q_l = 1$ we have

$$L_i^1 = (1 - q_i) \sum_{j=1}^{b} |C_{ij} - C_{lj})| \tag{6}$$

From equation (6), we see that L_i^1 is the product of $1 - q_i$ and Hamming Distance between code words. When all pairs of code words are equidistant, minimising L^1 implies maximising posterior probability which is equivalent to Bayes rule

$$ArgMax_i(q_i) = ArgMin_i(L_i^1) \tag{7}$$

From the foregoing discussion, the main considerations in designing ECOC matrices are as follows

- 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)

From the theory of error-correcting codes [14] we know that finding a matrix with long code words, and having maximum and equal distance between all pairs of rows is complex. In [13] we compare random, equidistant and non-equidistant code matrices as number of columns is varied, but do not address explicitly the distance requirement between columns. Lack of experimental results on equidistant codes in previous work can be attributed to the difficulty in producing them. In [13] we produced equidistant codes by using the BCH method [14], which employs algebraic techniques from Galois

field theory. Although BCH has been used before for ECOC, our implementation was different in that we first over-produced the number of rows (BCH requires number to be power of 2), before selecting a subset of rows.

Although various heuristics have been employed to produce better binary problem-independent codes there appears to be little evidence to suggest that performance significantly improves by a clever choice of code, [16, 12]. A three-valued code [33] was suggested which allows specified classes to be omitted from consideration (don't care for third value), thereby permitting integrated representation of methods such as all-pairs-of-classes [23]. Theoretical and experimental evidence indicates that, providing a problem-independent code is long enough and base classifier is powerful enough, performance is not much affected [30]. In this chapter, a random code with near equal split of labels in each column is used with *b=200 and k=12*.

In [34] problem-dependent codes were investigated and it is claimed that designed continuous codes show more promise than designed discrete codes. A sub-class problem-dependent code design is suggested in [35], in which SFFS is used to split classes based on maximising mutual information between data and respective class labels. In this chapter, it is proposed that a useful way to consider problem-dependence is to consider it as a generate-and-test search in the coding-decoding strategy. The question then is to decide how much intelligence is put into the coding or the decoding step. In Section 3.4, we discuss a method of problem-dependent decoding, that uses a random code with weighted decoding.

3.4 Weighted Decoding

One way to introduce problem-dependence is through the decoding scheme. First, consider a modification of the decoding step in which each column of the ECOC matrix is weighted. In the test phase, if the *jth* classifier produces an estimated probability $\hat{q}_j$ that a test pattern comes from the super-class defined by the *jth* decomposition. The *pth* test pattern is assigned to the closest code word, for which weighted distance of the *pth* pattern to the *ith* code word is defined as

$$D_{pi} = \sum_{j=1}^{B} \alpha_{jl} \left| C_{ij} - \hat{q}_{pj} \right| where l = 1, ...k \tag{8}$$

where αjl in equ. 8 allows for *lth* class and *jth* classifier to be assigned a different weight.

Although this appears to be an obvious way to introduce weighted decoding, there is a difficulty in estimation of the values of the weights. In this chapter we propose a different weighted decoding scheme, that treats the outputs of the base classifiers as binary features [8]. By using the diagonal

matrix $C_{ij} = 1$ if and only if $i = j$ the problem is recoded as k 2-class problems where each problem is defined by a different binary-to-binary mapping. There are many strategies that may be used to learn this mapping, but we use a weighted vote with weights set by class-separability measure applied to the training data, which was defined in [17].

Let z_{mj} indicate the binary output of the *jth* classifier applied to the *mth* training pattern, so that the output of base classifiers for the *mth* pattern is given by

$$\mathbf{z_m} = [z_{m1}, z_{m2}, ...z_{mb}]^T \tag{9}$$

Assuming in equ. 9 that a value of 1 indicates agreement of the output with target label and 0 disagreement, we can define counts for *jth* classifier as follows

$$N_j^{11} = z_{mj} \wedge z_{nj}$$

$$N_j^{00} = \bar{z}_{mj} \wedge \bar{z}_{nj}$$

where the *mth* and *nth* pattern are chosen from different classes.

The weight for the *jth* output is then defined as

$$w_j = \frac{1}{K} \left(\sum_{allpairs} N_j^{11} - \sum_{allpairs} N_j^{00} \right) \tag{10}$$

where K is a normalization constant and the summation is over all pairs of patterns from different class. The motivation behind equ. 10 is that the weight is computed as the difference between positive and negative correlation with respect to target class. In [17] this is shown to be a measure of class separability.

4 Dataset and Feature Extraction

The Cohn-Kanade database [36] contains posed expression sequences from a frontal camera from 97 university students. Each sequence goes from neutral to target display but only the last image is *au* coded. Facial expressions in general contain combinations of action units (*aus*), and in some cases *aus* are non-additive (one action unit is dependent on another). To automate the task of *au* classification, a number of design decisions need to be made, which relate to the following 1) subset of image sequences chosen from the database 2) whether or not the neutral image is included in training 3)image resolution 4)normalisation procedure 5)size of window extracted from the image, if at all 6) features chosen for discrimination. Furthermore classifier type/parameters,

and training/testing protocol need to be chosen. Researchers choose different decisions in these areas, and in some cases are not explicit about which choice has been made. Therefore it is difficult to make a fair comparison with previous results.

We concentrate on the upper face around the eyes, involving *au1(inner brow raised), au2(outer brow raised), au4(brow lowered), au5(upper eyelid raised), au6(cheek raised),* and *au7(lower eyelid tightened).* We use the MLP ensemble, given in figure 1 and random training/test split of 90/10 repeated twenty times and averaged. Other decisions we made were:

1. All image sequences of size 640 x 480 chosen
2. Last image in sequence (no neutral) chosen giving 424 images, 115 containing *au1*
3. Full image resolution, no compression
4. Manually located eye centres plus rotation/scaling into 2 common eye coordinates
5. Window extracted of size 150 x 75 pixels centred on eye coordinates
6. Principal Components Analysis (PCA) applied to raw image with PCA ordering

With reference to decision 2, some studies use only the last image in the sequence but others use the neutral image to increase the numbers of *non-aus*. Furthermore, some researchers consider only images with single *au*, while others use combinations of *aus*. We consider the more difficult problem, in which neutral images are excluded and images contain combinations of *aus*. With reference to decision 4 there are different approaches to normalisation and extraction of the relevant facial region. To ensure that our results are independent of any eye detection software, we manually annotate the eye centres of all images, and subsequently rotate and scale the images to align the eye centres horizontally. A further problem is that some papers only report overall error rate. This may be mis-leading since class distributions are unequal, and it is possible to get an apparently low error rate by a simplistic classifier that classifies all images as *non-au*. For the reason we report area under ROC curve, similar to [5].

With reference to decision 6, PCA, or Karhunen-Loeve expansion [37], is a well-known statistical method that was applied to the coding and decoding of images in [38]. PCA minimises mean-squared error when a finite number of basis functions are used in the expansion. Furthermore the entropy, defined in terms of average squared coefficients used in the expansion, is also minimsised. The latter property is desirable for pattern recognition, in that features are clustered in the dimensionality reduction process. In the context of face recognition, the principal components of the distribution of faces is found, which is equivalent to finding the eigenvectors of the set of face images. Each face image in the training set may be represented by a linear combination of the 'eigenfaces', which is the name given to each eigenvector in the context of facial decomposition. The corresponding eigenvalues give a numerical value

of the importance of each eigenface for reconstruction of the original images. Our purpose is not reconstruction, but we can characterise each image by the highest eigenvalues thereby reducing dimensionality.

A summary of the method follows, but for full details see reference [38]. First each 2-dim array of pixels of the window defined in decision 5 is represented by a 1-dimensional vector of size 150 x 75 = 11250. Now it is desired to find the μ orthonormal vectors u_j with associated eigenvalues λ_j of the covariance matrix W of the training set. Given the training set of vectors $x_i, i = 1, \ldots, \mu$, $x_i \in R^D$,each belonging to one of **k** classes $\{\omega_1, \omega_2, \ldots, \omega_k\}$, we compute the mean face image given by

$$x_{mean} = 1/\mu \sum_{i=1}^{\mu} x_i \tag{11}$$

The mean image in equ. 11 is subtracted from each training set image to give

$$t_i = x_i - x_{mean} \tag{12}$$

Now the covariance matrix is given by

$$W = 1/\mu \sum_{i=1}^{\mu} x_i {x_i}^T = BB^T \tag{13}$$

where $B = [t_1 t_2 \ldots t_\mu]$ and W is size $\mu^2 \times \mu^2$. Following [38], we solve the simpler problem $B^T B$, which is $\mu \times \mu$, for obtaining the eigenvectors and eigenvalues.

Now the eigenvalues may be sorted to indicate the order of significance of the eigenvectors. Thus each face image is represented by the set of real numbers, or weights, corresponding to the P most significant eigenvalues, where P is to be determined experimentally (using OOB). The low-dimensional representation of each training pattern t_i, given by $u_k^T(t_i - x_{mean})$ for $k = 1 \ldots P$ is used to train the network. An unknown test pattern t_T is projected using $u_k^T(t_T - x_{mean})$ for $k = 1 \ldots P$ and input to the trained network for classification.

Table 1 ECOC super-classes of action units and number of patterns

ID	**sc1**	**sc2**	**sc3**	**sc4**	**sc5**	**sc6**	**sc7**	**sc8**	**sc9**	**sc10**	**sc11**	**sc12**
au	{}	1,2	1,2,5	4	6	1,4	1,4,7	4,7	4,6,7	6,7	1	1,2,4
#pat	149	21	44	26	64	18	10	39	16	7	6	4

The ultimate goal in *au* classification is to detect combination of *aus*. In the ECOC approach, a random 200×12 code matrix is used to treat each *au* combination as a different class. After removing classes with less than four

patterns this gives a 12-class problem with *au* combinations as shown in Table 1. In Section 5, to compare the ECOC results with 2-class classification, we compute test error by interpreting super-classes as 2-class problems, defined as either containing or not containing respective *au*. For example, *sc2, sc3, sc6, sc11, sc12* in Table 1 are interpreted as *au1*, and remaining super-classes as *non-au1*

5 Experiments on Cohn-Kanade Database

This Section contains three sets of example experiments aimed at 2-class and multi-class formulations of *au* classification, for the Cohn-Kanade database described in Section 4. The goal is to demonstrate that weighted decoding ECOC outperforms conventional ECOC decoding, when base classifiers are tuned using OOB estimate. For experiments on UCI benchmark data [39] that demonstrate the use of OOB for ECOC ensemble design and provide an experimental comparison of feature selection schemes for ECOC ensembles, the reader is referred to [9, 10].

In the experiments in this Section, the MLP ensemble uses two hundred single hidden-layer MLP base classifiers, with Levenberg-Marquardt training algorithm [40] and default parameters. Random perturbation of the MLP base classifiers is caused by different starting weights on each run, combined with bootstrapped training patterns. In our framework, we vary the number of hidden nodes, with a single node for linear perceptron, and keep the number of training epochs fixed at 20. For a comparison of feature extraction, the first experiment uses Gabor features [2], which have generally been found to give better performance than PCA for single classifiers [41]. The second and third experiments use PCA as described in Section 4.

In the first experiment, which comes from [6], we use RFE with MLP weights to rank Gabor features. RFE is a simple algorithm [11], and operates recursively as follows:

1. Rank the features according to a suitable feature-ranking method
2. Identify and remove the r least ranked features

If $r \geq 2$, which is usually desirable from an efficiency viewpoint, this produces a feature subset ranking. The main advantage of RFE is that the only requirement to be successful is that at each recursion the least ranked subset does not contain a strongly relevant feature [42]. It was found that lower test error was obtained with non-linear base classifier and figure 3 shows test error rates, using an MLP ensemble with 16 nodes. The minimum base error rate for 90/10 split is 16.5 percent achieved for 28 features, while the ensemble is 10.0 percent at 28 features. Note that for 50/50 split there are too few training patterns for feature selection to have much effect. Since class distributions are unbalanced, the overall error rate may be mis-leading,

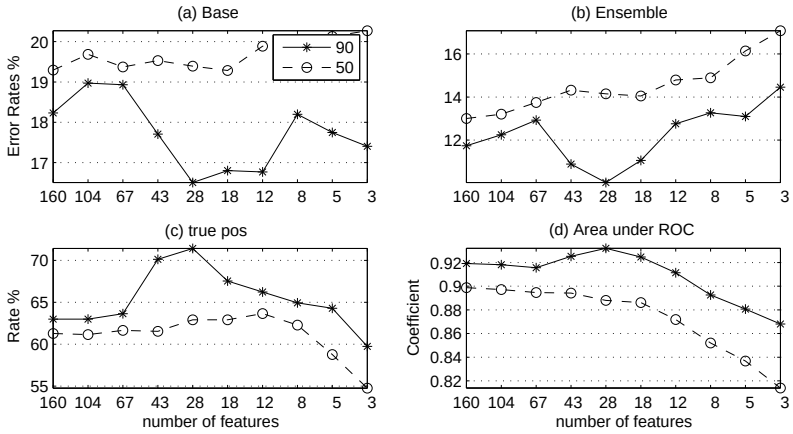

Fig. 3 Mean test error rates, True Positive and area under ROC for RFE MLP ensemble au1 classification 90/10. 50/50 train/test split

as explained in Section 4. Therefore, we show the true positive rate in Figure 3 c) and area under ROC in Figure d). Note that only 71 percent of au1s are correctly recognised. However, by changing the threshold for calculating the ROC, it is clearly possible to increase the true positive rate at the expense of false negatives.

Table 2 Mean best test error rates for 2-class problems and area under ROC showing nodes/features for au classification with optimized PCA features and MLP ensemble

	2-classTestError %	***2-classarea underROC***
au1	9.4/16/28	0.97/16/36
au2	3.5/4/36	0.99/16/22
au4	9.1/16/36	0.95/16/46
au5	5.5/1/46	0.97/1/46
au6	10.5/1/36	0.94/4/28
au7	10.3/1/28	0.92/16/60
mean	**8.1**	**0.96**

The second set of experiments detects *au1, au2, au4, au5, au6, au7* using six different 2-class classification problems, where the second class contains all patterns not containing respective *au*. The MLP ensemble uses majority vote combining rule and PCA features are used to train the base classifiers. The best error rate of 9.4 percent for *au1* was obtained with 16 nodes and 28 features. The 9.4 percent error rate for *au1* is equivalent to 73 percent of *au1s* correctly recognised. The best ensemble error rate, area under ROC with number of features and number of nodes for all upper face *aus* are shown in Table 2. Note that number of nodes for best area under ROC is generally

higher than for best error rate, indicating that error rate is more likely to be susceptible to over-fitting.

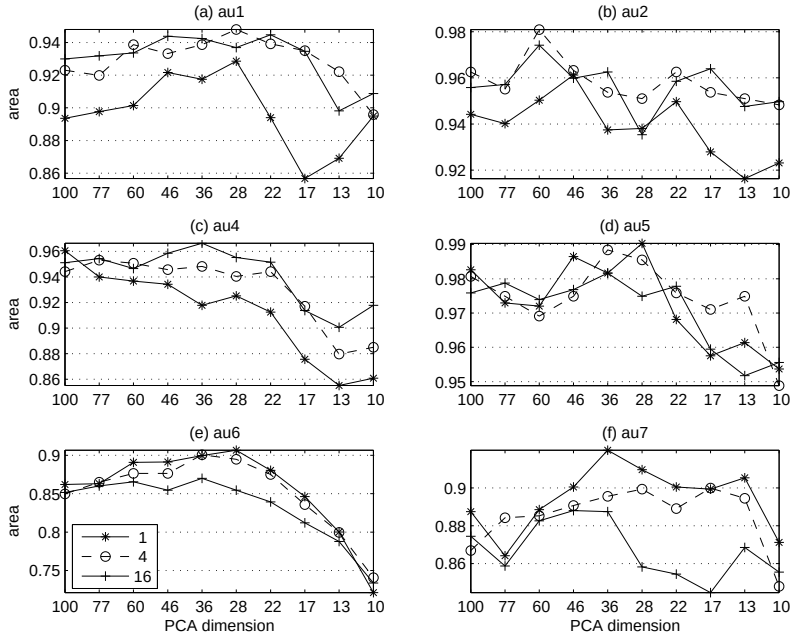

Fig. 4 Area under ROC for weighted decoding ECOC MLP ensemble [1,4,16] hidden nodes 20 epochs versus number PCA features (logscale)

Table 3 Mean best test error rates and area under ROC for ECOC L1 norm decoding showing nodes/features for au classification with optimized PCA features and MLP ensemble

	ECOCTestError %	***ECOCarea underROC***
au1	10.3/1/10	0.92/16/46
au2	3.4/1/36	0.96/16/28
au4	12.0/16/28	0.92/4/28
au5	3.6/16/36	0.99/1/36
au6	13.1/1/77	0.88/1/77
au7	11.6/1/28	0.89/4/46
mean	**9.0**	**0.93**

The third set of experiments uses ECOC method described in Section 3, and figure 4 shows area under ROC for the six *aus*, as number of PCA features is reduced. Table 3 shows best *L1 norm* decoding classification error and area under ROC, while Table 4 shows respective weighted decoding. It

Table 4 Mean best test error rates and area under ROC for ECOC weighted decoding showing nodes/features for au classification with optimized PCA features and MLP ensemble

	ECOC WeightedError %	***ECOC WeightedROC***
au1	9.2/4/36	0.94/16/36
au2	2.8/16/22	0.98/1/46
au4	9.5/1/28	0.94/4/28
au5	3.2/1/36	0.99/1/36
au6	12.8/1/77	0.90/1/28
au7	10.9/4/46	0.92/1/36
mean	**8.1**	**0.95**

may be seen that weighted consistently outperforms *L1 norm* decoding. Also it may be seen from Table 2 that 2-class classification with optimized PCA features on average slightly outperforms ECOC. However, the advantage of ECOC is that all problems are solved simultaneously, and furthermore the combination of *aus* is recognized. As a 12-class problem, the mean best error rate over the twelve classes defined in Table 1 is 38.2 percent, showing that recognition of combination of *aus* is a difficult problem.

6 Discussion

The results for upper face *aus*, shown in Table 2 and Table 4, are believed to be among the best on this database (recognising the difficulty of making fair comparison as explained in Section 3).There are two possible reasons why the ECOC decoding strategy works well. Firstly, the data is projected into a high-dimensional space and therefore more likely to be linearly separable [43]. Secondly, although the full training set is used to estimate the weights, each base classifier is bootstrapped and therefore is trained on a subset of the data, which guards against over-fitting. As indicated in Section 2, bootstrapping also facilitates the OOB estimate for removing irrelevant features without validation.

7 Conclusion

In this chapter, an information theoretic approach of coding and decoding has been applied to both feature extraction and multi-class classification. For upper face *au* classification, weighted decoding ECOC achieves comparable performance to optimized 2-class classifiers. However, ECOC has the advantage that all *aus* are detected simultaneously, and further work is aimed at

determining whether problem-dependent rather than random codes can improve results. Furthermore, the ultimate aim of this work is to apply the technique to improve robustness of face verification systems, and to better recognise driver fatigue.

References

1. Y. Tian, T. Kanade and J. F. Cohn, Recognising action units for facial expression analysis, IEEE Trans. PAMI 23(2), 2001, 97-115.
2. G Donato, M S Bartlett, J C Hager, P Ekman and T J Sejnowski, Classifying facial actions, IEEE Trans. PAMI 21(10), 1999, 974-989.
3. Bartlett, M.S. Littlewort, G. Lainscsek, C. Fasel, I. Movellan, J. Machine learning methods for fully automatic recognition of facial expressions and facial actions, IEEE Conf. Systems, Man and Cybernetics, Oct 2004, Vol. 1, 592- 597.
4. P. Silapachote, D. R. Karuppiah, and A. R. Hanson, Feature Selection using Adaboost for Face Expression Recognition, Proc. Conf. on Visualisation, Imaging and Image Processing, Marbella, Spain, Sept. 2004, 84-89.
5. M S Bartlett, G Littlewort, M Frank, C Lainscsek, I Fasel and J Movellan, Fully automatic facial action recognition in spontaneous behavior, Proc 7th Conf. On Automatic Face and Gesture Recognition, 2006, ISBN 0-7695-2503-2, 223-238.
6. T Windeatt, K Dias, Feature-ranking ensembles for facial action unit classification, IAPR Third Int. Workshop on artificial neural networks in pattern recognition, Paris, July, 2008, Springer Verlag Berlin Heidelberg, LNAI 5064, 267-279.
7. Wang W., Jones P. and Partridge D. Assessing the impact of input features in a feedforward neural network, Neural Computing and Applications 9, 2000, 101-112.
8. T. Windeatt, R S Smith, K. Dias, Weighted Decoding ECOC for facial action unit classification, Workshop on Supervised and Unsupervised Ensemble Methods and their Applications, European Conf. Artificial Intelligence, Patras, Greece, 2008, 26-30.
9. T. Windeatt., M. Prior, N. Effron, N. Intrator, Ensemble-based Feature Selection Criteria, Proc. Conference on Machine Learning Data Mining MLDM2007, Leipzig, July 2007, ISBN 978-3-940501-00-4, pp 168-182
10. T Windeatt, M Prior, Stopping Criteria for Ensemble-based Feature Selection, Proc. 7th Int. Workshop Multiple Classifier Systems, Prague May 2007, Lecture notes in computer science, Springer-Verlag, 271-281.
11. Guyon I., Weston J., Barnhill S. and Vapnik V., Gene selection for cancer classification using support vector machines, Machine Learning 46(1-3), 2002, 389-422.
12. T. G. Dietterich,G. Bakiri, Solving multiclass learning problems via error-correcting output codes, J. Artificial Intelligence Research 2, 1995, 263-286
13. T Windeatt and R Ghaderi, Coding and Decoding Strategies for multiclass learning problems, Information Fusion, 4(1), 2003, 11-21.
14. W.W. Peterson and J.R. Weldon. Error-Correcting Codes. MIT press, Cambridge, MA, 1972.
15. T.J. Sejnowski and C.R. Rosenberg. Parallel networks that learn to pronounce english text. Complex systems, 1:145168, 1987.
16. T.G Dietterich and G. Bakiri. Error-correcting output codes: A general method for improving multiclass inductive learning programs. In Proceedings of the Ninth National Conference on Artificial Intelligence, AAAI Press, 1991, 572577.
17. T Windeatt, Accuracy/Diversity and Ensemble Classifier Design, IEEE Trans. Neural Networks 17(5), 2006, 287-297.

18. T. Windeatt, Diversity Measures for Multiple Classifier System Analysis and Design, Information Fusion, 6 (1), 2004, 21-36.
19. T.G. Dietterich. Ensemble methods in machine learning. In J.Kittler and F.Roli, editors, Multiple Classifier Systems, MCS2000, pages 115, Cagliari, Italy, 2000. Springer Lecture Notes in Computer Science.
20. L. Breiman. Bagging predictors. Machine Learning, 24(2):123140, 1997.
21. Y. Freund and R.E. Schapire. A decision-theoretic generalisation of on-line learning and application to boosting. Journal of computer and system science 55, 1997, 119 139.
22. C.L.Wilson, P.J. Grother, and C.S. Barnes. Binary decision clustering for neuralnetwork- based optical character recognition. Pattern Recognition, 29(3):425 437, 1996.
23. T. Hastie and R Tibshirani. Classification by pairwise coupling. The annals of statistics, 2:451471, 1998.
24. L. K. Hansen, P. Salamon, Neural Network Ensembles, IEEE Trans. Pattern Analysis and Machine Intelligence, 12(10), 1990, 993-1001.
25. E.B. Kong and T.G. Diettrich. Error-correcting output coding corrects bias and variance. In 12th Int. Conf. of Machine Learning, pages 313321, San Fransisco, 1995. Morgan Kaufmann.
26. T. Windeatt, Ensemble MLP Classifier Design, chapter in book: Studies in Computational Intelligence, Vol. 137/2008, Springer Verlag Berlin Heidelberg, 2008, 133-147.
27. L. I. Kuncheva and C.J. Whitaker, Measures of Diversity in Classifier Ensembles, Machine Learning 51, 2003, 181-207.
28. T. Windeatt, Spectral Measure for Multi-class Problems, Proc. 5th Int. Workshop Multiple Classifier Systems, Editors: F. Roli, J. Kittler, T. Windeatt, Cagliari, Italy, June, 2004, Lecture notes in computer science, Springer-Verlag, 184-193.
29. T. Bylander, Estimating generalisation error two-class datasets using out-of-bag estimate, Machine Learning 48, 2002,287-297.
30. G. M. James and T. Hastie. The error coding method and PICTs. Computational and Graphical Statistics, 7:377387, 1998.
31. R.E. Schapire. Using output codes to boost multiclass learning problems. In 14th International Conf. on Machine Learning, pages 313321. Morgan Kaufman,1997.
32. T. Windeatt and R. Ghaderi. Multi-class learning and error-correcting code sensitivity. Electronics Letters, 36(19):16301632, Sep 2000.
33. E.L. Allwein, R.E. Schapire, and Y. Singer. Reducing multi-class to binary: A unifying approach for margin classifiers. Machine learning research, 1:113141,2000.
34. K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. Machine Learning 47(2-3),2002, 201-233.
35. S. Escalara, D. M. J. Tax, O. Pujol, P. Radeva, R. W. Duin, Subclass Problem-Dependent Design for Error-Correcting Output Codes, IEEE Trans. PAMI 30(8), 2008, 1041-1054.
36. T. Kanade, J. F. Cohn and Y. Tian, Comprehensive Database for facial expression analysis, Proc. 4th Int. Conf. automatic face and gesture recognition, Grenoble, France, 2000, 46-53.
37. J. T. Tou, R C Gonzales, Pattern Recognition Principles, Addison-Wesley, 1974.
38. M. A. Turk, A. P. Pentland, Face Recognition using Eigenfaces, Proc. Int. Conference on Computer Vision and Pattern Recognition, Maui, USA, 1991, 586-591.
39. C.J. Merz and P. M. Murphy, UCI Repository of Machine Learning Databases, 1998, http://www.ics.uci.edu/ mlearn/MLRepository.html
40. Haykin S., Neural Networks: A Comprehensive Foundation, Prentice Hall, 1999.
41. Y. Tian, T. Kanade, J. F. Cohn, Evaluation of Gabor-Based Facial Action Unit Recognition in Image Sequences of Increasing Complexity, Proc Int Conf. on Automatic Face and Gesture Recognition FGR02, 2002.

42. L. Yu, H. Liu, Efficient feature selection via analysis of relevance and redundancy, Journal of Machine Learning Research 5, 2004, 1205-1224.
43. T.M. Cover, Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition, IEEE Trans. Information Theory, vol. EC-14, 1965, 326-334.
44. G. Valentini, T. G. Dietterich, Bias-variance analysis of Support Vector Machines for the development of SVM-based ensemble methods, *Journal of Machine Learning Research*, 5, 2004, MIT Press, 725-775.