Error-correcting output codes based ensemble feature extraction

Guoqiang Zhong, Cheng-Lin Liu*

National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, No. 95 Zhongguancun East Road, Beijing 100190, PR China

A R T I C L E   I N F O

Article history:
Received 21 May 2012
Received in revised form 22 September 2012
Accepted 20 October 2012
Available online 30 October 2012

Keywords:
Feature extraction
Ensemble learning
Error-correcting output codes (ECOC)
Meta learner

A B S T R A C T

This paper proposes a novel feature extraction method based on ensemble learning. Using the error-correcting output codes (ECOC) to design binary classifiers (dichotomizers) for separating subsets of classes, the outputs of the dichotomizers are linear or nonlinear features that provide powerful separability in a new space. In this space, the vector quantization based meta classifier can be viewed as an ECOC decoder, where each learned prototype of a class can be seen as a codeword of the class in the new representation space. We conducted extensive experiments on 16 multi-class data sets from the UCI machine learning repository. The results demonstrate the superiority of the proposed method over both existing ECOC approaches and classic feature extraction approaches. In particular, the decoding strategy using a meta classifier is shown to be more computationally efficient than the linear loss-weighted decoding in state-of-the-art ECOC methods.

© 2012 Elsevier Ltd. All rights reserved.

1. Introduction

Feature extraction is an essential issue in many areas of pattern recognition and machine learning. Informative features extracted from data can benefit the subsequent learning, analysis and recognition. Traditional linear feature extraction methods, such as principal components analysis (PCA) [1] and linear discriminant analysis (LDA) [2], have been widely used for finding a linear subspace of the data. However, they may fail to discover the intrinsic low-dimensional structure when data lie on a nonlinear manifold. Since the publication of two seminal manifold learning algorithms, isometric feature mapping (Isomap) [3] and locally linear embedding (LLE) [4], a plenty of nonlinear manifold learning methods have been developed [5–9]. However, most of them are unsupervised and cannot deal with the out-of-sample problem easily [10]. Moreover, most of the existing feature extraction methods learn new representations directly from the data, without integrating the output information of classifiers. From our observation, probabilistic outputs of a group of classifiers are fairly effective features to indicate to which class a sample belongs. This motivates us to explore an ensemble feature extraction method using a set of basic classifiers.

The learning algorithms that construct a set of classifiers and then predict new data points based on these classifiers are generally called as ensemble learning [11]. The notable ensemble learning methods include bagging [12], boosting [13], error-correcting output codes (ECOC) [14], and stacking (stacked generalization) [15], among others. Many works in the literature have proven the advantages of ensemble over single classifiers, both theoretically and experimentally. Meanwhile, ensemble learning methods have been successfully applied to many real-world problems, such as optical character recognition (OCR) [16,17], face recognition [18], speaker recognition [19], remote sensing [20], and multimodal interaction [21]. In this paper, we focus on the ECOC technique, which is a framework for combining binary classifiers (dichotomizers) to address multi-class problems. Based on the ECOC framework, we present an ensemble feature extraction method to learning discriminative representations of the data.

The ECOC framework generally includes two steps: the coding step and the decoding step. The coding strategies include one-versus-all [22], one-versus-one [23], data-driven ECOC [24], discriminant ECOC (DEOC) [25], and ECOC-optimizing node embedding (ECOCONE) [26]. Among them, one-versus-all and one-versus-one are problem-independent ECOC design methods, whilst data-driven ECOC, DEOC and ECOCONE are problem-dependent. The commonly used decoding strategies are Hamming decoding [22] and Euclidean decoding [23]. Some researchers have introduced loss-based function [27] or probabilities [28,29] in decoding. Recently, Escalera et al. [30] proposed two novel ternary ECOC decoding strategies, β-density decoding and loss-weighted decoding, and showed their advantages over the state-of-the-art decoding strategies. To the best of our knowledge, however, there has not been a published work that attempts to integrate feature extraction into the ECOC framework. That is, all the existing methods train dichotomizers in the data space and combine the classifiers using decoding strategies, but not try to explore the intrinsic geometric structure of the data belonging to different classes. On the other hand, learning new features via the
combination of the basic classifiers can significantly benefit the classification accuracy. In addition, as far as we know, all the existing ECOC methods endow only one codeword for each class. Actually, from the perspective of vector quantization [31], if the data distribution is relatively complex, more than one codewords (or called prototypes) can be helpful to characterize the distribution of the class.

In this paper, we propose a novel ensemble feature extraction method based on the ECOC framework. It takes advantage of the discrimination ability of the dichotomizers for separating subsets of classes designed by ECOC. Accordingly, we call it ECOC based ensemble feature extraction (ECOC-EFE). In ECOC-EFE, the new representation of a datum is actually the probabilistic outputs of the combined dichotomizers, where each element indicates the probability that the datum belongs to the corresponding positive class. Based on the extracted features, we employ a generalized learning vector quantization (GLVQ) classifier [32] as a meta learner for classification in the new feature space. The learned prototypes of a class by GLVQ can be viewed as codewords of the class in the new space, while the classification of the meta learner can be viewed as a decoding step, corresponding to that in the ECOC framework. From the viewpoint of ensemble learning, our method can be considered as a new framework for multi-class learning problems, which includes four steps—ECOC coding, feature extraction, recoding (or meta learning) and decoding (or classification). In experiments on 16 data sets from the UCI machine learning repository using linear and nonlinear binary classifiers, the proposed method is demonstrated superior classification performance compared to state-of-the-art feature extraction and ECOC decoding methods.

In the remainder of this paper, Section 2 gives a brief review of related works on feature extraction and ECOC based ensemble learning; Section 3 introduces the notation used in this paper; Section 4 describes the proposed method in detail; Section 5 presents the experimental results; Section 6 concludes this paper with remarks.

2. Related works

Over the past few decades, many feature extraction methods have been proposed, such as PCA [1], LDA [2], kernel PCA (KPCA) [33] and generalized discriminant analysis (GDA) [34]. From the nature of feature representation, these methods can be classified into two categories: linear and nonlinear. Linear methods, such as PCA and LDA, generally find a linear projection to map the data into a low-dimensional subspace. In contrast, nonlinear methods, such as KPCA, GDA and some manifold learning methods, usually connect the original space with the feature space or low-dimensional manifold via a nonlinear function. Compared to linear methods, nonlinear feature extraction methods can be applied to more complex data, but they mostly suffer from a common problem that they can hardly derive the exact mapping function except the coordinates of the training data in the new space. Moreover, most of the existing linear and nonlinear feature extraction methods do not explore the class structure of the data adequately to yield sufficiently high classification accuracy.

The ECOC framework is to combine binary classifiers (dichotomizers), such as support vector machines (SVMs) [35] and Adaboost [36], to solve multi-class classification problems. Dietterich and Bakiri [14] presented the basic ECOC framework represented using a coding matrix of binary symbols. Each column of the coding matrix represents a binary partition of the whole classes in two subsets \((-1, +1)\). Alternatively, each row of the matrix is a codeword assigned to the corresponding class. The one-versus-all [22] strategy is a special case of the binary-symbol-based ECOC. Afterwards, Allwein et al. [27] extended the coding strategy by introducing a third symbol ‘0’, which allows some classes to be neglected by the dichotomizers and leads to the increment of subgroups of classes to be considered in the ternary ECOC framework. The one-versus-one (pairwise) classification strategy [23] can be viewed as a special case of the ternary ECOC framework. Most of the ECOC methods specify the coding matrix just in the coding step, i.e., predefine it independently of the problem, such as the above one-versus-all, one-versus-one, and the dense and sparse random coding strategies [27].

Considering the nature of classification problem or the structure of the data can lead to better coding matrix design. The first problem-dependent ECOC design method was proposed by Uttsch and Weichselberger [37]. However, their experimental results showed that for many multi-class problems, the best performance was still given by the one-versus-all method. Crammer and Singer [38] have reported improvement in the design of ECOC matrix, but they proved that finding the optimal discrete codes is NP-hard with the number of classes. The discriminant ECOC (DECOC) [25] is a heuristic method for learning the coding matrix by exploring the hierarchical structure of the class space. It generates a binary tree structure for the hierarchical partition by maximizing a discriminative criterion. In addition to its superior classification performance, DECOC leads to a very compact codeword with length \(C-1\), whereas \(C\) is the number of classes. Pujol et al. [26] proposed a new approach that improves the initial ECOC matrix in a sub-optimal way. It creates new dichotomizers by minimizing the confusion matrix among classes guided by a validation subset. A length of \(2C\) bits for the codeword has been suggested. Recently, Escalera et al. [39] proposed a method to redefine the ECOC matrix without re-training. This re-coding strategy can be applied over any coding design.

However, all these ECOC coding strategies, either problem-dependent or problem-independent, suffer from two shortcomings: they do not use the combined dichotomizers to extract useful features of the data, and they endow only one codeword for each class. Addressing these two problems may lead to significant improvement of classification accuracy.

We noticed that some researchers have tried meta learning for combining binary classifiers for multi-class classification [40–42], Savický and Fürnkranz [40], Lezoray and Cardot [41] combine pairwise (one-versus-one) classifiers using meta classifier (C4.5) trained with stacking. They observed more or less improved classification performance compared to the other pairwise coupling and fusion algorithms. Shiraishi and Fukumizu [42] combine one-versus-all or one-versus-one binary classifiers using the multinomial logistic regression as the meta learner. These methods were tested on binary classifiers in only one-versus-one or one-versus-all coding strategy, and were not compared to the advanced decoding methods like the recent loss-weighted decoding methods [30]. The results of [30] show that the ECOCONE coding strategy mostly give the best classification performance and the loss-weighted decoding is among the best for combining binary classifiers. Hence, we build our ensemble method on the ECOCONE coding strategy and compare with the loss-weighted decoding.

On the hand of feature extraction, Rueda et al. [43] proposed a method to extract linear subspace features from pairs of classes and combine the two-class decisions by voting and meta learning. This falls in the framework of ECOC but is limited in the sense that it considers only one-versus-one encoding and linear feature extraction.

3. Notation

We use boldface uppercase letters to denote matrices, such as \(K\), and boldface lowercase letters to denote vectors, such as \(v\). The \(i\)th row of a matrix \(K\) is denoted as \(K_i\). \(K_{ij}\) denotes the entry at the...
ith row and jth column of $K$, $v_i$ denotes the ith entry of $v$, $K^T$ and $v^T$ are the transpose of $K$ and $v$, respectively. $\text{tr}(K)$ is the trace of matrix $K$.

For multi-class problems, we denote the given training data as $X = [(x_1, c_1), (x_2, c_2), \ldots, (x_{N_{trn}}, c_{N_{trn}})]$, where $N_{trn}$ is the number of training samples, $x_i \in \mathbb{R}^D$ is a D-dimensional input vector, and $c_i$ is the class label of $x_i$. The number of classes is denoted as $C$, i.e., $c_i \in [1, C]$. The test data is denoted as $Y = [y_1, y_2, \ldots, y_{N_{tst}}]$, where $N_{tst}$ is the number of test samples. We use the matrices $X = [x_1, x_2, \ldots, x_{N_{trn}}]^T$ and $Y = [y_1, y_2, \ldots, y_{N_{tst}}]^T$ to denote the training data matrix and test data matrix, respectively.

For ECOC, we denote the coding matrix as $M$, where $M_{ij} \in [-1, 0, 1]$. The length of codewords is denoted as $p$, i.e., $M \in [-1, 0, 1]^{C \times p}$.

4. ECOC based ensemble feature extraction (ECOC-EFE)

In this section, we describe the details of the proposed method ECOC-EFE. For clarity, we present each step of ECOC-EFE—coding, feature extraction, recoding and decoding, respectively in the following subsections.

4.1. Coding

The coding step of ECOC-EFE is to design an ECOC matrix specifying the dichotomizers to be combined for multi-class classification. For completeness, we analyze the advantages and disadvantages of some existing coding strategies and describe the strategies used in our experiments.

The widely used coding strategies include one-versus-all [22], one-versus-one [23], dense random [27], sparse random [27], DECOC [25], and ECOCONE [26]. Among them, one-versus-all and dense random are binary-symbol-based strategies, while the others are ternary-symbol-based ones. On the other hand, the one-versus-all, one-versus-one, dense random and sparse random are problem-independent coding strategies, while DECOC and ECOCONE are problem-dependent ones.

The one-versus-one strategy can be considered as the most effective with respect to (w.r.t.) the training of the combined dichotomizers. However, it is the one that needs to combine the most number of dichotomizers, which is $O(C^2)$, against $O(C)$ for one-versus-all and DECOC. For large number of classes, the complexity of dichotomizers training and combining is formidable. On the other hand, the one-versus-all strategy, though combines $C$ dichotomizers only, needs to use all the training data to learn each dichotomizer. For some dichotomizers (such as the SVM), the training complexity is $O(N_{trn}^2)$, or higher, where $N_{trn}$ is the number of training samples. DECOC pursues a tree structure of the classes, and only involves all the classes in the first column of the ECOC matrix, whose length of codeword is fixed to $C−1$. Thus, DECOC can be considered as an ideal coding strategy for its compactness. ECOCONE is a method that extends incrementally an initial, any type of ECOC matrix. Hence, ECOCONE can be considered as a useful design to capture the discrimination between the subsets of the classes. Based on the experimental results reported in [30] that dense random and sparse random rarely outperform the four coding strategies discussed above, we will not take them into account in our discussion and experiments.

For our ECOC-EFE, we adopt ECOCONE as the coding strategy, which is initialized via a DECOC configuration. Thus, the length of the codewords in our method is generally around $C$, resulted from the extension of the DECOC initialization. For more algorithmic details of DECOC and ECOCONE, refer to [25,26]. The choice of the coding strategy actually determines the dimensionality of the new feature space of ECOC-EFE since the length of the codeword, i.e., the number of combined dichotomizers, is equal to the dimensionality of the new feature space. We use DECOC to initialize the ECOCONE algorithm such that the dimensionality of the resulting feature space is moderate. Due to the adaptation of ECOCONE, the dimensionality is not necessarily limited to $C−1$, i.e., the dimensionality can be intelligently learned depending on the data structure and separability so as to maximize the discrimination between classes.

After the ECOC matrix is specified, we can train $p$ dichotomizers independently according to the columns of the ECOC matrix, where $p$ is the length of the codeword, and as well, the dimensionality of the new feature space.

4.2. Feature extraction

On training dichotomizers according to the ECOC matrix, the outputs of the dichotomizers on a new input vector can be taken as the features in a new space. In principle, all types of binary classifiers can be used for this purpose. We nevertheless adopt the state-of-the-art SVM [35] as the base classifier and transform its output to probabilistic confidence using the sigmoid function [44–46]. Using the SVM as dichotomizer enables performance comparison with the state-of-the-art multi-class SVM classifier and kernel feature extraction (RPCA and GDA). Transforming the outputs of dichotomizers to probabilities (each value indicates the probability that the input pattern belongs to the corresponding positive class), as a measure of feature normalization, may help improve the classification performance in the new feature space. More specifically, we use the linear SVM dichotomizer for linear feature extraction (ECOC-LFE) and the SVM with radial basis function (RBF) kernel for nonlinear feature extraction (ECOC-NLE). The RBF kernel has been shown to be among the best choices for nonlinear SVM for classification.

Before introducing the details of feature extraction, we briefly outline the linear and nonlinear SVMs. The linear SVM solves a quadratic programming problem

$$
\min_{w,b,c} \quad \frac{1}{2} ||w||^2 \\
\text{s.t.} \quad \sum_{i=1}^{N_{trn}} c_i f(x_i) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, N_{trn},
$$

where $c_i \in \{+1, -1\}$, $w$ is the weight vector, $\xi_i$'s are the slack variables. The binary discriminant function is

$$
f(x) = w^T x + b.
$$

We solve this problem and obtain

$$
w = \frac{1}{N_{trn}} \sum_{i=1}^{N_{trn}} c_i x_i,
$$

and

$$
b = \frac{1}{N_{trn}} \sum_{i=1}^{N_{trn}} (w^T x_i - c_i),
$$

where $c_i$'s are the non-negative Lagrange multipliers, $N_{trn}$ is the number of support vectors. The dual form of Problem (1) can be written as

$$
\max_{\alpha} \quad \sum_{i=1}^{N_{trn}} \frac{1}{2} \alpha_i\sum_{ij} a_i a_j c_i c_j x_i x_j - \frac{1}{2} \alpha_i \sum_{ij} a_i a_j c_i c_j k(x_i, x_j)
$$

s.t. \quad 0 \leq \alpha_i \leq \lambda, \quad i = 1, \ldots, N_{trn},

$$
\sum_{i=1}^{N_{trn}} \alpha_i c_i = 0,
$$

where $\lambda$ is the upper bound of $\alpha_i$.
where \( k(x_i, x_j) = x_i^T x_j \) is the linear kernel function, \( \lambda \) is a constant number and \( z = [z_1, \ldots, z_{N_{trn}}]^T \) is the vector of Lagrange multipliers.

Replacing the linear kernel function in Problem (5) with a nonlinear kernel function gives the formulation of the nonlinear SVM. We use the RBF kernel

\[
k(x_i, x_j) = \exp(-\gamma^{-1}||x_i - x_j||^2),
\]

where \( \gamma \) is the parameter for the kernel function. On solving Problem (5), we can obtain \( z_i \)'s, \( b \) and the discriminant function

\[
f(x) = \sum_{i=1}^{N_{trn}} z_i c_i k(x_i, x) + b.
\]

In our experiments, we empirically set \( \gamma \) as half of the average within-class variance, i.e.

\[
\gamma = \frac{1}{2N_{lm}} \sum_l \sum_{1 \leq j \leq N_{tl}} ||x_j - m_l||^2,
\]

where \( N_{tl} \) is the number of samples in class \( l \) and \( m_l \) is the mean vector of data in class \( l \).

To extract new features of the data, we use the trained dichotomizers to classify each training sample \( x_i \) and transform the output discriminant function to approximate posterior probability using the sigmoid function

\[
Z_{ij} = \sigma(a_{fj}(x_i) + b_j) = \frac{1}{1 + \exp[-(a_{fj}(x_i) + b_j)]}, \quad j = 1, \ldots, p.
\]

The parameters \( (a_j, b_j) \), \( j = 1, \ldots, p \), can be estimated on a validation data set using a regularized cross-entropy criterion [44]. In practice, however, we found that for SVMs, the simple choice of \( (a_j = 1, b_j = 0) \) gives fairly high performance. On testing the training samples in the original space, \( Z = [z_1, \ldots, z_{N_{trn}}]^T \) is the data matrix of new representation for training meta classifiers in the \( p \)-dimensional space. More precisely, for linear feature extraction (ECOC-LFE), we use Eqs. (2) and (9) to calculate \( p \) confidence outputs of the linear SVMs as a new representation, while for nonlinear feature extraction (ECOC-NLFE), we use Eqs. (7) and (9) to calculate the new features based on the nonlinear SVMs.

4.3. Recoding (meta learning)

Although many approaches have been developed to learn the ECOC matrix from data, as far as we know, all the existing ECOC methods only define one codeword for each class. As discussed earlier, if the structure of the data is relatively complex, one codeword for each class may not guarantee satisfactory decoding. We address this problem by introducing a meta learning procedure.

We formulate the recoding as a learning vector quantization (LVQ) problem. Specifically, we learn \( m \) prototypes for each class in the new feature space of dichotomizers outputs and take the learned prototypes as codewords of that class. We choose the generalized learning vector quantization (GLVQ) [32] as the meta learner, which has demonstrated superiority in nearest-prototype-based classification. The GLVQ algorithm is outlined in the following.

Let \( \xi_1 \) be the nearest prototype vector that belongs to the same class of \( z \), \( \xi_2 \) be the nearest prototype vector that belongs to a different class from \( z \). The relative distance difference \( \varphi(z) \) is defined as

\[
\varphi(z) = \frac{d_1 - d_2}{d_1 + d_2},
\]

where \( d_1 = ||z - \xi_1||^2 \) and \( d_2 = ||z - \xi_2||^2 \) are the squared Euclidean distance of \( z \) from \( \xi_1 \) and \( \xi_2 \), respectively. The GLVQ learns the prototypes on a labeled data set by minimizing an empirical loss

\[
\min E = \sum_{i=1}^{N_{lm}} \sigma(\varphi(z_i)),
\]

where \( \sigma(\cdot) \) is the sigmoid function. To minimize \( E \), \( \xi_1 \) and \( \xi_2 \) are updated by stochastic gradient descent

\[
\begin{align*}
\xi_1 &\leftarrow \xi_1 + 4\tau \frac{\partial \sigma}{\partial \varphi} \frac{d_2}{(d_1 + d_2)^2} (z - \xi_1), \\
\xi_2 &\leftarrow \xi_2 - 4\tau \frac{\partial \sigma}{\partial \varphi} \frac{d_2}{(d_1 + d_2)^2} (z - \xi_2),
\end{align*}
\]

where \( \tau \) is the step size and \( \partial \sigma / \partial \varphi = \sigma(1 - \sigma) \) is the gradient of \( \sigma \) w.r.t. \( \varphi \). To speed up learning, the modified updating rules were suggested [32]

\[
\begin{align*}
\xi_1 &\leftarrow \xi_1 + 4\tau \frac{\partial \sigma}{\partial \varphi} \frac{d_2}{d_1 + d_2} (z - \xi_1), \\
\xi_2 &\leftarrow \xi_2 - 4\tau \frac{\partial \sigma}{\partial \varphi} \frac{d_2}{d_1 + d_2} (z - \xi_2).
\end{align*}
\]

To recode the codewords for each class, we train the GLVQ meta classifier on the new features of the data and take the learned prototypes of each class as new codewords. In our experiments, we empirically set the number of prototypes for each class as \( \lceil N_{trn}/(100 \times C_l) \rceil \times 3 \), where \( C_l \) is the most close integer that is larger than \( t \). We use the \( k \)-means clustering algorithm to cluster the training samples of each class and use the cluster centers as initial prototypes of GLVQ.

4.4. Decoding (classification)

In the decoding step, we feed all the test data into the learned dichotomizers and obtain the \( p \)-dimensional representation of them. The new feature vectors are then tested using the learned meta classifier. In the case of prototype based meta classifier learned by GLVQ, specifically, the test sample (in the new feature space) is assigned to the class of nearest prototype. This classification rule can be viewed as the extension of distance-based decoding with multiple codewords.

In summary, we show the process of our ECOC-EFE framework in Algorithm 1.

**Algorithm 1.** Process of ECOC-EFE.

1. **Input:**
   \[ X = \{ (x_1, c_1), (x_2, c_2), \ldots, (x_{N_{trn}}, c_{N_{trn}}) \}; \]
2. **Output:**
   \[ Y = \{ y_1, y_2, \ldots, y_{N_{lm}} \}; \]
3. **Steps:**
4. **Coding:**
5. **Decoding result;**
6. **Steps:**
7. **Coding:**
8. **Decoding result;**
9. **Steps:**
10. **Coding:**
11. **Decoding result;**
12. **Steps:**
13. **Coding:**
14. **Decoding result;**
15. **Steps:**
16: **Decoding:**
17: (1) Test each test sample using the dichotomizers and obtain the new feature representation;
18: (2) Decode via the meta classifier.

5. Experiments

To evaluate the classification performance of the proposed ECOC-EFE (including ECOC-LFE and ECOC-NLFE), we conducted extensive experiments on 16 multi-class data sets from the UCI machine learning repository. We compared ECOC-EFE with classic feature extraction methods and start-of-the-art ECOC methods. As below, we present the used data sets, parameter settings, statistical comparison methods, detailed results and discussions, respectively.

5.1. Data sets

Following [30], we test the compared methods on 16 multi-class data sets from the UCI machine learning repository. These data sets have various numbers of classes, attributes and samples. The details of them are summarized in Table 1. Particularly, for data sets that include class with less than 10 samples, we perturb the samples by adding standard Gaussian noise and append them to that class until its sample size exceeds 10. For each data set, we rescale all the features to be within \( \frac{1}{2}/C_0 \) to that class until its sample size exceeds 10. For each data set, the classification results and running times are reported based on average over stratified 10-fold cross-validation.

5.2. Parameter settings

As mentioned earlier, we use ECOCONE as the coding strategy of ECOC-EFE, which is initialized using DECOD. Thus, the dimensionality of the new feature space is around \( C \) based on the extension of the initial configuration, where \( C \) is the number of classes. Following [30], in the implementation of ECOC-EFE and all the compared ECOC methods, the penalty factor for SVMs is set as \( \lambda = 1 \). The kernel width for RBF kernel is set to half of the average within-class variance as shown in Eq. (8). We did not attempt to optimize the parameters of SVMs, but it is fair to compare different ECOC methods using the same parameter setting for the dichotomizers. For the GLVQ meta learner, the number of prototypes of each class is empirically set as \( \lfloor N_{\text{trn}}/(100 \times C) \rfloor \times 3 \), where \( \lfloor \cdot \rfloor \) is the most close integer that is larger than \( \tau \), and \( N_{\text{trn}} \) is the number of training samples.

Besides, we use the ECOC library [47] to implement the ECOC algorithms. The parameters for coding and decoding just follow the classification results. For the GLVQ meta learner, the number of prototypes of each class is set as \( \lfloor N_{\text{trn}}/(100 \times C) \rfloor \times 3 \), where \( \lfloor \cdot \rfloor \) is the most close integer that is larger than \( \tau \), and \( N_{\text{trn}} \) is the number of training samples.

5.3. Statistical comparison

To statistically compare the classification results, we conduct the Wilcoxon signed-ranks test [48] for the comparison between two methods, whilst the Friedman test [49] and the Nemenyi test [50] for comparing multiple methods, as suggested by [51]. The details of these tests can be found in [51].

5.4. Visualization of features

In this experiment, we show the 2D embedding learned by ECOC-NLFE, PCA and LDA. Fig. 1 plots the results of four data sets: Balance, Iris, Thyroid and Wine, which all have three classes. The training and test data are randomly partitioned with ratio around 9:1. As we can see from Fig. 1, the overlapping between classes in the 2D embeddings obtained by PCA and LDA is generally worse than that obtained by ECOC-NLFE. As a result, the discrimination between classes yielded by ECOC-NLFE is much better than that yielded by PCA and LDA. The classification results shown in Table 5 confirm this observation.

5.5. Comparison with state-of-the-art ECOC methods

In this experiment, we compare the proposed ECOC-EFE method with some state-of-the-art ECOC methods. The compared coding strategies include one-versus-one (OnevsOne), one-versus-all (OnevsAll), DECOD and ECOCONE, where ECOCONE is initialized by the DECOD method. For all the coding strategies, we use the linear loss-weighted (LLW) decoding method, which was shown superior for multi-class classification [30]. For ECOC-NLFE and the compared ECOC methods, SVMs with RBF kernel are used as dichotomizers. Similarly, for ECOC-LFE and the corresponding ECOC methods, linear SVMs are used as dichotomizers. The classification results obtained by ECOC-NLFE and ECOC based methods are shown in Table 2, while those obtained by ECOC-LFE and the ECOC methods are shown in Table 3.

To evaluate the significance of the performance differences, we conduct the Friedman and the Nemenyi test [51] with a confidence value 0.05 on the results presented in Table 2. The statistical comparison results show that ECOC-NLFE is significantly better than the one-versus-all coding design, and at least comparable with the other ones. We then conduct the same statistical tests on the results presented in Table 3 and observe the tendency as that for ECOC-NLFE. Although the mean rank of ECOC-LFE is a little lower than the one-versus-one method, the performance difference of them is not statistically significant. It is noteworthy that the ECOC-EFE uses much less dichotomizers than the one-versus-one strategy despite their comparable performance.

Table 4 shows the average decoding time of ECOC-NLFE (not including the recoding time) and the compared ECOC methods. It is easy to see, for all the data sets, that ECOC-NLFE is the fastest among the compared methods. For some data sets, such as Dermatology and Ecoli, the decoding of ECOC-NLFE is nearly 50 times faster than that of the one-versus-one method, and nearly 20 times faster than that of the DECOD method. For ECOC-LFE and the compared ECOC methods, we have the same observation that the decoding of ECOC-LFE is dramatically faster than that of the compared ECOC methods. For simplicity, the results are not listed here.

<table>
<thead>
<tr>
<th>Problem</th>
<th>( t ) of ( T )</th>
<th>( t ) of ( A )</th>
<th>( t ) of ( C )</th>
<th>Problem</th>
<th>( t ) of ( T )</th>
<th>( t ) of ( A )</th>
<th>( t ) of ( C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balance</td>
<td>625</td>
<td>4</td>
<td>3</td>
<td>Satimage</td>
<td>6435</td>
<td>36</td>
<td>7</td>
</tr>
<tr>
<td>Dermatology</td>
<td>366</td>
<td>34</td>
<td>6</td>
<td>Segmentation</td>
<td>2310</td>
<td>19</td>
<td>7</td>
</tr>
<tr>
<td>Ecoli</td>
<td>336</td>
<td>8</td>
<td>8</td>
<td>Shuttle</td>
<td>14,500</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>Glass</td>
<td>214</td>
<td>9</td>
<td>7</td>
<td>Thyroid</td>
<td>215</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Iris</td>
<td>150</td>
<td>4</td>
<td>3</td>
<td>Vehicle</td>
<td>840</td>
<td>18</td>
<td>4</td>
</tr>
<tr>
<td>Letter</td>
<td>20,000</td>
<td>16</td>
<td>25</td>
<td>Vowel</td>
<td>990</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>Optdigits</td>
<td>5620</td>
<td>64</td>
<td>10</td>
<td>Wine</td>
<td>178</td>
<td>13</td>
<td>3</td>
</tr>
<tr>
<td>Pendigits</td>
<td>10,992</td>
<td>16</td>
<td>10</td>
<td>Yeast</td>
<td>1484</td>
<td>8</td>
<td>10</td>
</tr>
</tbody>
</table>
From the results shown in Tables 3–5, we know that our method, either ECOC-NLFE or ECOC-LFE, is promising for the multi-class classification problems. Compared with the state-of-the-art ECOC methods, it performs significantly better than or at least comparable with them. More importantly, the decoding speed of our method is much faster than that of the other competitive ECOC methods.

From Tables 2 and 3, we find that the decoding results of one-versus-all strategy on the Letter and the Vowel data set are evidently worse than the other compared methods. This is mainly because the linear loss-weighted (LLW) decoding strategy was developed for the ternary coding design methods. Actually, for the one-versus-all coding strategy, straightforward classification without decoding (i.e., classify to the class of maximum dichotomizer output) performs very well. This “no decoding” classification rule should be compared with the LLW decoding, for clarifying the performance of the one-versus-all coding design.

Table 5 shows the classification results of one-versus-all with both linear and nonlinear dichotomizers. We conduct the Wilcoxon signed-ranks test [51] with confidence value 0.05 on the results shown in the second column and the third column of Table 5. The test results show that the performance difference
The statistical tests on the results in the fourth column and the fifth column of Table 5 (nonlinear dichotomizers) show that the number of prototypes per class learned by GLVQ. We conduct the Wilcoxon signed-ranks test [51] with confidence value 0.05 on the results. The statistical tests show that ECOC-NLFE performs significantly better than PCA, KPCA and LDA. Note that the dimensionality of learned features by LDA and GDA is at most $C-1$ (if $C < D$). However, ECOC-NLFE does not have this limitation since its dimensionality equals the number of dichotomizers, and has the potential of extracting more features and yielding higher classification performance.

### 5.7. Effect of the meta learner

In this experiment, we compare the classification performance of ECOC-NLFE using different types of meta learner. We first consider different prototype classifiers for meta learning: one prototype per class learned by GLVQ, nearest class mean, multiple prototypes learned by $k$-means clustering. For $k$-means clustering and the proposed ECOC-NLFE, the number of prototypes per class is $\lceil\frac{N_{trn}}{100 \times C}\rceil \times 3$. The classification results are shown in Table 7, where “1-prototype” denotes one prototype per class learned by GLVQ. We conduct the Wilcoxon signed-ranks test.
Vowel, where the classes are randomly grouped to form a three-
with multiple prototype per class and that with only one proto-
of each class distribute in multiple modes, GLVQ with only one prototype per class performs fairly well for these problems. However, if the data single mode. Consequently, GLVQ with only one prototype per class because for most of the data sets, the data of each class are in comparably on these multi-class learning problems. This is types per class and that with only one prototype per class perform Table 7 shows that the GLVQ meta learner with multiple proto-

Table 6
Comparison with classic feature extraction methods on the UCI data sets.

<table>
<thead>
<tr>
<th>Data sets</th>
<th>PCA</th>
<th>KPCA</th>
<th>LDA</th>
<th>GDA</th>
<th>ECOC-NLFE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balance</td>
<td>0.4700</td>
<td>0.4531</td>
<td>0.8236</td>
<td>0.7707</td>
<td>0.8931</td>
</tr>
<tr>
<td>Dermatology</td>
<td>0.9188</td>
<td>0.9188</td>
<td>0.9779</td>
<td>0.9636</td>
<td>0.9711</td>
</tr>
<tr>
<td>Ecoli</td>
<td>0.8284</td>
<td>0.8294</td>
<td>0.8588</td>
<td>0.8304</td>
<td>0.8127</td>
</tr>
<tr>
<td>Glass</td>
<td>0.6012</td>
<td>0.5716</td>
<td>0.5716</td>
<td>0.6084</td>
<td>0.6108</td>
</tr>
<tr>
<td>Iris</td>
<td>0.9000</td>
<td>0.9467</td>
<td>0.9800</td>
<td>0.9800</td>
<td>0.9733</td>
</tr>
<tr>
<td>Letter</td>
<td>0.9288</td>
<td>0.9043</td>
<td>0.9349</td>
<td>0.9392</td>
<td>0.9483</td>
</tr>
<tr>
<td>OptDigits</td>
<td>0.9615</td>
<td>0.9584</td>
<td>0.9143</td>
<td>0.9893</td>
<td>0.9869</td>
</tr>
<tr>
<td>Pendigits</td>
<td>0.9753</td>
<td>0.9743</td>
<td>0.9654</td>
<td>0.9904</td>
<td>0.9941</td>
</tr>
<tr>
<td>Satimage</td>
<td>0.8636</td>
<td>0.8683</td>
<td>0.8589</td>
<td>0.8788</td>
<td>0.8857</td>
</tr>
<tr>
<td>Segmentation</td>
<td>0.9208</td>
<td>0.9203</td>
<td>0.6398</td>
<td>0.9511</td>
<td>0.9550</td>
</tr>
<tr>
<td>Shuttle</td>
<td>0.9866</td>
<td>0.9888</td>
<td>0.9862</td>
<td>0.9930</td>
<td>0.9969</td>
</tr>
<tr>
<td>Thyroid</td>
<td>0.9656</td>
<td>0.9626</td>
<td>0.9522</td>
<td>0.9570</td>
<td>0.9742</td>
</tr>
<tr>
<td>Vehicle</td>
<td>0.5040</td>
<td>0.4944</td>
<td>0.7827</td>
<td>0.8340</td>
<td>0.7735</td>
</tr>
<tr>
<td>Vowel</td>
<td>0.5717</td>
<td>0.5727</td>
<td>0.5434</td>
<td>0.7343</td>
<td>0.7606</td>
</tr>
<tr>
<td>Wine</td>
<td>0.9750</td>
<td>0.9750</td>
<td>0.9875</td>
<td>0.9658</td>
<td>0.9813</td>
</tr>
<tr>
<td>Yeast</td>
<td>0.4999</td>
<td>0.5244</td>
<td>0.5033</td>
<td>0.5405</td>
<td>0.5545</td>
</tr>
<tr>
<td>Mean rank</td>
<td>3.8125</td>
<td>3.7813</td>
<td>3.3750</td>
<td>2.3438</td>
<td>1.6875</td>
</tr>
</tbody>
</table>

Table 7
Classification results of ECOC-NLFE using different meta learners.

<table>
<thead>
<tr>
<th>Data sets</th>
<th>1-prototype</th>
<th>Class mean</th>
<th>ECOC-NLFE</th>
<th>k-means</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balance</td>
<td>0.8820</td>
<td>0.8070</td>
<td>0.8931</td>
<td>0.8617</td>
</tr>
<tr>
<td>Dermatology</td>
<td>0.9711</td>
<td>0.9711</td>
<td>0.9711</td>
<td>0.9750</td>
</tr>
<tr>
<td>Ecoli</td>
<td>0.8529</td>
<td>0.8353</td>
<td>0.8127</td>
<td>0.7892</td>
</tr>
<tr>
<td>Glass</td>
<td>0.6510</td>
<td>0.5831</td>
<td>0.6108</td>
<td>0.4984</td>
</tr>
<tr>
<td>Iris</td>
<td>0.9533</td>
<td>0.9533</td>
<td>0.9733</td>
<td>0.9600</td>
</tr>
<tr>
<td>Letter</td>
<td>0.8190</td>
<td>0.8097</td>
<td>0.9483</td>
<td>0.9448</td>
</tr>
<tr>
<td>OptDigits</td>
<td>0.9816</td>
<td>0.9801</td>
<td>0.9869</td>
<td>0.9886</td>
</tr>
<tr>
<td>Pendigits</td>
<td>0.9872</td>
<td>0.9871</td>
<td>0.9941</td>
<td>0.9934</td>
</tr>
<tr>
<td>Satimage</td>
<td>0.8773</td>
<td>0.8757</td>
<td>0.8837</td>
<td>0.8612</td>
</tr>
<tr>
<td>Segmentation</td>
<td>0.9442</td>
<td>0.9472</td>
<td>0.9550</td>
<td>0.9442</td>
</tr>
<tr>
<td>Shuttle</td>
<td>0.9880</td>
<td>0.9757</td>
<td>0.9969</td>
<td>0.9914</td>
</tr>
<tr>
<td>Thyroid</td>
<td>0.9665</td>
<td>0.9742</td>
<td>0.9742</td>
<td>0.9608</td>
</tr>
<tr>
<td>Vehicle</td>
<td>0.7537</td>
<td>0.7400</td>
<td>0.7735</td>
<td>0.7482</td>
</tr>
<tr>
<td>Vowel</td>
<td>0.7212</td>
<td>0.7313</td>
<td>0.7606</td>
<td>0.7424</td>
</tr>
<tr>
<td>Wine</td>
<td>0.9875</td>
<td>0.9875</td>
<td>0.9813</td>
<td>0.9813</td>
</tr>
<tr>
<td>Yeast</td>
<td>0.5987</td>
<td>0.5664</td>
<td>0.5545</td>
<td>0.4497</td>
</tr>
</tbody>
</table>

Confidence value 0.05 on the classification accuracies obtained by 10-fold cross-validation on each data set. The statistical tests show that the GLVQ meta learner with multiple prototypes per class performs significantly better than that with only one prototype per class in all three data sets. This indicates the necessity of using the GLVQ meta learning with multiple prototypes for each class.

Besides the GLVQ, many classifiers can also be applied as meta learners, such as the one nearest neighbor (1-NN) classifier and SVMs combined with one versus one strategy or one versus all strategy. To evaluate the performance of different meta learners, we compare ECOC-NLFE with GLVQ as meta learner against that with one nearest neighbor meta learner (1-NN meta), that with combined SVMs based on the one-versus-all strategy (with no decoding as introduced in Section 5.5, SVM meta1v1A), that with combined SVMs based on the one-versus-one strategy (using majority voting for the decoding, SVM meta1v1), 1-NN classifier in the original space (1-NN original), and GLVQ in the original space (GLVQ original). Both the SVMs based meta learners are implemented using linear kernels, which perform sufficiently well in the new feature space spanned by the dichotomizers outputs. The classification results are shown in Table 8. We conduct the Wilcoxon signed-ranks test [51] with confidence value 0.05 to compare the results in the second column (1-NN original) and GLVQ in the original space (1-NN original), and GLVQ in the original space (1-NN meta) against that with one nearest neighbor meta learner (1-NN meta), that with combined SVMs based on the one-versus-all strategy (with no decoding).
before the cross-validation. Except the difference of meta learner ECOCONE, we learn the ECOC matrix using all the training data the same length of codewords for different partitions of data by ing data to extract new features for meta learning. To guarantee (Stacking-FE).

performance of ECOC-NLFE with stacking based feature extraction sification by combining dichotomizers. We hence compare the tive in combining multiple classifiers, including multi-class clas-

meta learning by cross-validation, has been demonstrated effec-
other hand, the stacking strategy, that generates training data for formed data from the training samples for training the dichot-

5.8. Comparison with stacking based feature extraction

SVMs based meta learners and that with GLVQ meta learner are comparable. In contrast to combination of SVMs based meta learners, the GLVQ meta learner has better interpretation because the learned prototypes can be seen as codewords. However, it is hard to consider the support vectors learned by the base classifiers, SVMs, as codewords. All these results indicate the effective-
ness of GLVQ as the meta learner of ECOC-NLFE.

5.8. Comparison with stacking based feature extraction

Our method trains the meta learner by ‘reusing’ the transformed data from the training samples for training the dichot-
omizers. This raises a question of possible overfitting. On the other hand, the stacking strategy, that generates training data for meta learning by cross-validation, has been demonstrated effective in combining multiple classifiers, including multi-class class-
ification by combining dichotomizers. We hence compare the performance of ECOC-NLFE with stacking based feature extraction (Stacking-FE).

For Stacking-FE, we use 10-fold cross-validation on the train-
ing data to extract new features for meta learning. To guarantee the same length of codewords for different partitions of data by ECOCON, we learn the ECOC matrix using all the training data before the cross-validation. Except the difference of meta learner training data generation (reuse versus stacking), the settings of dichotomizers and meta learner (GLVQ) are the same for ECOC-NLFE and Stacking-FE. We implement Stacking-FE with four ECOC coding strategies: one-versus-one (S-OnevsOne), one-versus-all (S-OnevsAll), DEOC (S-DECOC) and ECOCON (S-ECO-
CON). The classification results are shown in Table 9.

We conduct the Friedman and the Nemenyi test [51] with confidence value 0.05 on the results in Table 9. The statistical test shows that ECOC-NLFE performs significantly better than S-DECOC and S-ECOCONE, and meanwhile, it is at least comparable with S-OnevsOne and S-OnevsAll. This indicates that there is no evident overfitting caused by ECOC-NLFE (reusing).

The inferior performance of Stacking-FE compared to ECOC-
NLFE can be explained as follows. First, although ECOCON can learn the ECOC matrix by extending an initial configuration, it cannot be used in the cross-validation procedure of Stacking-FE, since the learned codewords may have different lengths and render the corresponding partitions of data having different dimensionalities. To overcome this problem, we learn the ECOC matrix of Stacking-FE on all the training data. This loses the optimality of cross-validation for Stacking-FE. In contrast, ECOC-
NLFE is more flexible, which learns the new features of the data directly from the ECOC matrix and the trained dichotomizers. Second, when the sizes of the data in different classes are dramatically unbalanced, such as the Shuttle data set, Stacking-
FE may fail to learn effective new features of the data. Generally speaking, this problem can be overcome by carefully tuning the dichotomizers, but it is a time consuming. On the contrary, the ECOC-NLFE turns out to be less sensitive to data imbalance, as it generates the meta learner training data directly from all the training samples.

6. Conclusion

In this paper, we propose an ECOC based ensemble feature extraction (ECOC-EFE) method to take advantage of the coding matrix learning ability and discrimination ability of the dichot-
omizers. Both linear features and nonlinear features can be extracted easily. Using the probabilistic outputs of the dichotomizers as new features of the data, we use a meta classifier to perform multi-class classification. Specifically, we use the general-
ized learning vector quantization (GLVQ) for meta learning. The learned prototype of a class can be viewed as a codeword of that class. Extensive experiments on 16 data sets from the UCI machine learning repository demonstrated the effectiveness, efficiency, robustness and flexibility of our method. Particularly, the ECOC-EFE performs superiorly or comparably with the state-of-the-art ECOC methods and feature extraction methods. In the future, we would like to exploit the fast ECOC-EFE algorithm and its application to large-scale problems using new models of dichotomizers [53].

Acknowledgments

This work is supported by the National Natural Science Foundation of China (NSFC) Grant 60825301 and the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant XDA06030300). The authors would like to thank Sergio Escalera for the private communication and suggestions to this work.

References

G. Zhong, C.-L. Liu / Pattern Recognition 46 (2013) 1091–1100

Guoqiang Zhong received the BS degree in Mathematics and Applied Mathematics from Hebei Normal University, Shijiazhuang, China, in 2004 and the MS degree in Operational Research and Cybernetics from Beijing University of Technology, Beijing, China, in 2007. He got his Ph.D degree in Pattern Recognition and Intelligent System from the Institute of Automation of Chinese Academy of Sciences, Beijing, China, in 2011. Currently, he is a postdoctoral fellow at the Ecole de technologie supérieure, University of Quebec, Montreal, Canada. His research interests include pattern recognition and machine learning.

Cheng-Lin Liu is a Professor at the National Laboratory of Pattern Recognition (NLPR), Institute of Automation of Chinese Academy of Sciences, Beijing, China, and is now the deputy director of the laboratory. He received the B.S. degree in electronic engineering from Wuhan University, Wuhan, China, the M.E. degree in electronic engineering from Beijing Polytechnic University, Beijing, China, and the Ph.D. degree in pattern recognition and intelligent control from the Chinese Academy of Sciences, Beijing, China, in 1989, 1992 and 1995, respectively. He was a postdoctoral fellow at Korea Advanced Institute of Science and Technology (KAIST) and later at Tokyo University of Agriculture and Technology from March 1996 to March 1999. From 1999 to 2004, he was a research staff member and later a senior researcher at the Central Research Laboratory, Hitachi, Ltd., Tokyo, Japan. His research interests include pattern recognition, image processing, neural networks, machine learning, and especially the applications to character recognition and document analysis. He has published over 140 technical papers at prestigious international journals and conferences, and co-authored a book on character recognition.