Maximum Margin Tree Error Correcting Output Codes

Fa Zheng[1,2], Hui Xue[1,2(✉)], Xiaohong Chen[3], and Yunyun Wang[4]

[1] School of Computer Science and Engineering, Southeast University, Nanjing 210096, People's Republic of China
{faaronzheng,hxue}@seu.edu.cn
[2] Key Laboratory of Computer Network and Information Integration, Southeast University, Ministry of Education, Nanjing, People's Republic of China
[3] College of Science, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, People's Republic of China
lyandcxh@nuaa.edu.cn
[4] Department of Computer Science and Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210046, People's Republic of China
wangyunyun@njupt.edu.cn

Abstract. Encoding is one of the most important steps in Error Correcting Output Codes (ECOCs). Traditional encoding strategies are usually data-independent. Recently, some tree-form encoding algorithms are proposed which firstly utilize mutual information to estimate inter-class separability in order to create a hierarchical partition of the tree from top to down and then obtain a coding matrix. But such criterion is usually computed by a non-parametric method which would generally require vast samples and is more likely to lead to unstable results. In this paper, we present a novel encoding algorithm which uses the maximum margins between classes as the criterion and constructs a bottom-up binary tree based on the maximum margin. As a result, the corresponding coding matrix is more stable and discriminative for the following classification. Experimental results have shown that our algorithm performs much better than some state-of-the-art coding algorithms in ECOC.

Keywords: Multi-class classification · Maximum margin tree · Error Correcting Output Codes

1 Introduction

The multi-class classification problem has attracted a lot attentions in machine learning field. The traditional solutions tend to transform it into multiple binary problems. The corresponding strategies include decision tree, neural networks, and so on.

Error Correcting Output Codes (ECOCs) [1,2] is a widely-used method in these strategies, which was originally proposed by Dietterich and Bakiri [3].

R. Booth and M.-L. Zhang (Eds.): PRICAI 2016, LNAI 9810, pp. 681–691, 2016.
DOI: 10.1007/978-3-319-42911-3_57

It usually involves two parts: encoding and decoding. Encoding part generates a sequence of bits, i.e. a code word for each class. All code words form a coding matrix. Decoding part predicts class labels for unseen data through comparing their output code words with the code words of classes in the coding matrix depending on some specific strategies such as Hamming decoding (HD) [4] and Euclidean decoding [5]. In this paper, we will focus on encoding part.

The goal of encoding is to design a coding matrix M. Each row of M represents one class and each column of M is one binary problem (dichotomizer). For each class, encoding aims to create a corresponding code word where each bit is the prediction of the dichotomizer. Traditionally, the coding matrix is coded by $+1$ and -1. In Table 1(a), $+1$ means that the corresponding dichotomizer takes this class as a positive class and -1 otherwise. However, the length of the code words in this scenario is actually fixed. As a result, a limited number of dichotomizers can be used which would restrict the performance of ECOC to some extent. Allwein et al. [6] further presented a ternary coding matrix which allows some bits of coding matrix to be zero as Table 1(b). The symbol zero denotes that the corresponding class does not participate in the specific classification. Result from the zero symbols, the ternary coding matrix is more flexible and could have much longer code words than the binary one.

Consequently, the core task in encoding has boiled down to how to build such an appropriate coding matrix. The simplest strategy is one-versus-all (OVA) [4] which takes one class as a positive class and all the others as a negative class to build the binary coding matrix. One-versus-one (OVO) [5] forms a ternary coding matrix where each column only considers two classes to be positive and negative classes respectively and the rest are represented by zero symbols. Random codes [6] generate the coding matrix randomly, where the binary coding matrix is called dense random while the ternary one is termed as spare random. Though these traditional strategies are simple, they are all data-independent. As a result, they either perform poorly or get too long code words which would require more dichotomoizers with higher computational costs.

Recent proposed tree-form encoding algorithms [7–10] utilize some criterions to estimate inter-class separability so as to build a tree and obtains a data-dependent coding matrix. Discriminant ECOC (DECOC) [8] applies the sequential forward floating search (SFFS) to generate the tree from top to down

Table 1. Coding matrix for a 4-class problem

(a) Binary

	h_1	h_2	h_3	h_4
C_1	+1	-1	-1	-1
C_2	-1	+1	-1	-1
C_3	-1	-1	+1	-1
C_4	-1	-1	-1	+1

(b) Ternary

	h_1	h_2	h_3	h_4	h_5	h_6
C_1	+1	+1	+1	0	0	0
C_2	-1	0	0	+1	+1	0
C_3	0	-1	0	-1	0	+1
C_4	0	0	-1	0	-1	-1

through maximizing the mutual information (MI) [11] between classes heuristically. Then a ternary coding matrix is constructed according to the hierarchical partition of the tree. Based on DECOC, subclass ECOC (SECOC) [9] further uses a cluster method to create subclasses while the original classification problem is linearly non-separable. However, the MI criterion used in DECOC and SECOC is computed by a non-parametric method which generally requires a large number of samples and further leads to an unstable result. Hierarchical ECOC (HECOC) [12] utilizes support vector domain description (SVDD) [12] as the criterion to estimate inter-class separability, which is more stable than DECOC and SECOC. However, when building the tree, HECOC chooses two classes which have the smallest inter-class separability as a node. As a result, the base dichotomizers will face a relatively difficult binary classification problem which limits the performance of ECOC to some degree.

In this paper, we propose a novel encoding method termed as maximum margin tree ECOC (M^2ECOC). M^2ECOC estimates the maximal inter-class separability by the maximum margins between classes rather than the MI criterion. Consequently, the corresponding coding matrix is more stable and discriminative for the following classification. Concretely, M^2ECOC uses support vector machine (SVM) [13] to compute the maximum margins between classes and then obtains a maximum margin matrix. Depending on this matrix, M^2ECOC further generates a bottom-up binary tree based on choosing the maximal maximum margin. Finally, such maximum margin tree will be converted into a ternary coding matrix according to the hierarchical partition of the tree.

The paper is organized as follows. Section 2 introduces the tree-form encoding algorithms DECOC and SECOC. Section 3 introduces our M^2ECOC algorithm in detail. In Sect. 4, the compared experiments with some state-of-the-art encoding algorithms are shown. Finally, the last section concludes the paper.

2 Tree-Form Encoding Algorithms

2.1 Discriminant ECOC (DECOC)

DECOC [8] firstly applies SFFS to find the hierarchical partition of the tree and builds the tree from top to down. As Fig. 1(a) shown, DECOC separates the original class set $\{C_1, C_2, C_3\}$ into two partitions $\{C_1, C_3\}$ and $\{C_2\}$ until each partition has only one class. SFFS is one kind of suboptimal sequential search methods which dynamically changes the number of forward steps until the resulting subsets are better than the previously ones based on some criterions [8]. MI, which is an often-used metric to compute the relativity between two random variables in information theory, is selected to evaluate the discriminability of class sets in DECOC. MI is defined as follow:

$$I(\boldsymbol{x}, \boldsymbol{y}) = \int\int p(x, y) \log\left(\frac{p(x, y)}{p(x)p(y)}\right) dx\, dy \tag{1}$$

where $\boldsymbol{x}$ denotes the sample in the class sets and $\boldsymbol{y}$ denotes the class label. $p(\mathbf{x})$ and $p(\mathbf{y})$ are their probability density functions respectively. DECOC aims to

(a) DECOC (b) SECOC

Fig. 1. Illustration of the trees in DECOC and SECOC

maximize the MI value between the data in the class sets and the class label to maximize the discriminability of class sets. However, when computing the MI value, DECOC uses a non-parametric parzen estimation method which usually requires a large number of samples in order to reach a relatively better performance and is more likely to lead to unstable experimental results.

When the tree has been completely constructed, DECOC further fills a ternary coding matrix based on the tree. Particularly, the classes in the left partition are represented by $+1$ and the classes in the right partition are represented by -1. Meanwhile, classes which are not shown up in the hierarchical partition are represented by 0.

2.2 Subclass ECOC (SECOC)

On the basis of DECOC, SECOC [9] contributes to solving the linearly non-separable problems by dividing the original class into some new subclasses. SECOC also uses SFFS to find the hierarchical partition of the tree by maximizing the MI value. When the original partition is linearly non-separable, SECOC further uses the cluster method K-means to split it into simpler and smaller sub-partitions. Usually, the number of sub-partitions is set to 2 [9]. As Fig. 1(b) shown, SECOC splits the original linearly non-separable problem $\{C_1, C_3\}$ into two linearly separable problems $\{C_1, C_{3\_1}\}$ and $\{C_1, C_{3\_2}\}$ by dividing class C_3 into two new subclasses C_{3_1} and C_{3_2}. Therefore, if the original classification problem is linearly non-separable, SECOC can transform it into a linearly separable one through several times of decompositions.

3 Maximum Margin Tree ECOC (M²ECOC)

3.1 Maximum Margin

Margin, which is defined as the minimum distance between the decision boundary and samples, is one of the most famous concepts in SVM proposed by Vapnik [13].

Specifically, decision boundary which is also called decision hyperplane, is denoted as follows:

$$\boldsymbol{w}^T \Phi(\boldsymbol{x}) + b = 0 \tag{2}$$

where $\Phi(\boldsymbol{x})$ is a fixed feature-space transformation function, $\boldsymbol{w}$ is a weight vector and b is the bias. The functional margin can be formulated as:

$$\hat{r} = \min_i \{ y_i (\boldsymbol{w}^T \Phi(\boldsymbol{x}_i) + b) \} \quad i = 1, 2, ..., N \tag{3}$$

where $y_i \in \boldsymbol{y}$ denotes the corresponding class label and N is the number of samples. However, the functional margin does not have the scaling invariance. So, we further get the geometric margin by normalizing (3):

$$\tilde{r} = \min_i \{ y_i (\frac{\boldsymbol{w}^T}{||\boldsymbol{w}||} \Phi(\boldsymbol{x}_i) + \frac{b}{||\boldsymbol{w}||}) \} \quad i = 1, 2, ..., N \tag{4}$$

According to (3) and (4), we can easily obtain the relationship with the functional margin and the geometric margin as follows:

$$\tilde{r} = \frac{\hat{r}}{||\boldsymbol{w}||} \tag{5}$$

Let $\hat{r}$ equal to 1. The maximum margin can be optimized by solving the following problem:

$$\begin{aligned} \max_{\boldsymbol{w}, b} \quad & \frac{1}{||\boldsymbol{w}||} \\ s.t. \quad & y_i (\boldsymbol{w}^T \Phi(\boldsymbol{x}_i) + b) - 1 \geqslant 0, \quad i = 1, 2, ..., N \end{aligned} \tag{6}$$

It is obvious that the maximization of $||\boldsymbol{w}||^{-1}$ is equivalent to the minimization of $||\boldsymbol{w}||$. So we can transform (6) into the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{w}, b} \quad & \frac{1}{2} ||\boldsymbol{w}||^2 + C \sum_{i=1}^{N} \xi_i \\ s.t. \quad & y_i (\boldsymbol{w}^T \Phi(\boldsymbol{x}_i) + b) \geqslant 1 - \xi_i, \quad i = 1, 2, ..., N \\ & \xi_i \geqslant 0, \quad i = 1, 2, ..., N \end{aligned} \tag{7}$$

where the parameter $\boldsymbol{\xi}$ is the slack variable and C is used to balance $\boldsymbol{\xi}$ and the margin. (7) can be further changed into a dual problem using Lagrange multipliers with kernel functions:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ s.t. \quad & \sum_{i=1}^{N} \alpha_i y_i = 0, \quad i = 1, 2, ..., N \\ & 0 \leqslant \alpha_i \leqslant C, \quad i = 1, 2, ..., N \end{aligned} \tag{8}$$

Through (8), we can solve the $\boldsymbol{\alpha}$. Consequently, the maximum margin can be finally computed as follows [14]:

$$margin = \frac{1}{||\boldsymbol{w}||} \tag{9}$$

where the vector $\boldsymbol{w}$ is determined by

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \tag{10}$$

3.2 Maximum Margin Matrix

Given a k-class classification problem, we can compute the maximum margin between each pair of classes according to (9). Then all maximum margins can be combined as a maximum margin matrix:

$$\begin{bmatrix} 0 & m_{12} & \ldots & m_{1(k-1)} & m_{1k} \\ m_{21} & 0 & \ldots & m_{2(k-1)} & m_{2k} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ m_{(k-1)1} & m_{(k-1)2} & \ldots & 0 & m_{(k-1)k} \\ m_{k1} & m_{k2} & \ldots & m_{k(k-1)} & 0 \end{bmatrix}$$

where m_{ij} is the maximum margin between the ith and jth classes. Obviously, this matrix is symmetric. So we just compute the values of upper triangular matrix elements. As can be seen, the bigger the value of m_{ij} is, the larger the maximum margin between these two classes would be. Furthermore, a larger maximum margin means that the corresponding two classes are more well-separated. Consequently, the maximum margin actually gives us a natural criterion to evaluate the discriminability between classes. In M^{2}ECOC, we will directly use the maximum margin to build the tree.

3.3 Maximum Margin Tree

Traditional tree algorithms such as DECOC and SECOC usually built the tree from top to down as Fig. 1 shown. In fact, such strategy emphasizes more on the discriminability between internal nodes, but ignores the discriminability between leaf nodes. For example, in Fig. 1(a), DECOC firstly separates the original class set $\{C_1, C_2, C_3\}$ into two partitions $\{C_1, C_3\}$ and $\{C_2\}$ and then divides the internal node $\{C_1, C_3\}$ into two leaf nodes $\{C_1\}$ and $\{C_3\}$. As a result, DECOC can guarantee that the internal partition between the internal node $\{C_1, C_3\}$ and $\{C_2\}$ has good discriminability and the corresponding dichotomizer in the internal node can achieve satisfactory performance. However, DECOC can not guarantee that the two leaf nodes $\{C_1\}$ and $\{C_3\}$ also have similarly good discriminability, which leads to the performance of the corresponding dichotomizer uncontrollable.

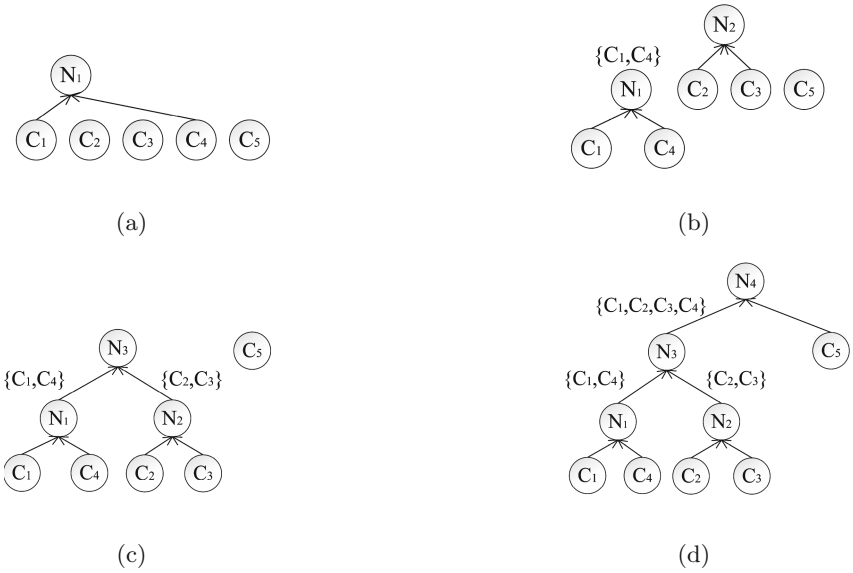

Fig. 2. Construction of a bottom-up maximum margin tree in M^{2}ECOC

In M^{2}ECOC, we adopt a bottom-up strategy [10] to construct the maximum margin tree. In order to illustrate the strategy more clearly, we will take a five-class classification problem as an example in Fig. 2. Concretely, we firstly regard each class as a subclass and use (9) to compute the maximum margin matrix. According to this matrix, $\{C_1\}$ and $\{C_4\}$ have the maximal maximum margin. So we combine them as a new subclass (Fig. 2(a)) but still keep their original classes labels. Then the new subclass and the rest classes generate a new four-class classification problem. Repeating the above process, we take $\{C_2\}$ and $\{C_3\}$ between which have the maximal maximum margin as another new subclass (Fig. 2(b)). Consequently, the subclasses $\{C_1, C_4\}$, $\{C_2, C_3\}$ and $\{C_5\}$ boil down to a new three-class classification problem. Following the same steps, the two subclasses $\{C_1, C_4\}$ and $\{C_2, C_3\}$ are integrated as a new subclass $\{C_1, C_2, C_3, C_4\}$ (Fig. 2(c)). Particularly, when computing the margin between subclasses $\{C_1, C_4\}$ and $\{C_2, C_3\}$, the classes in the same subclass will be considered as one class temporarily. Finally the expected maximum margin tree can be obtained as Fig. 2(d).

After the optimal hierarchical partition of the maximum margin tree has been finished, we obtain the coding matrix M as follows:

$$M(r,l) = \begin{cases} 1 & C_r \in P_l^{left} \\ 0 & C_r \notin P_l \\ -1 & C_r \in P_l^{right} \end{cases} \tag{11}$$

where $M(r,l)$ denotes the element lying in the rth row and the lth column in the coding matrix and the C_r denotes the rth class. P_l^{left} and P_l^{right} are

Table 2. Coding matrix of the example in Fig. 2

	h_1	h_2	h_3	h_4
C_1	+1	0	+1	+1
C_2	0	+1	−1	+1
C_3	0	−1	−1	+1
C_4	−1	0	+1	+1
C_5	0	0	0	−1

the left and right partition of the lth partition respectively (Regardless of the root node). Table 2 lists the corresponding coding matrix of the above example following (11).

4 Experimental Results

In this section, we compare M^{2}ECOC with some state-of-the-art coding algorithms like OVA [4], OVO [5], dense random [6], sparse random [6], DECOC [8][1], SECOC [9] and HECOC [10] to validate the superiority of our approach.

Ten multi-class datasets from common-used UCI datasets [15] are used in the experiments, that is, Wine (178,13,3), Lenses (24,4,3), Glass (214,9,6), Balance (625,4,3), Cmc (1473,9,3), Ecoli (332,6,6), Iris (150,4,3), Tea (151,5,3), Thyriod (215,5,3), Vehicle (846,18,4), where the numbers of samples, dimension and classes are listed in the bracket. We randomly split each dataset into two non-overlapping training and testing set. The training set contains almost seventy percent of the samples and the rest samples are composed as the testing set. The whole process is repeated ten times. The average accuracies are also reported.

Moreover, in dense and sparse random algorithms, all random matrices are selected from a set of 10000 randomly generated matrices where $P(1) = P(-1) = 0.5$ for the dense random matrix as well as $P(1) = P(-1) = 0.25$ and $P(0) = 0.5$ for the sparse random matrix [6]. In SECOC, the parameter set $\Theta = \{\Theta_{size}, \Theta_{perf}, \Theta_{impr}\}$ is fixed to $\Theta_{size} = \frac{|J|}{50}$, $\Theta_{perf} = 0$ and $\Theta_{impr} = 0.95$ according to [9]. The regularization parameter C and the width σ in radial basis function kernel in HECOC and M^{2}ECOC are selected from the interval $\{2^{-6}, 2^{-5}, ..., 2^5, 2^6\}$ by cross-validation.

The decoding strategy HD is used to evaluate the performance of different coding algorithms. Two base classifiers Nearest Mean Classifier (NMC) and SVM with radial basis function kernel are applied as the dichotomizers, where the regularization parameter C is set to 1 [9]. Moreover, the width σ in the kernel is also selected from the same interval in HECOC and M^{2}ECOC.

[1] We download the DECOC code from http://jmlr.csail.mit.edu/papers/v11/escalera10a.html which was provided by Sergio Escalera, Oriol Pujol, Petia Radeva in 2010.

The classification results on the ten datasets are reported in Tables 3 and 4. From the tables, we can see that M^{2}ECOC can reach better or comparable performance than compared algorithms on most datasets. Especially, the accuracies of M^{2}ECOC exceed the other algorithms' accuracies beyond 3 % on the Glass and Vehicle sets with NMC in Table 3. In Table 4, Its accuracy even excels the other accuracies nearly 12 % on the Lenses set with SVM. Furthermore, we also list the average accuracies and standard deviations on all the datasets in the bottom of the tables. It obviously can be seen that M^{2}ECOC possesses the best performance compared with the other algorithms, which further indicates the superiority of M^{2}ECOC. On the contrary, DECOC and SECOC perform much poorly with SVM in some datasets. For example, their accuracies are even 10 % lower than the other algorithms' accuracies on the Lenses, Balance, Iris sets

Table 3. Classification results (mean ± std) of NMC and HD on ten datasets (•/◦ indicates that our algorithm is significantly better or worse than other algorithms based on the t-test at 95 % significance level)

	OVO	OVA	Dense	Sparse	DECOC	SECOC	HECOC	M^{2}ECOC
Wine	97.55±1.79	94.91±2.82•	93.40±5.85•	94.34±4.71•	97.36±3.11	97.36±3.11	95.09±3.47•	**98.30**±1.88
Lenses	77.50±7.91	78.75±8.44	72.50±9.86	70.00±10.50•	78.75±8.44	78.75±8.44	71.25±11.90	**80.00**±8.74
Glass	46.62±6.07•	23.23±5.69•	40.31±9.22•	47.85±6.62	42.15±11.40•	49.08±6.97	42.31±10.00•	**52.77**±6.03
Balance	73.69±4.63•	**88.07**±1.90◦	69.79±20.10	75.72±14.60	80.32±3.72	81.60±5.27	81.60±3.16	81.17±2.77
Cmc	46.74±2.10	45.86±1.58	46.02±2.20	45.81±2.35	46.29±1.19	46.31±1.18	45.48±2.38	**46.97**±1.60
Ecoli	**84.90**±2.88◦	70.50±1.90•	77.50±5.46•	78.50±4.14•	74.20±8.27•	77.90±4.07•	71.90±13.40•	81.90±2.60
Iris	86.00±5.55	82.89±2.97	76.67±9.03•	74.44±10.10•	79.78±8.35	85.11±3.93	**86.22**±5.52	86.00±5.55
Tea	**56.52**±5.12	52.83±6.49	54.13±6.60	51.96±4.64	53.26±5.64	55.87±5.80	54.13±6.44	56.09±5.21
Thyriod	92.81±1.98	94.06±2.42	92.50±3.52	91.41±2.97•	92.97±3.40	93.44±3.59	94.06±2.42	**94.53**±2.68
Vehicle	46.10±3.51	43.62±1.52•	40.20±4.22•	38.03±5.40•	40.04±3.37•	43.94±5.22•	40.67±3.40•	**49.25**±4.86
Average	70.84±4.15	67.47±3.57	66.30±7.61	66.81±6.60	68.51±5.69	70.94±4.76	68.27±6.21	**72.70**±4.19
win/tie/loss	2/7/1	4/5/1	5/5/0	6/4/0	3/7/0	2/8/0	4/6/0	/

Table 4. Classification results (mean ± std) of SVM and HD on ten datasets (•/◦ indicates that our algorithm is significantly better or worse than other algorithms based on the t-test at 95 % significance level)

	OVO	OVA	Dense	Sparse	DECOC	SECOC	HECOC	M^{2}ECOC
Wine	97.92±1.39•	98.49±1.73	98.11±1.26•	98.11±1.78•	96.23±1.99•	97.36±1.82•	98.30±1.65•	**99.62**±0.80
Lenses	61.25±3.95•	62.50±0.00•	63.75±3.95•	61.25±3.95•	56.25±8.84•	58.75±6.04•	62.50±0.00•	**75.00**±10.20
Glass	68.31±2.43◦	53.23±5.04•	**68.46**±3.34◦	67.23±2.62◦	62.31±9.84	64.15±9.97◦	61.85±5.88	62.15±3.78
Balance	90.27±0.97	89.36±0.89•	91.50±2.85	**91.55**±3.07	76.84±1.99•	77.17±2.03•	89.79±1.02•	90.86±0.64
Cmc	54.43±1.46	50.54±1.53•	48.10±1.79•	47.83±1.58•	51.97±1.46•	52.04±1.42•	**54.52**±1.19	54.00±0.98
Ecoli	86.40±3.03	82.40±2.17•	86.50±2.07	86.80±2.04	80.30±15.10	83.10±6.59	70.30±15.50•	**86.90**±2.59
Iris	**95.56**±3.63	93.33±2.34	94.00±3.32	95.11±3.11	72.89±4.42•	74.67±2.81•	95.11±3.60	94.89±1.83
Tea	51.09±4.38	51.09±6.58	49.57±4.44•	51.30±4.25	44.35±5.99•	47.83±5.02•	51.30±3.58•	**55.43**±4.94
Thyriod	**96.09**±1.11	94.84±1.66	95.00±2.31	95.31±1.47	94.37±1.32•	94.37±1.32•	95.78±1.06	**96.09**±1.11
Vehicle	74.92±1.76	66.50±2.48•	72.28±5.20	71.93±2.87•	72.76±2.37•	72.76±2.37•	74.53±1.93	**75.08**±1.76
Average	77.62±2.41	74.23±2.44	76.73±3.05	76.64±2.67	70.83±5.33	72.22±3.94	75.40±3.54	**79.00**±2.86
win/tie/loss	2/7/1	6/4/0	4/5/1	4/5/1	8/2/0	8/1/1	5/5/0	/

in Table 4. The reason lies more on they are more sensitive to different base classifier and their using a non-parametric estimation to compute the MI value, which indeed requires numerous training data to achieve acceptable results.

In order to further statistically measure the significance of performance difference, the pairwise t-tests [16] at 95 % significance level are conducted between the algorithms. Specifically, whenever M^{2}ECOC achieves significantly better/worse performance than the compared algorithms on most datasets, a win/loss is counted and a marker $\bullet/\circ$ are shown. Otherwise, a tie is counted and no marker is given. The resulting win/tie/loss counts for M^{2}ECOC against the compared algorithms are provided in the last line of Tables 3 and 4. As the tables shown, M^{2}ECOC can achieve statistically better or comparable performance on most datasets, which just accords with our conclusion.

5 Conclusion

In this paper, we present a novel encoding algorithm M^{2}ECOC for ECOC. Different from the existing tree-form encoding algorithms, M^{2}ECOC directly utilizes the maximum margin which actually is a natural criterion to evaluate the discriminability between classes to get the optimal hierarchical partition of the tree. Specifically, M^{2}ECOC regards each class as a subclass and computes the maximum margin matrix. According to this matrix, the classes with the maximal maximum margin are selected to combine as a new subclass. Then the new subclass and the rest classes generate a new multi-class classification problem. Repeating the same steps until all classes in one subclass. M^{2}ECOC constructs the maximum margin tree in a bottom-up manner and the corresponding coding matrix can be obtained easily by the tree. The experimental results on several UCI datasets have shown that M^{2}ECOC is superior to some state-of-the-art ECOC encoding algorithms, which further validates that the maximum margin is indeed an effective criterion for building the tree in ECOC.

Acknowledgements. This work was supported by National Natural Science Foundation of China (Grant Nos. 61375057, 61300165 and 61403193) and Natural Science Foundation of Jiangsu Province of China (Grant No. BK20131298). Furthermore, the work was also supported by Collaborative Innovation Center of Wireless Communications Technology.

References

1. Japkowicz, N., Barnabe-Lortie, V., Horvatic, S., et al.: Multi-class learning using data driven ECOC with deep search and re-balancing. In: IEEE International Conference on DSAA, pp. 1–10 (2015)
2. Liu, M., Zhang, D., Chen, S., et al.: Joint binary classifier learning for ECOC-based multi-class classification. IEEE Trans. Pattern Anal. Mach. Intell. **1**, 0162–8828 (2015)
3. Dietterich, T.G., Bakiri, G.: Solving multiclass learning problems via error-correcting output codes. J. Artif. Intell. Res. **2**, 263–286 (1995)

4. Nilsson, N.J.: Learning Machines: Foundations of Trainable Pattern-Classifying Systems. McGraw-Hill, New York (1965)
5. Hastie, T., Tibshirani, R.: Classification by pairwise coupling. Ann. Stat. **26**(2), 451–471 (1998)
6. Allwein, E.L., Schapire, R.E., Singer, Y.: Reducing multiclass to binary: a unifying approach for margin classifiers. J. Mach. Learn. Res. **1**, 113–141 (2001)
7. Escalera, S., Pujol, O., Radeva, P.: Boosted landmarks of contextual descriptors and forest-ECOC: a novel framework to detect and classify objects in cluttered scenes. Pattern Recogn. Lett. **28**(13), 1759–1768 (2007)
8. Pujol, O., Radeva, P., Vitrial, J.: Discriminate ECOC: a heuristic method for application dependent design of error correcting output codes. IEEE Trans. Pattern Anal. Mach. Intell. **28**(6), 1001–1007 (2006)
9. Escalera, S., Tax, D.M.J., Pujol, O., et al.: Subclass problem-dependent design for error-correcting output codes. IEEE. Trans. Pattern Anal. Mach. Intell. **30**(6), 1041–1054 (2008)
10. Lei, L., Wang, X., Luo, X., et al.: Hierarchical error-correcting output codes based on SVDD. J. Syst. Eng. Electron. **37**(8), 1916–1921 (2015). (In Chinese)
11. Principe, J.C., Xu, D., Fisher, J.: Information theoretic learning. In: Unsupervised Adaptive Filtering, vol. 1, pp. 265–319 (2000)
12. Tax, D.M.J., Duin, R.P.W.: Support vector domain description. Pattern Recogn. Lett. **20**(11), 1191–1199 (1999)
13. Cortes, C., Vapnik, V., Guyaon, I.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
14. Lu, M., Huo, J., Chen, C.L.P., et al.: Multi-stage decision tree based on inter-class and inner-class margin of SVM. In: IEEE International Conference on SYST, pp. 1875–1880 (2009)
15. Asuncion, A., Newman, D.: UCI Machine Learning Repository (2007)
16. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley, New York (2004)