A Comparison of Random Forest with ECOC-Based Classifiers

R.S.Smith[1], M.Bober[2] and T.Windeatt[1]

[1]Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford, Surrey GU2 7XH, UK
[2]Mitsubishi Electric R&D Centre Europe B.V, 20 Frederick Sanger Road, The Surrey Research Park, Guildford, Surrey GU2 7YD, UK
{Raymond.Smith, T.Windeatt}@surrey.ac.uk.

Abstract We compare experimentally the performance of three approaches to ensemble-based classification on general multi-class datasets. These are the methods of random forest, error-correcting output codes (ECOC) and ECOC enhanced by the use of bootstrapping and class-separability weighting (ECOC-BW). These experiments suggest that ECOC-BW yields better generalisation performance than either random forest or unmodified ECOC. A bias-variance analysis indicates that ECOC benefits from reduced bias, when compared to random forest, and that ECOC-BW benefits additionally from reduced variance. One disadvantage of ECOC-based algorithms, however, when compared with random forest, is that they impose a greater computational demand leading to longer training times.

1 Introduction

Two of the most popular approaches to constructing multiple classifier systems (MCS) to solve multi-class classification problems are random forest [1] and error-correcting output codes (ECOC) [2,3]. In this paper we present the result of an experimental comparison of these two methods when applied to a selection of real-world datasets taken from the UCI repository [4]. We also consider an enhanced version of ECOC, referred to as ECOC-BW, in which bootstrapping[1] is applied when constructing base-classifier training sets and weighting is applied to base-classifier decisions. Previous work has shown these enhancements to be beneficial [5].

The random forest algorithm was introduced by Breiman in 2001 [1]. A number of variants of random forest have been proposed but here we focus on the method that is often cited as a reference in the literature, known as Forest-RI. This consists of building an ensemble of unpruned decision tree classifiers whose

[1] Bootstrapping is a technique whereby new training sets are constructed from a given training set by repeated sampling with replacement. Each new training set (referred to as a bootstrap replicate) has, on average, 63% of the patterns in the original set but with some patterns repeated so as to form a set of the same size.

classification decisions are combined by a voting procedure. Each decision tree is randomised in two ways: firstly, the training set is modified by constructing a bootstrap replicate and secondly, at each node of the tree, the search for the best split is limited to a subset of features that is randomly selected (without replacement) from the full set of features. Forest-RI thus aims to achieve good classification performance by combining the principles of bagging and random feature selection.

In the ECOC approach, first described by Dietterich in 1991 [6], a multi-class problem is decomposed into a series of 2-class problems, or dichotomies, and a separate base classifier trained to solve each one. These 2-class problems are constructed by repeatedly partitioning the set of target classes into pairs of super-classes so that, given a large enough number of such partitions, each target class can be uniquely represented as the intersection of the super-classes to which it belongs. The classification of a previously unseen pattern is then performed by applying each of the base classifiers so as to make decisions about the super-class membership of the pattern. Redundancy can be introduced into the scheme by using more than the minimum number of base classifiers and this allows errors made by some of the classifiers to be corrected by the ensemble as a whole.

The operation of the ECOC algorithm can be broken down into two distinct stages - the coding stage and the decoding stage. The coding stage consists of applying the base classifiers to the input pattern $\mathbf{x}$ so as to construct vector of base classifier outputs $\mathbf{s}(\mathbf{x})$ and the decoding stage consists of applying some decoding rule to this vector so as to make an estimate of the class label that should be assigned to the input pattern. A commonly used decoding method is to base the classification decision on the minimum distance between $\mathbf{s}(\mathbf{x})$ and the vector of target outputs for each of the classes, using a distance metric such as Hamming or L^1. This, however, treats all base classifiers as equal, and takes no account of variations in their reliability. In the ECOC-BW variant of ECOC we assign different weights to each base classifier and target class combination so as to obtain improved ensemble accuracy. The weighting algorithm is referred to as class-separability weighting (CSEP) because the weights are computed in such a way that they measure the ability of a base classifier to distinguish between examples belonging to and not belonging to a given class [7].

Although, unlike random forest, bootstrapping is not a standard feature of the ECOC algorithm, we have shown in [5] that it can be beneficial, particularly when combined with the CSEP weighting scheme. For this reason, in ECOC-BW we apply bootstrapping to the training set when each base classifier is trained. The effect of bootstrapping is to increase the desirable property of diversity [8] among the base classifiers in the ensemble. By this is meant that the errors made by component classifiers should, as far as possible, be uncorrelated so that the error correcting properties of the ensemble can have maximum effect. A further potential benefit of bootstrapping is that each base classifier is trained on only a subset of the available training data and this leaves the remaining data, known as

the out-of-bootstrap (OOB) set, to be used for other purposes such as parameter tuning. Note, however, that the OOB set is unique to each base classifier.

When considering the errors made by statistical pattern classifiers it is useful to group them under three headings. Firstly there is the unavoidable error, known as *Bayes error*, which is caused by noise in the process that generates the patterns. A second source of error is *variance*; this is caused by the sensitivity of a learning algorithm to the chance details of a particular training set and causes slightly different training sets to produce classifiers that give different predictions for some patterns. Thirdly there are errors caused by *bias* in a learning algorithm[2]; here the problem is that the classifier is unable, for whatever reason, to adequately model the class decision boundaries in the pattern feature space. When training a classifier there is often a tradeoff between bias and variance [9] so that a high value of one implies a low value of the other.

In this paper we use the concepts of bias and variance to investigate the reasons for the differences in the accuracy achieved by different classification methods.

The ideas of bias, variance and noise originally emerged from regression theory. In this context they can be defined in such a way that the squared loss can be expressed as the sum of noise, bias (squared) and variance. The goal of generalising these concepts to classification problems, using a 0-1 or other loss function, has proved elusive and several alternative definitions have been proposed (see [10] for a summary). In fact it is shown in [10] that, for a general loss function, these concepts cannot be defined in such a way as to possess all desirable properties simultaneously. For example the different sources of error may not be additive, or it may be possible for variance to take negative values. In this study we adopt the Kohavi-Wolpert definitions [11]. These have the advantage that bias and variance are non-negative and additive. A disadvantage, however, is that no explicit allowance is made for Bayes error and it is, in effect, incorporated into the bias term.

The remainder of this paper is structured as follows. The technique of CSEP weighting is described in detail in section 2. Here we also derive an alternative probabilistic interpretation of the method. Section 3 then describes the experimental results obtained from each of the three classification methods: random forest, ECOC and ECOC-BW. Finally, section 4 summarises the conclusions to be drawn from this work.

2 ECOC Weighted Decoding

The ECOC method consists of repeatedly partitioning the full set of N classes Ω into L super-class pairs. The choice of partitions is represented by an $N \times L$ binary *coding matrix* $\mathbf{Z}$. The rows $\mathbf{Z}_i$ are unique *codewords* that are associated with the individual target classes ω_i and the columns $\mathbf{Z}^j$ represent the different super-class partitions. Denoting the jth super-class pair by S^j and $\overline{\mathrm{S}^j}$, element

[2] Bias is actually measured as the quantity bias^2 .

Z_{ij} of the coding matrix is set to 1 or 0 depending on whether class ω_i has been put into S^j or its complement[3]. A separate base classifier is trained to solve each of these 2-class problems.

Given an input pattern vector $\mathbf{x}$ whose true class $y(\mathbf{x}) \in \Omega$ is unknown, let the soft output from the jth base classifier be $s_j(\mathbf{x}) \in [0,1]$. The set of outputs from all the classifiers can be assembled into a vector $\mathbf{s}(\mathbf{x}) = [s_1(\mathbf{x}), \ldots, s_L(\mathbf{x})]^{\mathrm{T}} \in [0,1]^L$ called the *output code* for $\mathbf{x}$. Instead of working with the soft base classifier outputs, we may also first harden them, by rounding to 0 or 1, to obtain the binary vector $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), \ldots, h_L(\mathbf{x})]^{\mathrm{T}} \in \{0,1\}^L$. The principle of the ECOC technique is to obtain an estimate $\hat{y}(\mathbf{x}) \in \Omega$ of the class label for $\mathbf{x}$ from a knowledge of the output code $\mathbf{s}(\mathbf{x})$ or $\mathbf{h}(\mathbf{x})$.

In its general form, a *weighted* decoding procedure makes use of an $N \times L$ weights matrix $\mathbf{W}$ that assigns a different weight to each target class and base classifier combination. The class decision, based on the L^1 metric, is made as follows:

$$\hat{y}(\mathbf{x}) = \arg\min_{\omega_i} \sum_{\mathrm{j}=1}^{\mathrm{L}} \mathbf{W}_{\mathrm{ij}} \left| \mathrm{s_j}(\mathbf{x}) - \mathbf{Z}_{\mathrm{ij}} \right|, \tag{1}$$

where it is assumed that the rows of $\mathbf{W}$ are normalised so that $\sum_{j=1}^{L} \mathbf{W}_{ij} = 1$ for $i = 1 \ldots N$. If the base classifier outputs $s_j(\mathbf{x})$ in Eqn. 1 are replaced by hardened values $h_j(\mathbf{x})$ then this describes the weighted Hamming decoding procedure.

The values of $\mathbf{W}$ may be chosen in different ways. For example, if $\mathbf{W}_{ij} = \frac{1}{L}$ for all i, j then the decoding procedure of Eqn. 1 is equivalent to the standard unweighted L^1 or Hamming decoding scheme. In this paper we make use of the class separability measure [7,5] to obtain weight values that express the ability of each base classifier to distinguish members of a given class from those of any other class.

In order to describe the class-separability weighting scheme, the concept of a correctness function must first be introduced: given a pattern $\mathbf{x}$ which is known to belong to class ω_i, the correctness function for the j'th base classifier takes the value 1 if the base classifier makes a correct prediction for $\mathbf{x}$ and 0 otherwise:

$$C_j(\mathbf{x}) = \begin{cases} 1 & \text{if } h_j(\mathbf{x}) = \mathbf{Z}_{ij} \\ 0 & \text{if } h_j(\mathbf{x}) \neq \mathbf{Z}_{ij} \end{cases}. \tag{2}$$

We also consider the complement of the correctness function $\overline{C}_j(\mathbf{x}) = 1 - C_j(\mathbf{x})$ which takes the value 1 for an incorrect prediction and 0 otherwise.

For a given class index i and base classifier index j, the class-separability weight measures the difference between the positive and negative correlations of base classifier predictions, ignoring any base classifiers for which this difference

[3] Alternatively, the values +1 and -1 are often used.

is negative:

$$\mathbf{W}_{ij} = \max\left\{0, \frac{1}{K_i}\left[\sum_{\substack{\mathbf{p}\in\omega_i \\ \mathbf{q}\notin\omega_i}} C_j(\mathbf{p})\, C_j(\mathbf{q}) - \sum_{\substack{\mathbf{p}\in\omega_i \\ \mathbf{q}\notin\omega_i}} \overline{C}_j(\mathbf{p})\, \overline{C}_j(\mathbf{q})\right]\right\}, \quad (3)$$

where patterns $\mathbf{p}$ and $\mathbf{q}$ are taken from a fixed training set T and K_i is a normalisation constant that ensures that the i'th row of $\mathbf{W}$ sums to 1. The algorithm for computing $\mathbf{W}$ is summarised in fig. 1.

Inputs: matrix of training patterns $\mathbf{T} \in \mathbb{R}^{P\times M}$, binary coding matrix $\mathbf{Z} \in \{0,1\}^{N\times L}$, trained ECOC coding function $E : \mathbb{R}^M \mapsto [0,1]^L$.
Outputs: weight matrix $\mathbf{W} \in [0,1]^{N\times L}$ where $\sum_{j=1}^{L} \mathbf{W}_{ij} = 1$, for $i = 1 \ldots N$.
Apply E to each row of $\mathbf{T}$ and round to give prediction matrix $\mathbf{H} \in \{0,1\}^{P\times L}$.
Initialise $\mathbf{W}$ to $\mathbf{0}$.
for $c = 1$ to N
  for $i =$ indices of training patterns belonging to class c
    for $j =$ indices of training patterns *not* belonging to class c
      let d be the true class of the pattern $\mathbf{T}_j$.
      for k $= 1$ to L
        if $\mathbf{H}_{ik} = \mathbf{Z}_{ck}$ and $\mathbf{H}_{\mathrm{jk}} = \mathbf{Z}_{\mathrm{dk}}$, add 1 to $\mathbf{W}_{ck}$
          as the predictions for both patterns $\mathbf{T}_i$ and $\mathbf{T}_j$ are correct.
        if $\mathbf{H}_{ik} \neq \mathbf{Z}_{ck}$ and $\mathbf{H}_{\mathrm{jk}} \neq \mathbf{Z}_{\mathrm{dk}}$, subtract 1 from$\mathbf{W}_{ck}$
          as the predictions for both patterns $\mathbf{T}_i$ and $\mathbf{T}_j$ are incorrect.
      end
    end
  end
end
Reset all negative entries in $\mathbf{W}$ to 0.
Normalise $\mathbf{W}$ so that each row sums to 1.

Figure 1. Pseudo-code for computing the class-separability weight matrix for ECOC.

The weights matrix $\mathbf{W}_{ij}$ of Eqn. 3 was derived from a consideration of the spectral properties of the Boolean functions that map base classifier outputs to the ensemble decisions. In this interpretation base classifiers are weighted by their ability to distinguish the members of a given class from patterns which do not belong to that class. An alternative interpretation may also be given in terms of base classifier accuracy probabilities. Let

$$P_{ij} = \frac{1}{M_i}\sum_{\mathbf{x}\in\omega_i} c_j(\mathbf{x}),\; Q_{ij} = \frac{1}{(M-M_i)}\sum_{\mathbf{x}'\notin\omega_i} c_j(\mathbf{x}') \quad (4)$$

where M is the total number of training patterns and M_i is the number of belonging to class ω_i. Then P_{ij} and Q_{ij} respectively represent estimates of the probability that the jth base classifier makes correct decisions for patterns belonging to and not belonging to class ω_i. By substituting $M_i P_{ij}$ and $(M - M_i) Q_{ij}$ for $\sum_{\mathbf{x} \in \omega_i} c_j(\mathbf{x})$ and $\sum_{\mathbf{x}' \notin \omega_i} c_j(\mathbf{x}')$ in Eqn. 3 and making use of the fact that $\overline{c}_j(\mathbf{x}) = 1 - c_j(\mathbf{x})$, it can be easily shown that an alternative definition of the CSEP weights is given by:

$$\mathbf{W}_{ij} = \max\left\{0, \frac{1}{K_i'}\left[P_{ij} + Q_{ij} - 1\right]\right\}, \tag{5}$$

where $K_i' = K_i / M_i (M - M_i)$ is a modified normalisation constant.

From Eqn. 5 it can be seen that CSEP weighting rewards those base classifiers that have a high true detection rate and a low false detection rate for class ω_i. Any base classifier that cannot outperform random guessing, where $P_{ij} = Q_{ij} = 0.5$, will be zero weighted under this algorithm.

3 Experiments

In this section we present the results of performing classification experiments on 11 multi-class datasets obtained from the publicly available UCI repository [4]. The characteristics of these datasets in terms of size, number of classes and number of features are given in table 1 All experiments were based on a 20/80 training/test set split and each run used a different randomly chosen stratified training set. These training sets were first normalised to have zero mean and unit variance.

Table 1. Experimental datasets showing the number of patterns, classes, continuous and categorical features.

Dataset	Num. Patterns	Num. Classes	Cont. Features	Cat. Features
dermatology	366	6	1	33
ecoli	336	8	5	2
glass	214	6	9	0
iris	150	3	4	0
segment	2310	7	19	0
soybean	683	19	0	35
thyroid	7200	3	6	15
vehicle	846	4	18	0
vowel	990	11	10	1
waveform	5000	3	40	0
yeast	1484	10	7	1

For each dataset, ECOC ensembles of 200 base classifiers were constructed. Each base classifier consisted of a multi-layer perceptron (MLP) neural network

with one hidden layer. The Levenberg-Marquardt algorithm was used for base classifier training as this has been shown to converge more rapidly than back-propagation. Base classifier complexity was adjusted by varying the hidden node counts and training epochs. Each such combination was repeated 10 times and the lowest mean test error was obtained. Random variations between each run were introduced by generating a different random code matrix and by randomly setting the initial MLP weights. The code matrices were constructed in such a way as to place an approximately equal number of target classes in the super-classes S^j and $\overline{\mathrm{S}^j}$.

The ECOC experiments were repeated using ECOC-BW (i.e. with CSEP weighting and bootstrapping being applied). Each base classifier was trained on a separate bootstrap replicate drawn from the full training set for that run. The CSEP weight matrix was computed from the full training set each time so its value was determined in part by patterns (the OOB set) that were not used in the training of the base classifier.

The random forest experiments were conducted in a similar way to ECOC except that it was found necessary to repeat each experiment 100 times in order to obtain stable results. The number of decision trees in each forest was varied up to 400 and the optimal number required to minimise test error was obtained. The number of random features selected at each node was chosen using Breiman's heuristic $\log_2 F + 1$ where F is the total number of features available[4]. This has been shown to yield near optimal results [12].

The outcome of these experiments on individual datasets is shown graphically in Fig. 2. This shows a bar chart of the lowest test error attained by the three classification methods. Also shown is the average test error taken over all datasets.

Inspection of Fig. 2 shows that no single classification algorithm gave the best results on all datasets. Indeed, random forest gave the lowest generalisation error on *glass* and *soybean*, ECOC gave the lowest error on *segment* and *vowel* whilst ECOC-BW was optimal on *dermatology*, *ecoli*, *iris*, *thyroid*, *vehicle*, *waveform* and *yeast*.

Comparing the different algorithms, it can be seen that ECOC gave a lower error than random forest on 7/11 datasets and also had a lower average error taken over all datasets. ECOC-BW yielded lower error than random forest on 9/11 datasets and also beat standard ECOC on 9/11 datasets. ECOC-BW also had the lowest mean error over all datasets. The evidence from these experiments is then that the ECOC-BW algorithm tends to give the best generalisation performance out of the three methods. There is also evidence that standard ECOC tends to perform a little better random forest but the advantage is not so consistent.

One further consideration that is worth taking into account when comparing these classification methods is that of computational overheads. The decision tree base classsifiers used by random forest are of a more lightweight nature that

[4] Another commonly used heuristic is $\sqrt{F}$. For these datasets, however, both formulae selected a very similar, and in many cases identical, number of features.

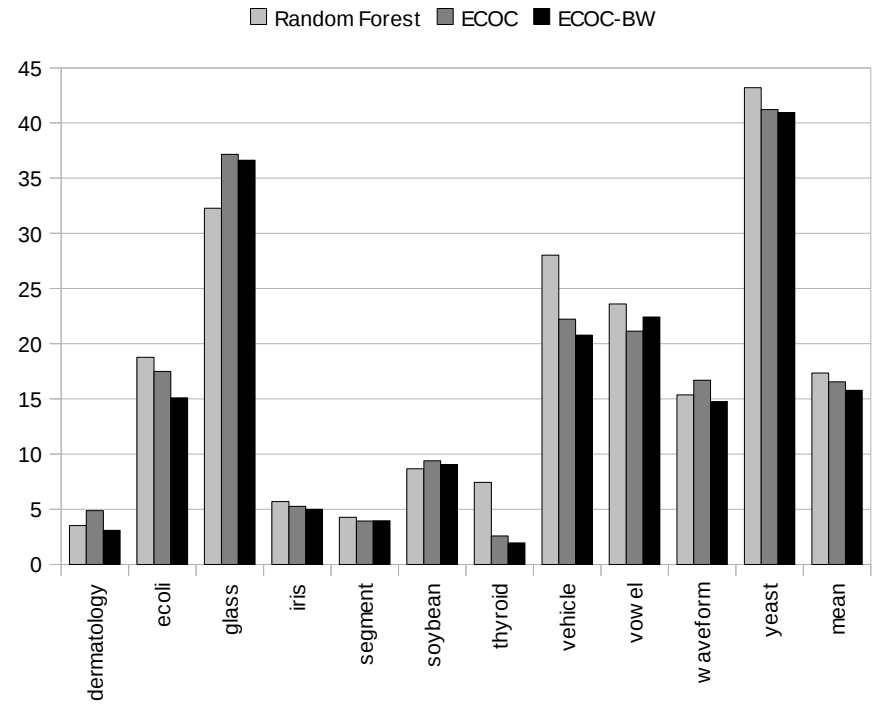

Figure 2. The lowest percentage ensemble test error attained by different classifiers on 11 datasets.

the MLP base classifiers that were used in ECOC classification and this was reflected in the elapsed times of the experiments which were, typically, about 15 times greater for the ECOC-based methods than for random forest.

It is interesting to look at the performance of these classifiers in terms of a bias-variance decomposition of the error. Table 2 shows a breakdown of the error incurred by each ensemble type when averaged over all datasets.

Inspection of table 2 suggests that when ECOC is compared with random forest, the variances of the two algorithms are the same and the slightly greater accuracy of ECOC may be attributed to a lower level of bias. It seems likely that lower bias of ECOC is due to the fact that the MLP base classifiers themselves will tend to have lower bias due to the fact that they are able to model non-linear decision boundaries between classes. By contrast, random forest uses decision trees as base classifiers are thus constrained to model decision boundaries as segments of hyperplanes that run parallel to the feature-space axes.

When ECOC-BW is used, variance is also reduced, leading to a further reduction in classification error. This is consistent with previous work [13] which has demonstrated that the use of CSEP weighting and bootstrapping tends to

make the ECOC ensemble less prone to over-fitting the data so that classifier decisions become less sensitive to variations in the training set.

Table 2. A comparison of percentage bias, variance and total error incurred by different classifiers. The values are averaged over 11 datasets.

	$Bias^2$	Variance	Total Error
Random Forest	9.8	7.5	17.4
ECOC	9.1	7.5	16.5
ECOC-BW	9.1	6.7	15.8

4 Discussion and Conclusions

In this paper we have compared experimentally the generalisation performance of three types of ensemble classifier on general multi-class datasets. The classifier types were random forest, ECOC and ECOC-BW in which ECOC is enhanced by the application of class-separability weighting and bootstrapping. The evidence from this set of experiments is that, although each classifier type can be optimal on some datasets, in general ECOC-BW tends to yield better accuracy than either random forest or ECOC. There is evidence that accuracy of ECOC is slightly better than that of random forest but the advantage cannot be so consistently observed as for ECOC-BW.

A breakdown of the error into bias and variance components reveals that standard ECOC has similar variance properties to random forest but benefits from slightly lower bias. It is suggested that this is due to the lower bias of the MLP base classifiers that were used with ECOC, when compared to the decision tree base classifiers of the random forest algorithm. For ECOC-BW the variance is also lower than for random forest and this leads to a further reduction in overall classification error.

Although ECOC-BW tends to yield greater classification accuracy, it is worth noting that random forest has an advantage over the ECOC-based algorithms in the sense that it has substantially reduced computational requirements.

5 Acknowledgements

This work was supported by EPSRC grant E061664/1. The authors would also like to thank the providers of the PRTools [14] and Weka [15] software.

References

1. Breiman L. Random Forests. *Journal of Machine Learning*, Vol. 45, No. 1, pp. 5-32, Springer, October 2001.

2. Dietterich TG, Bakiri G. Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research* 2: 263-286, 1995.
3. James G. Majority Vote Classifiers: Theory and Applications. PhD Dissertation, Stanford University, 1998.
4. Merz CJ, Murphy PM. UCI Repository of Machine Learning Databases, 1998. http://www.ics.uci.edu/~mlearn/MLRepository.html.
5. Smith RS, Windeatt T. Class-Separability Weighting and Bootstrapping in Error Correcting Output Code Ensembles. *Proc. 9th Int. Conf. on Multiple Classifier Systems*, LNCS 5997, pp. 185-194, 2010.
6. Dietterich TG, Bakiri G. Error-correcting output codes: A general method for improving multiclass inductive learning programs. *Proceedings of the Ninth National Conference on Artificial Intelligence* (AAAI-91) (pp. 572-577). Anaheim, CA: AAAI Press, 1991.
7. Windeatt T. Accuracy/ Diversity and Ensemble Classifier Design, *IEEE Trans Neural Networks*, 17(4), July, 2006.
8. Brown G, Wyatt J, Harris R, Yao X. Diversity Creation Methods: A Survey and Categorisation. Journal of Information Fusion, 6(1), 2005.
9. Geman S, Bienenstock E. Neural networks and the bias / variance dilemma. Neural Computation, 4:1-58, 1992.
10. James G. Variance and Bias for General Loss Functions. *Machine Learning*, 51 (2), 115-135, 2003.
11. Kohavi R, Wolpert D. Bias plus variance decomposition for zero-one loss functions. *Proc. 13th International Conference on Machine Learning*, pp. 275-283, 1996.
12. Bernard S, Heutte L, Adam S. Influence of Hyperparameters on Random Forest Accuracy. *Proc. 8th Int. Conf. on Multiple Classifier Systems*, LNCS 5519, pp. 171-180, 2009.
13. Smith RS, Windeatt T. A Bias-Variance Analysis of Bootstrapped Class-Separability Weighting for Error-Correcting Output Code Ensembles. *Proc. 20th Int. Conf. on Pattern Recognition (ICPR)*, pp. 61-64, 2010.
14. Duin RPW, Juszczak P, Paclik P, Pekalska E, D. de Ridder, Tax DMJ, Verzakov S. PRTools 4.1, A Matlab Toolbox for Pattern Recognition, Delft University of Technology, 2007.
15. Witten IH, Frank E. Data Mining: Practical machine learning tools and techniques. 2nd edition Morgan Kaufmann, San Francisco. 2005.