Rojas I., Joya G., Catala A. (eds) (2017). *Advances in Computational Intelligence: 14th International Work-Conference on Artificial Neural Networks, IWANN 2017 Cadiz, Spain, June 14–16, 2017, Proceedings*, Part I (Lecture Notes in Computer Science, 10305) Springer, 339-350.

DOI:https://doi.org/10.1007/978-3-319-59153-7_30

Pre-emphasizing Binarized Ensembles to Improve Classification Performance

Lorena Álvarez-Pérez(✉), Anas Ahachad, and Aníbal R. Figueiras-Vidal
GAMMA-L+/Department Signal Theory and Communications,
University Carlos III of Madrid, Av. de la Universidad 30,
28911 Leganés, Madrid, Spain
{lalvarez,anas,arfv}@tsc.uc3m.es

Abstract. Machine ensembles are learning architectures that offer high expressive capacities and, consequently, remarkable performances. This is due to their high number of trainable parameters.

In this paper, we explore and discuss whether binarization techniques are effective to improve standard diversification methods and if a simple additional trick, consisting in weighting the training examples, allows to obtain better results. Experimental results, for three selected classi-fication problems, show that binarization permits that standard direct diversification methods (bagging, in particular) achieve better results, obtaining even more significant performance improvements when pre-emphasizing the training samples. Some research avenues that this find-ing opens are mentioned in the conclusions.

Keywords: Classification · Multi-Layer Perceptron · Ensemble classi-fiers · Bagging · ECOC

1 Introduction

Although the expressive power –the capacity of a machine to establish general input-output correspondences– of one-hidden layer Multi-Layer Perceptrons (MLPs) is theoretically unbounded if an appropriate number of hidden units is included [1,2], the limited number of labeled training examples that are available in practice reduces it.

A way of overcoming this limitation is to build ensembles of MLPs –or other Learning Machines– by diversifying the training of each of them. An ensemble of classifiers is basically composed of a group of machines that try to solve the same problem but under different conditions –diversity– and their outputs are combined aiming at obtaining a system that hopefully is more accurate than any of its members [3]. To achieve this goal, the members of the ensemble, usually known as base learners or simply learners, must be diverse. Put it very simple, two machines are diverse when their errors are not coincident.

In this regard, there are two basic families of ensembles. The first, we are called committees, consists in training the base learners with different examples

and their outputs are aggregated, usually by simple procedures (direct averaging or majority vote, for example). Among committees, we can mention two relevant techniques: Bagging [4], in which bootstrap resampling of the samples provides the training sets for the learners, and label switching [5], in which randomly switching examples' labels serves to introduce diversity. The second family of ensembles, which we call consortia, simultaneously train the units and the aggregation. Among consortia, boosting, whose basic forms appeared in [6, 7], has proved to be very effective for improving the performance of single learners. The key aspect is to iteratively design and aggregate classification units paying more attention to the examples that present higher classification errors. Mix-ture of Experts [8], the second most relevant consortia designs, show moderate performance when designed for classification, and the existing modifications to increase it, such as [9], require a huge computational effort.

A different kind of ensembles are those resulting from decomposing a multi-class problem into a number of binary problems whose solutions indicate the class corresponding to each sample. They are called binarization techniques.

The idea of combining binarization methods with one of the above-mentioned techniques (committees or consortia) for solving multi-class problems is not something novel. In [10], it was studied the performance of combining both bagging and binarization techniques over one dataset. The results obtained were not very satisfactory since the overall performance was the same as with only bag-ging. In [11], bagging and binarized ensembles were evaluated for nine datasets using a bias-variance framework and making comparisons with single neural networks based classifiers. The results obtained outperformed bagging and bina-rization in some cases, while in other cases gave similar results.

In this work, we take a further step and, we also explore a simple additional alternative that could be applied together with the combination of binarization techniques and standard diversification methods, aiming at increasing the performance of the overall ensemble. This alternative is pre-emphasis, i.e., weighting the training samples according to an auxiliary classifier, taking into account the critical character of each sample, i.e., its proximity to the classification bor-der and its classification error following the ideas of [12, 13]. In [14], we proposed flexible enough pre-emphasis approaches that allowed remarkable improvements, requiring a very modest computational cost in the design phase, but not in operation, i.e., to classify unseen samples.

The rest of the paper is organized is follows. In Sect. 2, we present a brief review of ensembles that are produced by binarizing multi-class prob-lems. Section 3 presents the weighting function that is used for pre-emphasis in multi-class problems. Section 4 describes the experimental framework: the selected datasets and the MLP based machines, the ensemble architectures to be studied, the binarization method to be applied, as well as the type of auxiliary classifier that is used to determine the amount of pre-emphasis needed for the training samples. Experimental results and their discussion are also included in this section. The most important conclusions of our work and some open research lines close this contribution in Sect. 5.

2 Binarization

There are three popular techniques for reducing a multi-class problem into a series of binary classification problems, namely: One vs. One (OvO), One vs. Rest (OvR) (also known as One vs. All), and Error Correcting Output Codes (ECOC) [15]. ECOC methods come from the area of communications for correcting data errors during transmission. They are based on adding some redundant information to the block to be transmitted, hence obtaining a codeword. In the context of ensembles, the use of ECOC consists in creating base classifiers and training them according to the information obtained from a pre-established code matrix. Experimental work has shown that ECOC offers improvement over OvO and OvR classification methods [15,16]. This is basically the reason why in this work we explore the application of ECOC for binarizing ensembles.

We continue with a short review of ECOC technique. Table 1 shows an example of an ECOC matrix for a 4-class classification problem [16].

Table 1. An exhaustive ECOC for a 4-class decision problem [16]. C_0 to C_3 are the classes, P_0 to P_6 are the binary problems. The codeword (row) that is nearest to the vector of units' outputs indicates the class to be selected.

Class	Problem						
	P_0	P_1	P_2	P_3	P_4	P_5	P_6
C_0	1	1	1	1	1	1	1
C_1	0	0	0	0	1	1	1
C_2	0	0	1	1	0	0	1
C_3	0	1	0	1	0	1	1

In the example at hand, each class is assigned a unique binary string of length 7. The string is also called codeword. For example, Class 2 (C_2) has the codeword 0011001. During the training process, one binary problem is learned for each column. In this respect, for the first column, we build a binary classifier to separate $\{C_0\}$ from $\{C_1, C_2, C_3\}$. Thus, it seems clear to notice that seven classifiers are trained in this way. To classify a new sample, $\mathbf{x}^{(n)}$, all seven binary classifiers are evaluated to obtain a 7-bit string. Finally, the given sample is classified by computing the similarity between the obtained 7-bit string and the codeword for each class, by using the Hamming distance metric.

3 Proposed Pre-emphasis Function

As mentioned in the Introduction, we have considered in this work the training of machines using pre-emphasized samples where the amount of pre-emphasis is determined by an auxiliary classifier.

For binary classifiers, the pre-emphasis used on training sample $\mathbf{x}^{(n)}$ can be written as indicated in the following double-convex combination of three terms:

$$p\left(\mathbf{x}^{(n)}\right) = \alpha + (1-\alpha)\left[\beta\left(e_a^{(n)}\right)^2 + (1-\beta)\left(1 - \left[o_a^{(n)}\right]^2\right.\right. \quad (1)$$

where $e_a^{(n)}$ is the classification error for input sample $\mathbf{x}^{(n)}$ and $o_a^{(n)}$ is the output of the auxiliary classifier for sample $\mathbf{x}^{(n)}$. It seems clear to notice that the underlying idea of the pre-emphasis function is that the training samples should be weighted based on two measures: How large the error is in the auxiliary classifier and how close its output is to the decision boundary.

The two pre-emphasis parameters, α and β, have values between zero and one ($0 \leq \alpha, \beta \leq 1$), and are determined by a process of Cross-Validation (CV) in our experiments.

Extending expression (1) to the case of multi-class formulations is not an easy task since the term $\left(1 - \left[o_a^{(n)}\right]^2\right.$ does not have a direct equivalent. However, if discriminative forms are considered, which are effective for training multi-class machines, it is possible to replace it for $(1 - |o_{ac}^{(n)} - o_{ac'}^{(n)}|)$ or similar forms, where $o_{ac}^{(n)}$ is the softmax output of the auxiliary classifier for the true class and $o_{ac'}^{(n)}$ is the output whose value is the nearest to $o_{ac}^{(n)}$ among the rest.

4 Experiments and Their Discussion

4.1 Databases

We will limit our presentation of results to a few appropriately selected multi-class databases that are frequently used as benchmark sets for this kind of experiments: the synthetic dataset Firm-Teacher Clave-Direction Classification [17] and two real problems, Satimage [18] and Vehicle [19]. All of them are obtained from the UCI Repository of Machine Learning Databases [20]. Table 2 illustrates the main characteristics of these problems: number of classes, number of dimensions and numbers of training and test samples. When the dataset had no predefined train/test partitions (that is, the case of Firm-Teacher Clave-Direction Classification and Vehicle datasets), a random partition has been created with 70/30% for training and test, respectively, keeping the relative proportions of the classes in each subset. From now on, we will denote the databases by their three first letters.

4.2 Machines and Their Designs

The architectures under study as well as the single classifier employ MLPs with one-hidden layer as base learners because they are unstable and powerful enough machines. They are trained by the Back-Propagation algorithm to minimize the mean squared error between the desired output and what the network actually

Table 2. Characteristics of the benchmark problems.

Dataset	Notation	# Train samples	# Test samples	Dimension	# Classes
Firm-Teacher Clave-Direction	Fir	10800*	-	16	4
Satimage	Sat	4435	2000	36	6
Vehicle	Veh	946*	-	18	4

*The total number of samples of the datasets without separate train and test sets (Fir and Veh) are listed under the number of training samples. For these datasets, a random partition has been created with 70/30% for training and test, respectively, keeping the relative proportions of the classes in each subset.

outputs, initializing all the weights at random values from a $[-0.2, 0.2]$ uniform distribution. The learning rate for both layers (hidden and output layers) is set to be 0.01, which has been experimentally proven to allow to reach convergence.

As mentioned, the architecture of the MLP based classifiers, explored in the experiments, consists of three layers (input, hidden and output layers). In this architecture, the number of input neurons corresponds to that of the attributes used to characterize the input samples (that is, it is consistent with the dimension of the datasets shown in Table 2), the number of output neurons is related to the classes we are interested in (in a binary classification problem may be enough to have a single output neuron) and the number of hidden neurons (H) depends on the adjustment of the complexity of the MLP. To achieve an optimal ensemble behavior, this appropriate number of hidden neurons has to be fixed. Despite each ensemble can need a different H value, aiming at carrying out a fair comparison in computational terms, we have preferred to fix the same H for all the bagging sampling rates. Then, this parameter is established by means of a 20-run $\times$ 5-fold CV.

However, when using the binarization technique, for each binary problem, a different H has to be fixed. In this case, for each dichotomic problem, the value of H is separately obtained also by means of a 20-run $\times$ 5-fold CV. This obviously increases the computational cost.

An 80/20 early stopping mode is applied to stop training.

4.3 Conventional Diversification

Bagging is carried out by means of conventional bootstrap (sampling with replacement) in our experiments, exploring different sizes B of the diversified training sets: 60, 80, 100, 120 and 140% the size of the original set of training examples. We also examine different number of ensemble units, M = 11, 21, 31, 41, 51, 101 and 201. We remark that these values serve to ensure that performances saturate, as we will see later.

We apply bagging in two different forms:

1. To construct ensembles of different, diversified MLPs;
2. Just at the classification layer, applying the diversification after getting the output values of a single MLP which is trained without diversity.

We will denote these designs by MLP-O (overall) and MLP-T (T-form) architectures, respectively. With this in mind, when we apply bagging to multi-class problems, two different architectures are obtained:

- MLP-BINARIZED-O: As shown in Fig. 1(a), bagging is applied to input data to construct M machines, and the final class is decided by a majority vote of their outputs that are obtained by means of ECOC ensembles that follow each diverse MLP;
- MLP-BINARIZED-T: In this scenario, as illustrated in Fig. 1(b), there is only one MLP structure and bagging is being imposed to its outputs values to construct M diverse ECOC ensembles, whose outputs are also aggregated by a majority vote rule.

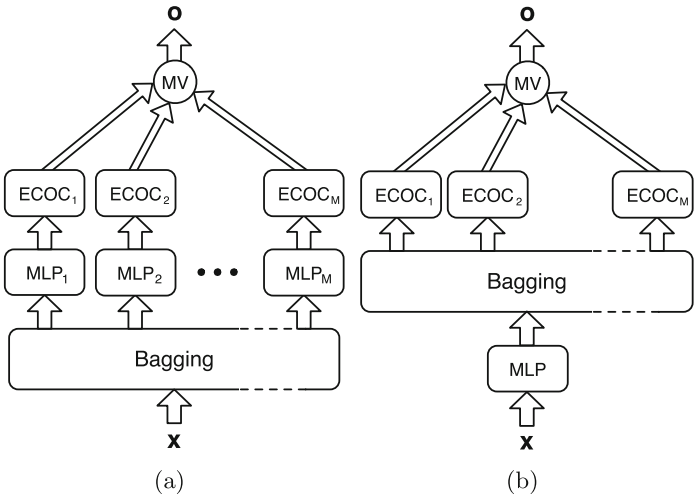

Fig. 1. Illustrative architectures showing the ways of combining diversity and binarization techniques in this work for solving multi-class problems: MLP-BINARIZED-O architecture (Fig. 1(a)) and MLP-BINARIZED-T architecture (Fig. 1(b)). MLP is a single Multi-Layer Perceptron; $ECOC_n$ are the diverse ECOC ensembles. MV: majority vote; **x**: input; **o**: output class decision.

It is clear to notice that O structures theoretically require a higher design effort, because M MLP machines have to be trained. To reduce it to affordable computational charges, we apply a frequently used simplification: to design more than M MLP bagging units (450, here) for each value of B, and to select M of them at random. Average performances are obtained for 10 independent selections. Of course, this reduces a little bit the diversity advantage, but we remark that our goal here is just to verify if diversity can improve single shallow MLP performance.

4.4 Binarization

In our series of experiments, we will use the C-class exhaustive ECOCs of [16], for the combination between overall and T-form architecture and bagging shown in Fig. 1. According to [16], for classification problems of $3 \leq C \leq 7$ classes, we construct a code length $2^{C-1} - 1$ as follows. Row 1 is all ones. Row 2 consists of 2^{C-2} zeros followed by $2^{C-2} - 1$ ones. Row 3 consists of 2^{C-3} zeros, followed by 2^{C-3} ones, followed by 2^{C-3} zeros, followed by $2^{C-3} - 1$ ones. In row i, there are alternating runs of 2^{C-i} zeros and ones.

4.5 Pre-emphasis

In the design of a single classifier, it seems clear that pre-emphasis weighting should be applied to the input training samples. However, for both MLP-BINARIZED-O and MLP-BINARIZED-T architectures previously described, the question arising here is: What is the way to apply the pre-emphasis weighting? In the first scenario, MLP-BINARIZED-O structure, the pre-emphasis is applied to the inputs to each diverse MLP, and in the second situation, MLP-BINARIZED-T architecture, two options have been considered in our experiments: The first is to weight the inputs to the single MLP and the second consists of separately pre-emphasizing each binary problem that appears after applying bagging at the outputs values of the single MLP based classifier. Preliminary results showed that best performances were obtained when weighting the input data to the binarized problems and, consequently, this is the way we have chosen for pre-emphasizing samples in the MLP-BINARIZED-T architecture in our experiments.

To determine the values for the pre-emphasis parameters, α and β, we used 10-run $\times$ 5-fold CV for the three datasets. As mentioned, the values for these parameters that we considered were in the interval $[0, 1]$ at increments of 0.1.

The auxiliary machine used in the experiments that provides the values to compute pre-emphasis weights according to expression (1) is an MLP with one-hidden layer of 50 neurons, because it was experimentally shown that this number of neurons provided the best results over a validation set for the three datasets.

4.6 Experimental Results

The results of our experiments (% error rate averages $\pm$ standard deviations, averaged for 10 different selections of learners in the case of bagging and for 10 runs when using ECOC ensembles) are shown in Tables 3 (Fir), 4 (Sat), and 5 (Veh) for the test sets. Best results are indicated in boldface (even if statistical differences are not significant). Additionally, we also include the performance of the design which does not apply neither bagging diversity nor ECOC binarization (denoted as MLP), as well as the designs which only apply bagging (indicated as MLP bagging) and the designs which employ ECOC without bagging (marked as MLP-ECOC), aiming at appreciating what is the advantage which can be attributed to the inclusion of the combination of different forms of diversity

explored in this work. As mentioned in Sect. 4.3, although different sizes B of the diversified training sets (60, 80, 100, 120 and 140%) have been explored as well as different number of ensemble units (M = 11, 21, 31, 41, 51, 101 and 201), Tables 3, 4 and 5 summarized the most remarkable results.

Having a look at the tree mentioned tables, the first thing to remark is that clear performance saturation effects appear when M or B increase. This indicates that extending the exploration margins is unnecessary. Furthermore, for the three databases under study, all kinds of diversity explored in this work appear to be effective, both O and T bagging: as an example, for Veh dataset, error reduction achieved is about 30%. Obviously, the T-form architecture must be preferred, because it requires lower computational training and operation efforts. Please note that in general, no significant differences appear between separate bagging and ECOC diversifications.

A detailed observation of the mentioned tables leads to the following conclusions:

- For Fir dataset: Directly applying bagging or ECOC achieved success in improving performance. Note that a simple ECOC reduces 18% the error rate, while the best scenario when applying bagging achieves a reduction of about 14%. Results are slightly better when applying jointly ECOC and bagging, M = 31 and B = 120% T-design being the best option in which error rate is approximately reduced up to 22%.
- For Sat dataset: Applying bagging or ECOC separately achieves reducing the error, ECOC ensembles being those which obtain the greater reduction (≈8%), as just happened for the previous dataset. However, when jointly applying ECOC and bagging, results are negative for small values of B, but moderate benefits appear when parameter B increases, M = 21 and B = 140% T-design achieving an error reduction of about 11%.
- For Veh dataset: Simple ECOC approximately reduces 25% the error rate, which is greater than the best error reduction obtained when applying bagging (≈22%). On the other hand, we can observe that jointly applying ECOC and bagging, results are better, especially when M and B parameters increase, M = 51 and B = 120% T-design reaching an error reduction of roughly 31%.

  Please note that if the number of training samples in the datasets is low, the results are more unstable and differences are statistically less significant.

We have taken a further step and we have applied pre-emphasis techniques to the designs that provided the best performances. Table 6 shows the results. For comparative purposes, the results obtained when the pre-emphasis is not applied (see Tables 3, 4 and 5 for further details) are also presented, as well as the results of the single MLP classifier when the input samples are pre-emphasized (MLP PrE) and when they are not pre-emphasized (MLP). For each problem, the values for α, β obtained by CV are given in parentheses.

It can be observed that the PrE MLP-BINARIZED-T performances are clearly and systematically better than any other results. Error rates decay around 40% in some cases.

Table 3. Test (%) error rate averages ± standard deviations for the diversified classifiers applied to problem Firm-Teacher Clave-Direction (Fir).

MLP	14.66 ± 0.18		
MLP-ECOC	12.03 ± 0.13		
MLP bagging	B		B
M	120%		140%
11	12.87 ± 0.11		12.75 ± 0.15
21	12.73 ± 0.10		$\mathbf{12.66 \pm 0.14}$
31	12.83 ± 0.09		12.71 ± 0.12
MLP-BINARIZED-O bagging	B	B	B
M	100%	120%	140%
11	12.75 ± 0.13	12.04 ± 0.11	12.13 ± 0.10
21	12.64 ± 0.11	$\mathbf{11.70 \pm 0.10}$	12.01 ± 0.11
31	12.58 ± 0.09	11.83 ± 0.09	11.97 ± 0.09
MLP-BINARIZED-T bagging	B	B	B
M	100%	120%	140%
21	12.47 ± 0.07	11.65 ± 0.08	11.97 ± 0.08
31	12.35 ± 0.07	$\mathbf{11.53 \pm 0.07}$	11.86 ± 0.07
51	12.33 ± 0.07	11.60 ± 0.07	11.75 ± 0.08

Table 4. Test (%) error rate averages ± standard deviations for the diversified classifiers applied to problem Satimage (Sat).

MLP	10.28 ± 0.98		
MLP-ECOC	9.44 ± 0.40		
MLP bagging	B	B	B
M	100%	120%	140%
11	10.58 ± 0.84	10.04 ± 0.63	10.14 ± 0.60
21	10.39 ± 0.71	$\mathbf{9.88 \pm 0.55}$	10.19 ± 0.62
31	10.43 ± 0.69	9.94 ± 0.57	10.21 ± 0.55
MLP-BINARIZED-O bagging	B	B	B
M	100%	120%	140%
11	10.34 ± 0.44	9.24 ± 0.38	9.35 ± 0.39
21	10.23 ± 0.37	$\mathbf{9.31 \pm 0.42}$	9.29 ± 0.37
MLP-BINARIZED-T bagging	B		B
M	120%		140%
11	9.31 ± 0.39		9.23 ± 0.34
21	9.29 ± 0.36		$\mathbf{9.18 \pm 0.29}$
31	9.25 ± 0.31		9.22 ± 0.28

Table 5. Test (%) error rate averages ± standard deviations for the diversified classifiers applied to problem Vehicle (Veh).

MLP	18.91 ± 3.79		
MLP-ECOC	14.20 ± 3.40		
MLP bagging	B	B	
M	120%	140%	
51	14.93 ± 3.41	14.73 ± 3.16	
101	14.80 ± 3.35	$\mathbf{14.69 \pm 3.11}$	
201	14.89 ± 3.27	14.75 ± 3.08	
MLP-BINARIZED-O bagging	B	B	B
M	100%	120%	140%
31	14.72 ± 3.24	13.59 ± 3.33	13.82 ± 3.20
51	14.68 ± 3.19	$\mathbf{13.51 \pm 3.29}$	13.85 ± 3.21
101	14.51 ± 3.20	13.63 ± 3.26	13.80 ± 3.18
MLP-BINARIZED-T bagging	B	B	B
M	100%	120%	140%
31	14.87 ± 3.29	13.19 ± 3.30	13.42 ± 3.25
51	14.78 ± 3.19	$\mathbf{13.04 \pm 3.25}$	13.18 ± 3.14
101	14.80 ± 3.15	13.08 ± 3.22	13.16 ± 3.11

Table 6. Comparison of test (%) error rate averages ± standard deviations for the benchmark problems considered in our experiments when pre-emphasis is applied.

Dataset	MLP		MLP-BINARIZED-T bagging	
	No PrE	PrE (α, β)	No PrE	PrE (α, β)
Fir	14.66 ± 0.18	11.87 ± 0.05 $(0.3, 0.5)$	11.53 ± 0.07	10.34 ± 0.03 $(0.4, 0.6)$
Sat	10.28 ± 0.98	9.26 ± 0.31 $(0.4, 0.3)$	9.18 ± 0.29	8.13 ± 0.17 $(0.4, 0.5)$
Veh	18.91 ± 3.79	14.05 ± 2.32 $(0.6, 0.3)$	13.04 ± 3.25	11.59 ± 1.67 $(0.5, 0.4)$

All these results seem to indicate that pre-emphasizing the samples of binarized ensembles is extremely useful to obtain high performance MLP-based classifiers and that T-form diversity is fruitful if binarization is applied.

5 Conclusions

In this paper, we have not only checked that binarization is effective to improve standard diversification techniques –bagging, in particular–, but we have also seen how a simple technique consisting in weighting the training examples allows to obtain even more important performance improvements. In all the three datasets under study, experimental results show that to combine a flexible enough pre-emphasis function with ECOC binarized and diversified MLP based

classifiers, permits an error reduction bigger than their separate application, achieving until 40% error rate reductions in a dataset.

Between the two explored ensemble architectures –O form, corresponding to a full diversification and T-form, in which diversity is applied after designing the MLP based classifier, the second approach reaches lower error rates. Since it is also better from the perspective of the required training and operating computational efforts, it must be preferred for MLP based classifiers. It is also worth mentioning that the saturation performance with respect to the diversification parameters make their selection an easy validation problem.

Needless to say, much more work –considering other problems, other classifiers, and other sources of diversity– is needed to completely appreciate the potential of combining diversity and binarization techniques, including appropriate pre-emphasis sample weighting schemes. Finally, we must remark that to combine this kind of designs with other auxiliary techniques that serve to improve the performance of MLPs classifiers is a promising way to get excellent, or even record performance practical implementations.

Acknowledgments. This work has been partly supported by research grants CASI-CAM-CM (S2013/ICE-2845, DGUI-CM and FEDER) and Macro-ADOBE (TEC2015-67719-P, MINECO).

References

1. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Math. Control Sig. Syst. **2**, 303–314 (1989)
2. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are uni-versal approximators. Neural Netw. **2**, 359–366 (1989)
3. Dietterich, T.G.: Ensemble methods in machine learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000). doi:10.1007/3-540-45014-9_1
4. Breiman, L.: Bagging predictors. Mach. Learn. **4**, 123–140 (1996)
5. Hinton, G.E., Osindero, S., Teh, Y.: A fast learning algorithm for deep belief networks. Neural Comput. **18**, 1527–1554 (2006)
6. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55**, 119–139 (1997)
7. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Mach. Learn. **37**, 297–336 (1999)
8. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixture of local experts. Neural Comput. **3**, 79–87 (1991)
9. Omari, A., Figueiras-Vidal, A.R.: Feature combiners with gate-generated weights for classification. IEEE Trans. Neural Netw. Learn. Syst. **24**, 158–163 (2013)
10. Leisch, F., Hornik, K.: Combining neural network voting classifiers and error correcting output codes. In: MEASURMENT 1997 (1997)
11. Cemre, Z., Windeatt, T., Yanikogl, B.: Bias-variance analysis of ECOC and bagging using neural nets. In: Okun, O., Valentini, G., Re, M. (eds.) Ensembles in Machine Learning Applications. Studies in Computational Intelligence, vol. 373, pp. 59–73. Springer, Heidelberg (2011)

12. Gómez-Verdejo, V., Ortega-Moral, M., Arenas-García, J., Figueiras-Vidal, A.R.: Boosting by weighting critical and erroneous samples. Neurocomput. **69**, 679–685 (2006)
13. Gómez-Verdejo, V., Arenas-García, J., Figueiras-Vidal, A.R.: A dynamically adjusted mixed emphasis method for building boosting ensembles. IEEE Trans. Neural Netw. **19**, 3–17 (2008)
14. Alvear-Sandoval, R.F., Figueiras-Vidal, A.R.: An experiment in pre-emphasizing diversified deep neural networks. In: 24th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, pp. 527–532 (2016)
15. Rokach, L.: Pattern Classification Using Ensemble Methods. World Scientific, Singapore (2010)
16. Dietterich, T.G., Bakiri, G.: Solving multi-class learning problems via error-correcting output codes. J. Artif. Intell. Res. **2**, 263–286 (1995)
17. Vurkaç, M.: Clave-direction analysis: a new arena for educational and creative of music technology. J. Music Technol. Educ. **4**, 27–46 (2011)
18. Giannakopoulos, X., Karhunen, J., Oja, E.: An experimental comparison of neural algorithms for independent component analysis and blind separation. Int. J. Neural Syst. **9**, 99–114 (1999)
19. Siebert, J.P.: Vehicle Recognition Using Rule Based Methods. Turing Institute Research Memorandum TIRM-87-018, Glasgow, Scotland (1987)
20. Lichman, M.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA (2013)