New One Versus$^{All}_{One}$ Method: NOV@

A. Arruti[a], I. Mendialdua[b,*], B. Sierra[b], E. Lazkano[b], E. Jauregi[b]

[a]*Department of Computer Architecture and Technology*
University of the Basque Country UPV/EHU
Donostia-San Sebastian 20018, Spain.
[b]*Department of Computer Science and artificial Intelligence*
University of the Basque Country UPV/EHU
Donostia-San Sebastian 20018, Spain.

Abstract

Binarization strategies decompose the original multi-class dataset into multiple two-class subsets, learning a different binary model for each new subset. One-vs-All (OVA) and One-vs-One (OVO) are two of the most well-known techniques: One-vs-One separates a pair of classes in each binary sub-problem, ignoring the remaining ones; and One-vs-All distinguishes one class from all the other classes. In this paper, we present two new OVA and OVO combinations where the best base classifier is applied in each sub-problem. The first method is called OVA+OVO since it combines the outputs obtained by OVA and OVO decomposition strategies. The second combination is named *New One Versus*$^{All}_{One}$ (NOV@), and its objective is to solve the problems found in OVA when different base classifiers are used in each sub-problem. In order to validate the performance of the new proposal, an empirical study has been carried out where the two new methods are compared with other well-known decomposition strategies from the literature. Experimental results show that both methods obtain promising results, especially NOV@.

Keywords: Decomposition Strategies, One against One, One against All

1. Introduction

The goal in a supervised classification problem consists in classifying a new unlabelled example x in its correct class using a training set. Let $TR = \{x_i, \theta_i\}_{i=1}^{N}$ denote a training set of N well-labelled examples, where x_i represents the i-th individual feature vector and θ_i represents the class the individual belongs to. In the particular case of the K-class problem, being $\theta \in \{1, .., K\}$, the class label is commonly defined as an integer. Based on the TR, the supervised classification techniques create a "general rule" or a function which is used to classify each new unlabelled case. This general rule is also known as a classifier.

For some types of classifiers, such as SVM, it is much more easier to build a classifier to distinguish just between two classes. However, many real world problems are multi-class problems, i.e. $K > 2$. In view of that, there are some techniques, known as class-binarization techniques, which divide the original multi-class problem into many binary classification problems. These techniques are two-step methods: in the first step, called decomposition step, a

*Corresponding author at: Robotics and Autonomous System Research Group, University of the Basque Country UPV/EHU, Donostia-San Sebastian 20018, Spain

Email address: inigo.mendialdua@ehu.es (I. Mendialdua)

classifier is learned for each of the sub-problems, and in the second step, called combination step, the outputs of these classifiers are combined to obtain the final prediction.

There are three different class-binarization strategies: "One-vs-All" (OVA), "One-vs-One" (OVO) and "Error Correcting Output Codes" (ECOC). OVA and OVO are the most relevant ones in the literature due to their simplicity and clarity.

- One-vs-All (OVA) [3]: In each sub-problem one class is compared with the rest of classes.
- One-vs-One (OVO) [13]: In each sub-problem only the cases belonging to two classes are compared with each other, and the remaining ones are ignored.
- Error Correcting Output Codes (ECOC) [9]: In each sub-problem all the classes are grouped into two groups, and the two groups are compared with each other.

The procedure followed in the classical binary classification strategies is to use the same base classifiers in each binary sub-problem. However, if the selected base classifier does not correctly discriminate in some of the sub-problems, the obtained results will be wrong. To overcome this drawback, each sub-problem can be treated as an independent classification problem so that a different base classifier can be used for each sub-problem.

In this paper, we propose two new combinations of the methods OVA and OVO. We compare these two methods with other class-binarization strategies over 20 UCI databases, and obtain promising results. In all methods we try to find the best base classifiers for each sub-problem. To do so, we have chosen several well-known classifiers from different Machine Learning paradigms: SVM, C4.5 Decision Tree, Ripper, K-NN, Multilayer Perceptron and Naive Bayes. The first combination that we have proposed is called OVA+OVO. OVA+OVO basically combines the sub-problems obtained after the OVA and OVO methods are applied. Comparing this new method with other class-binarization strategies we have found that strategies – such as OVA – are not suitable when different base classifiers are used in each sub-problem. Therefore, we propose a second approach that we have called $New\ One\ Versus^{All}_{One}$ (NOV@). NOV@ is an extension of OVA: at decomposition time OVA is applied, whereas at combination time the majority vote is used to make the final decision. In case of a tie among several classes, OVO is applied for tie-breaking – taking just into account the tied classes.

Although in the specialized literature, there are several proposals that select the best base classifier for each sub-problem, none of them compares different class-binarization techniques in order to study how each technique performs. In our work we have compared the two new methods with other state-of-the-art class-binarization strategies in an empirical study.

The rest of the paper is organized as follows. In Section 2 we review the decomposition techniques, with special attention to OVA and OVO strategies. In Section 3 we show the compatibility between OVA and OVO strategies and we present the first proposed method: OVA+OVO. Section 4 describes our second approach and Section 5 shows the results of the experiments carried out. Finally, Section 6 states the conclusions of our work and suggests future research lines.

2. Class-Binarization

The first class-binarization strategies were made to solve the problems that some base classifiers have for the multi-class problems, since algorithms such as Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP), worked better for binary problems. However, due to the good results obtained, the use of these strategies has been extended to other base classifiers, such as Ripper [13] or C4.5 [37].

Class-binarization is composed of two steps: decomposition and combination.

In the decomposition step the original problem is divided into several binary sub-problems. The most popular strategies consist of grouping classes, in this way each binary classifier compares two groups of classes between them. Commonly the code-matrix is used to represent how the classes are grouped.

Figure 1 shows a code-matrix example, where each row represents a class and each column represents a binary classifier. Each class takes values in the set {-1,0,+1}, where +1 indicates the classes associated to the positive-class, -1 indicates the classes associated to the negative-class and 0 indicates that the class is ignored for this binary problem. In Figure 1 how a 5-class problem $\{\theta_1,\theta_2,\theta_3,\theta_4,\theta_5\}$ is decomposed into 5 binary problems $\{f_1,f_2,f_3,f_4,f_5\}$ can be seen.

$$
\begin{array}{c}
 \\
 \\
classes \\
 \\
 \\
 \\
 \\
\end{array}
\begin{cases}
 \\
 \\
 \\
 \\
 \\
\end{cases}
\begin{array}{c}
\overbrace{}^{classifiers} \\
f_1\ f_2\ f_3\ f_4\ f_5 \\
\left(\begin{array}{ccccc}
1 & 0 & -1 & -1 & 0 \\
1 & 1 & -1 & -1 & 1 \\
-1 & 1 & 1 & -1 & 0 \\
0 & -1 & 0 & 1 & 0 \\
-1 & -1 & 0 & -1 & -1
\end{array}\right)
\end{array}
\begin{array}{l}
\theta_1 \\
\theta_2 \\
\theta_3 \\
\theta_4 \\
\theta_5
\end{array}
\qquad
\begin{array}{l}
f_1 \rightarrow \theta_1, \theta_2 \ vs\ \theta_3, \theta_5 \\
f_2 \rightarrow \theta_2, \theta_3 \ vs\ \theta_4, \theta_5 \\
f_3 \rightarrow \theta_3 \ vs\ \theta_1, \theta_2 \\
f_4 \rightarrow \theta_4 \ vs\ \theta_1, \theta_2, \theta_3, \theta_5 \\
f_5 \rightarrow \theta_2 \ vs\ \theta_5
\end{array}
$$

Figure 1: Example of a code-matrix

For instance, it can be seen that the classifier f_1 is constructed in such a manner that the cases belonging to θ_1 and θ_2 are grouped in class +1 and the cases in θ_3 and θ_5 in class -1. So this classifier compares θ_1 and θ_2 classes with θ_3 and θ_5, while the cases that belong to θ_4 are ignored.

In classification time, each binary classifier returns a prediction. So the combination step consists of combining these predictions. Therefore, it is crucial to select a proper combination of the outputs in order to make a correct prediction.

Different decomposition strategies have been developed. Two of the most popular are OVA and OVO, which are described next.

2.1. One-Vs-All (OVA)

OVA decomposition scheme divides a K class multi-class problem, $\theta_1, ..., \theta_K$, into K two-class problems, where each binary problem discriminates one class from the others.

In Figure 2(a) OVA's code-matrix for 4 classes is shown: in each classifier one class is represented as positive class while all the other 3 classes are represented as negative-class.

As one class is compared with all the other classes, most of the binary sub-problems are unbalanced. It is known that one of the drawbacks of an unbalanced problem is the underestimation for the minority classes, thereby the most represented class is selected in most cases. In view of that, in OVA it is very common that all sub-problems return a class-negative prediction, hence, ties are usual in the final decision when the majority vote is used. Thus, it is more efficient to use the confidence level of each classifier to decide the final output. The class with the highest confidence is the selected decision.

2.2. One-Vs-One (OVO)

OVO decomposition scheme divides a K class multi-class problem, $\theta_1, ..., \theta_K$, into $K(K-1)/2$ two-class sub-problems. In each sub-problem a classifier is learned using only the cases that belong to a pair of classes (θ_i,θ_j), where $\theta_i \neq \theta_j$; the remaining cases are ignored.

In Figure 2(b) OVO's code-matrix for 4 classes can be observed: in each classifier one class is represented as +1 class, another one is represented as -1 and the remaining 2 classes are represented as 0.

There are several strategies to combine the output, the simplest way to combine the outputs is to use the majority vote strategy [13, 12] also called as Max-Wins; the most voted class is the selected one. An immediate extension is the Weighted Voting (WV): to use the confidence level of each binary problem as a vote. Its robustness has been shown in [14]. Hastie and Tibshirani [22] proposed a new method called Pairwise Coupling (PC). The aim of the method is to find the best approximation of the class posterior probabilities given the posterior probabilities of the pairwise sub-problems. To do so, they transform the problem into an iterative problem where they try to minimize the average weighted Kullback-Leiber divergence between the obtained pairwise estimates and the true pairwise probability values.

OVO has several drawbacks 3 of the main disadvantages of OVO are the following:

1. Unclassifiable regions: it is possible that each binary classifier votes for a different class, hence there is no winner. Thus, some tie-breaking technique has to be applied.

$$\begin{pmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \end{pmatrix} \quad \begin{pmatrix} +1 & +1 & +1 & 0 & 0 & 0 \\ -1 & 0 & 0 & +1 & +1 & 0 \\ 0 & -1 & 0 & -1 & 0 & +1 \\ 0 & 0 & -1 & 0 & -1 & -1 \end{pmatrix}$$

(a) One-Vs-All (b) One-Vs-One

Figure 2: OVA and OVO code-matrix

2. Number of classifiers: Compared with OVA, it can be seen that OVO creates more sub-problems. Moreover, the disadvantage of having so many sub-problems is that most of them are irrelevant and they are forced to give wrong answers for many instances, because each classifier must assign every pattern to one of two classes. If a pattern belongs to class i, all the classifiers that are not trained to differentiate this class will cast wrong votes. However, OVO uses fewer examples in each sub-problem and, thus, has more freedom for fitting a decision boundary between the two classes.
3. Weak classifiers: The classical way selects the optimal base classifier for the database and all the sub-problems are classified with this classifier. As there are too many sub-problems, it is possible that this base classifier has difficulties to distinguish between all of them, thus, the classifiers return wrong results. This raises the question – should the same base classifier be used on all sub-problems or should sub-problems be tuned independently?

Several proposals has been developed in the literature in order to solve these problems.

In order to resolve the unclassifiable regions Platt et al. [35] published a new combination proposal called Decision Directed Acyclic Graph (DDAG). DDAG builds a rooted binary acyclic graph where in each node a classifier discriminates between two classes. The final answer is the class assigned by the leaf node. Liu et al. [29] proposed a tie-breaking technique, where OVO is applied using only the examples in the unclassifiable region.

On the other hand, other approaches try to reduce the number of binary classifiers in OVO using the Dynamic Classifier Selection [15][5]. Other authors propose the use of hierarchical structure. Fei and Liu [10] proposed a new architecture called Binary Tree of SVM (BTS). BTS is a binary tree where in each node two classes are distinguished. The main idea of BTS is to use the separating plane for these two classes, also to distinguish other classes. Chen et al. [6] proposed a new BTS version where they tried to select the binary SVM with the fewest number of separating lines. Following the same idea, to reduce the number of classifiers, the hierarchical structure has been extended to other class-binarization strategies [38, 30, 19, 32].

Finally other approaches try to solve the weak classifier problem using each sub-problem independently. Szepannek et al. [43] proposed to extend OVO selecting the optimal classifier for each pair of classes, i.e. the base classifier which obtains the best result. Something similar was proposed by Lebrun [27], where they tried to find the best hyper-parameters of the SVM for each sub-problem. Due to the high number of hyper-parameters in the SVM, they proposed to use an evolutionary algorithm. The experimental results of both works showed that they outperformed the classical individual base-classifier option. Liepert [28] also proposed a similar approach to Lebrun, but she concluded that the selection of the best models for each binary-classifier does not obtain a significant improvement.

In his PhD thesis, Reid [40] concluded that despite it is better to use the same base classifier for all the sub-problems when decision boundaries of the sub-problems have similar shapes, in the case where the decision boundaries have a different shape it is better to treat sub-problems independently.

2.3. Related Work

In several works in the literature, OVA and OVO have been compared, showing that in most of the cases OVO outperformed OVA [13, 23]. Rifkin and Klautau [41] did not consider that these experiments were carefully controlled and they demonstrated that when OVA classifier is well-tuned it performs as well as OVO.

In some recent works, it is possible to find some reviews related with the class-binarization strategies [31]. Other recent works made empirical studies for different binarization strategies [14, 18]. Both works analyse the behaviour of OVA and OVO for different base classifiers and the results of both works concluded that OVO outperformed OVA. However, neither work dares to contradict Rifkin and Klautau [41] arguing that they did not use fine-tuned classifiers.

Some authors proposed new approaches based on the combination of OVA and OVO. On the one hand, Moreira and Mayoraz [33] proposed to apply OVO taking into account the probability that the new example belonged to each

pair of classes. This probability was obtained following the OVA idea: creating a classifier that distinguishes between the two classes and the rest of the classes. On the other hand, García-Pedrajas and Ortiz-Boyer [17] and Ko and Byun [25] proposed a very similar idea to combine OVA and OVO. In an interesting motivation section García-Pedrajas and Ortiz-Boyer [17] showed that in the majority of the cases the correct class was between the two largest confident outputs of OVA. Thus, in their new method they obtain the two classes with the highest confidence level in OVA first. After that, OVO is applied and a classifier is built only taking into account the cases that belong to these two classes. This method was called All-And-One (A&O) [17].

3. Combining OVA and OVO

Hanse and Salomon [21] showed that in order to obtain good performance when classifiers are combined, the classifiers should be diverse among them. It is said that two algorithms are diverse when they commit different errors. If two algorithms are not diverse they commit the same errors, whereas if they are diverse they may be able to correct the committed errors.

It must be said that, although several diversity measures exists in the literature [26], they have been proved to be ineffective [44]. Hence, instead of applying those measures, we choose to perform a simple experiment using decision trees as base classifier, in order to conclude if OVA and OVO are compatible.

Twenty databases are used to test our hypothesis. All of them are obtained from the *UCI Machine Learning Repository* [4]. The characteristics of the databases are given in Table 1.

Domain	Num. of Instances	Num. of Attributes	Num. of Classes
Abalone	4177	8	29
Annealing	798	38	5
Arrhythmia	452	279	13
Balance	625	4	3
Car	1728	6	4
Cmc	1473	9	3
Dermatology	366	33	6
Ecoli	336	7	8
Flare	1389	11	6
Glass	214	9	6
Iris	150	4	3
Nursery	12960	8	5
Page-blocks	5473	10	5
Optdigits	5620	64	10
Pendigits	10992	16	10
Satimage	6435	36	7
Segment	2310	19	7
Vehicle	846	18	4
Waveform	5000	21	3
Wine	178	13	3
Winequality Red	1599	10	6
Winequality White	4898	10	7
Yeast	1484	8	10
Zoo	101	16	7

Table 1: The characteristics of the 20 databases used in this experiment

In Table 2 we show the results of the compatibility test. In the first two columns we show the accuracy obtained by OVO and OVA, while in the third the accuracy of OVA is shown considering only the cases where OVO makes errors. In the fourth column the accuracy of OVO is shown for the cases that OVA has failed.

	Accuracy of OVO	Accuracy of OVA	Accuracy of OVA when OVO fails	Accuracy of OVO when OVA fails
Abalone	24.946	18.793	12.268	18.910
Annealing	92.383	92.116	25.214	26.508
Arrhythmia	65.885	63.496	29.720	34.319
Balance	78.910	78.910	20.898	20.742
Car	95.914	95.972	34.780	32.725
Cmc	51.460	52.016	24.408	23.597
Dermatology	95.574	91.530	41.702	66.882
Ecoli	81.667	78.571	25.653	35.844
Flare	59.962	56.023	20.195	26.7482
Glass	63.084	60.374	30.566	35.531
Iris	93.467	92.800	1.250	12.500
Nursery	98.622	98.647	6.287	4.172
Optdigits	92.281	87.434	52.325	70.732
Page-blocks	96.824	96.642	17.794	22.202
Pendigits	95.950	94.039	58.541	71.864
Satimage	86.017	83.708	40.617	49.052
Segment	95.039	94.251	45.869	53.228
Vehicle	68.842	68.251	39.939	40.984
Waveform	76.212	74.144	49.187	53.193
Wine	88.202	89.888	54.609	47.018
WineRed	57.486	58.649	29.100	26.977
WineWhite	54.924	53.973	26.774	28.273
Yeast	56.267	55.418	17.459	19.071
Zoo	90.297	90.693	43.222	38.770

Table 2: Compatibility between OVO and OVA

It is interesting to note that in the case where one strategy fails the other gets a hit rate higher than 20% in most of the cases, which leads us to consider that the methods are compatible and that they could correct some of the errors made by the other strategy.

3.1. Our First Proposal to Combine OVA and OVO

This new method combines the sub-problems obtained applying OVA and OVO decomposition strategies. To take the final decision the results obtained for each sub-problem are combined applying the majority vote. We have called this new method OVA+OVO. Following the idea proposed by Szepannek et al. [43], we apply the most reliable base classifier for each sub-problem.

3.1.1. Decomposition

In our method, we propose to combine the sub-problems obtained with OVA and OVO; on the one hand, we create several sub-problems where the classes are compared by pairs, ignoring the other classes, and on the other hand, we create several sub-problems which compare one class with all the other classes.

Figure 3 shows the code-matrix of the sub-problems obtained in the decomposition phase in a 4-class problem. The 4 columns of the left are the sub-problems obtained with OVA and the next 6 columns are the sub-problems obtained with OVO.

We consider that the combination of these methods could obtain a classifier that outperforms both methods separately for two reasons:

1. As shown in Section 3 both methods are able to correct the errors committed by the other one.
2. As mentioned before, when majority vote is used for the final decision it is common for both, OVA and OVO, to produce ties.

$$\begin{array}{c} \overbrace{}^{One-Vs-All} \quad \overbrace{}^{One-Vs-One} \\ \left(\begin{array}{cccc|cccccc} +1 & -1 & -1 & -1 & +1 & +1 & +1 & 0 & 0 & 0 \\ -1 & +1 & -1 & -1 & -1 & 0 & 0 & +1 & +1 & 0 \\ -1 & -1 & +1 & -1 & 0 & -1 & 0 & -1 & 0 & +1 \\ -1 & -1 & -1 & +1 & 0 & 0 & -1 & 0 & -1 & -1 \end{array}\right) \end{array}$$

Figure 3: Code-matrix of OVA+OVO

$$\begin{array}{c} \begin{array}{cccccccccc} NB & 3NN & C4.5 & 3NN & C4.5 & SVM & NB & SVM & SVM & 3NN \end{array} \\ \left(\begin{array}{cccccccccc} +1 & -1 & -1 & -1 & +1 & +1 & +1 & 0 & 0 & 0 \\ -1 & +1 & -1 & -1 & -1 & 0 & 0 & +1 & +1 & 0 \\ -1 & -1 & +1 & -1 & 0 & -1 & 0 & -1 & 0 & +1 \\ -1 & -1 & -1 & +1 & 0 & 0 & -1 & 0 & -1 & -1 \end{array}\right) \end{array}$$

Figure 4: Different base classifier for each sub-problem

- In the case of OVA, it is possible that all sub-problems return a negative-class prediction.
- In the case of OVO, the most voted class could be more than one.

Therefore, we consider that combining their outputs could serve to break some of these ties.

3.1.2. Best base classifier for each sub-problem

Each sub-problem is treated independently from the others, the optimal base classifier for each one is sought. In a validation process each binary sub-problem is tested with different base classifiers, and the base classifier which obtains the best accuracy is selected.

In Figure 4 an example where a different base classifier is applied for each sub-problem can be seen.

3.1.3. Combination

To take the final decision, we have decided to apply the majority vote; the most voted class is that which is selected. We consider this due to three reasons:

1. As different base classifiers are used for each sub-problem, combination strategies based on the confidence level are not appropriate because each base classifier confidence level is calculated differently.
2. As stated in Galar et al. [14] the majority vote obtains robust results compared with other combination strategies in OVO.
3. It is the simplest one.

3.2. Experimental Setup

In this section we show the experimental results obtained with different databases. We compare our method with other state-of-the-art strategies and we run our experiments over the datasets shown in Tabla 1.

3.2.1. Classifiers

To carry out our experiments we use some classifiers from WEKA (Waikato Environment for Knowledge Analysis) [20].

Among the classifiers that Weka offers, we have selected the following ones:

- Naive Bayes [24], statistical learning algorithm. It is based on Bayesian rules and, given that the value of the class is known, it assumes independence between the occurrences of feature values to predict the class.
- J48 (C4.5 clone) [39], decision tree algorithm. It makes a post-pruning phase, based on error based pruning algorithm.

- IBK (K-NN clone) [2], distance based algorithm. An object is classified by a majority vote of its K nearest neighbors. The value of K is set to 3.
- SM0 (SVM clone) [34], kernel methods. It creates a hyperplane where the categories are divided by a clear gap that is as wide as possible.
- JRip (Ripper clone) [7], rule induction classifier. It builds a rule-set by repeatedly adding rules to an empty rule-set until all positive examples are covered.
- MultilayerPerceptron [42], an artificial neural network. It is a feedforward network of neurons which map input vectors to output vectors.

As it can be seen, the selected classifiers are from different natures in order to give variability and reliability to the experimental phase. It is worth saying that in our experiments we have used the default parameters of the classifiers.

3.2.2. Strategies summarized

In this sub-section we briefly describe the class-binarization strategies that are used for the comparison. It is worth remembering that in all the strategies the best base classifier is selected for each sub-problem.

- One-Vs-All (OVA): Each sub-problem compares one class with the rest of classes. The class with the highest confidence level is selected.
- All-And-One (A&O) [17, 25]: Combination of OVA and OVO. First OVA is applied and the two classes with the highest confidence level are selected. A classifier that discriminates between the selected classes is built and the result of the classifier is the final decision.
- Max-Wins [13, 12]: For decomposition OVO is applied: each sub-problem compares two classes between them, ignoring the rest. And the majority vote is used to take the final decision.
- Weighted Voting (WV): The weight for the vote is given by the confidence level of the classifier. The class with the largest sum value is predicted.
- Pairwise Coupling (PC) [22]: PC tries to find the posterior probability of each class ($p_1..p_K$) given the posterior probability of all the pairwise sub-problems (r_{ij}). To do so, the problem is transformed into an iterative problem where the Kullback-Leibler distance between r_{ij} and μ_{ij} is minimized ($\mu_{ij} = p_i/(p_i + p_j)$).
- Decision Directed Acyclic Graph (DDAG) [35]: The DDAG is equivalent to operating on a list. A list is initialized with all the classes. In each step a classifier discriminates between two classes selected from the list, and the class which is not selected is eliminated. The DDAG terminates when only one class remains in the list.
- OVA+OVO: Combination of OVA and OVO outputs. The majority vote is used to take the final decision.

3.2.3. Experimental Results

In order to give a real perspective, we have applied 5x2 fold cross validation to each database [8]. But firstly each binarization strategy needs a validation process to select the most accurate base classifiers for each binary sub-problem. Therefore, we have applied 5-hold out for each fold, where we have used the 70% as training set and the 30% as testing set.

In Table 3 we show the results obtained for our new method and those obtained with state-of-the-art methods. The best result is highlighted in bold. It can be observed that OVA+OVO obtains the best result in 14 databases. Although OVA+OVO obtains promising results, we have continued with the experiments and in the next Section we will show the main proposal of this paper.

4. Proposed Approach: NOV@

In this section we present our new proposal $New\ One\ Versus^{All}_{One}$. But before explaining the method, we will show the reasons that have led us to develop this proposal.

Database	OVA	A&O	Max-Wins	WV	PC	DDAG	OVA+OVO
Abalone	16.553	22.993	**26.598**	26.589	26.426	25.171	**26.598**
Annealing	98.062	97.929	98.196	98.196	98.129	98.085	**98.307**
Arrhythmia	63.850	65.265	69.292	69.292	**71.150**	68.407	71.018
Balance	90.128	89.840	90.096	90.929	90.865	90.641	**91.058**
Car	96.273	95.914	95.787	95.984	95.741	95.845	**96.296**
Cmc	48.350	49.220	52.831	52.492	52.465	51.840	**53.646**
Dermatology	95.519	97.158	97.541	**97.596**	**97.596**	97.541	97.432
Ecoli	82.202	83.452	83.095	**83.988**	83.869	83.631	83.810
Flare	54.540	58.668	**60.094**	59.756	59.587	59.606	**60.094**
Glass	64.206	63.458	64.112	64.486	**65.327**	63.458	65.140
Iris	**95.333**	94.800	94.800	94.800	94.800	94.800	94.933
Nursery	99.336	99.384	99.451	99.477	99.452	99.431	**99.653**
Optdigits	98.456	**98.480**	98.466	98.463	98.470	98.395	98.473
Page-blocks	96.707	96.799	96.751	96.828	96.667	96.777	**96.897**
Pendigits	**99.250**	99.221	99.210	99.214	99.207	99.143	99.238
Satimage	89.958	90.126	90.051	90.058	90.098	89.961	**90.256**
Segment	95.844	96.017	96.104	96.277	96.329	95.983	**96.476**
Vehicle	**80.567**	79.551	78.842	79.362	79.433	79.102	79.905
Waveform	84.152	86.436	**86.636**	**86.636**	**86.636**	**86.636**	86.276
Wine	95.955	96.404	96.404	96.292	96.404	96.404	**96.742**
WineRed	57.949	57.423	57.711	57.761	57.811	57.386	**58.111**
WineWhite	52.748	52.523	54.332	54.075	54.067	53.050	**55.039**
Yeast	56.631	56.536	57.615	57.655	57.251	57.049	**57.803**
Zoo	92.875	92.282	**93.267**	92.875	92.678	93.071	92.875

Table 3: Accuracy using the six compared methods and different base classifiers for each sub-problem

4.1. Motivation through NOV@

In the previous experiment the results obtained by A&O and OVA methods were lower than expected. Rifkin and Klautau [41] showed the strength of OVA when the classifier was well-tuned and we consider that selecting the best base classifier for each sub-problem is a good way to tune the sub-problems. Consequently we have made an analysis in order to find the reason for these low results.

4.1.1. The strength of OVA

In order to analyse the behaviour of OVA, we have carried out a new experiment. Firstly, we have seen how OVA works by only taking into account the cases where there is only one sub-problem that returns a positive-class prediction. In other words, there is only one case among all the sub-problems in which one class is selected instead of the rest of the classes.

In Table 4 it can be seen the percentage of the cases where OVA takes the final decision under the aforementioned circumstances and which is the accuracy obtained. The results show that a large amount of cases are classified under these circumstances, obtaining high accuracy. Therefore, we deduce that OVA fails with the remaining cases, so they are analysed.

4.1.2. The weakness of OVA

After the analysis, we have deduced that the reason for the bad results of OVA and A&O is because the type of classifier used in each sub-problem has a big influence on the final decision. OVA and A&O obtain each class confidence level taking into account only one sub-problem, and in each sub-problem different base classifiers are applied. Each type of classifier uses a different methodology to calculate the confidence level, hence, these confidence levels have different meanings. Some classifiers tend to distribute the confidence level among the classes more equally

	Accuracy	Percentage
Abalone	40.146	0.656
Annealing	99.070	97.996
Arrhythmia	79.207	71.593
Balance	95.746	89.327
Car	98.024	94.005
Cmc	63.399	51.677
Dermatology	98.401	95.574
Ecoli	87.537	88.691
Flare	82.237	47.636
Glass	74.842	69.626
Iris	95.830	99.067
Nursery	99.818	98.307
Optdigits	98.545	99.797
Page-blocks	97.809	97.552
Pendigits	99.393	99.556
Satimage	90.873	97.725
Segment	98.240	94.762
Vehicle	87.301	76.809
Waveform	89.241	83.200
Wine	96.901	96.854
WineRed	63.311	69.206
WineWhite	58.871	65.039
Yeast	67.274	59.973
Zoo	97.613	91.287

Table 4: Accuracy and percentage of the cases where OVA takes the final decision when in only one sub-problem the class is selected instead of the rest of the classes

	C_1		C_2	
Sub-problem	θ_i	All	θ_i	All
$\theta_1 - vs - All$	0.33	0.67	0.17	0.83
$\theta_2 - vs - All$	0.20	0.80	0.11	0.89
$\theta_3 - vs - All$	0.07	0.93	0.05	0.95

Table 5: Confidence levels obtained in OVA for each sub-problem for C_1 and C_2 classifiers

than others. As a consequence, in the cases where all the output of OVA is negative, the classes obtained using these classifiers are more likely to be selected. We will try to clarify this problem with the following example:

Example: Let us consider a 3-class problem $\{\theta_1, \theta_2, \theta_3\}$ and a new case to be classified. The confidence levels obtained for each OVA sub-problem for classifiers C_1 and C_2 can be seen in Table 5.

It can be observed in Table 5 that in both classifiers, θ_1 obtains the highest confidence levels; as a consequence, θ_1 should be assigned to the new case. Moreover, it is possible to observe that all the classes obtain higher confidence levels with C_1 than with C_2. So, let us consider that after the validation phase, C_2 is selected for $\theta_1 - vs - all$ and C_1 for $\theta_2 - vs - all$ and for $\theta_3 - vs - all$. In this case, θ_2 is the class with the highest confidence level (0.20), so θ_2 is assigned to the new case.

In this example, it can be seen that the classes classified with C_1 have higher probability to be selected than the classes classified with C_2. Hence, the final decision could be different, depending on the type of classifier selected in each sub-problem. That is why we consider it is not fair to compare the confidence levels among them.

In order to avoid this problem, the accuracy obtained for each classifier of the sub-problems in the validation phase, could be used as a confidence level. But we do not consider this appropriate because the accuracy depends on how the classes are distributed. The best differentiated classes are more likely to be selected, because the sub-problems where

they take part obtain a higher accuracy in the validation phase.

4.2. New One Versus$^{All}_{One}$ (NOV@)

We have shown that it is not a sound alternative to depend on the confidence levels when different base classifiers are used, moreover we show the strength of OVA in the case that among the sub-problems in only one of them one class outperforms the rest. Furthermore, previously was shown that OVO is able to correct some errors made by OVA.

Considering these three facts, we propose a new version of OVA where the confidence level is not taken into account; instead the majority vote is used. As previously mentioned, the problem of the majority vote in OVA is that it is common for there to be ties. In this case, the ties are broken by applying OVO only taking into account the tie-classes. We have denoted this new method as *New One Versus*$^{All}_{One}$ (NOV@). When a new case to be classified arrives there are 3 possibilities:

- If only one of the sub-problems gives a positive result, we consider that it is sufficiently reliable, therefore NOV@ returns this class.
- If in more than one case a positive result is obtained, then there is a tie. Hence, Max-Wins is applied only taking into account the tie classes.
- If all the sub-problems obtain a negative result, we consider that OVA has not enough reliability to take the final decision, so Max-Wins is applied with all the classes.

With this new algorithm our aim is to improve OVA's performance. Moreover since the majority of instances are classified applying OVA (Table 4), the new algorithm reduces the number of sub-problems of OVO.

5. Experimental Results

We run this new method with the same characteristics as run in the previous experiments (Section 3.2). Table 6 shows the obtained results. It can be seen that NOV@ obtains the best result in 15 of the databases, whereas OVA+OVO obtains the best result in 5. Moreover NOV@ obtains the best mean followed by OVA+OVO.

We have shown in the previous section (Section 4.1) when different base classifiers are being used, care must be taken when strategies that depend on the confidence level are used. However WV and PC also depend on the confidence levels of the sub-problems and, oddly, WV and PC are the state-of-the-art algorithms that obtain the best mean results. The difference between A&O and OVA with WV and PC is that A&O and OVA obtain the confidence level of each class only taking into account one sub-problem, while WV and PC take into account the confidence levels of the different sub-problems. This leads us to think that this combination tends to compensate the confidence levels.

In order to obtain a meaningful conclusions, we carry out statistical analysis to find whether significant differences among the results obtained exists or not. According to [16], we have used the Iman-Davenport test to detect statistical differences among the different strategies. This test rejects the null hypothesis of equivalence between algorithms since p-value (0.0001) is lower than our α-value (0.1). Thus, we have applied Shaffer post-hoc test in order to find out which algorithms are distinctive among them. Table 7 shows the most relevant results of the test, where "+" symbol implies that the first algorithm is statistically better than the confronting one, whereas "=" means that there are not significant differences between the compared algorithms. The method having the best performance is NOV@, closely followed by OVA+OVO. Both methods significantly improve all the remaining strategies, except WV. However, it can be seen that our proposed methods obtain more robust results since WV only outperforms significantly OVA and A&O. Moreover if we compare the rank of NOV@, OVA+OVO and WV, our two methods obtain more stable results.

5.1. Computational Load

In order to measure the computational cost and complexity of our proposals, in Tables 8, 9 and 10 we show the training and testing times and the number of binary classifiers used in each strategy.

Table 8 shows the training time of each strategy for the different databases. It is observed that the strategies are divided into three groups: OVA, OVO aggregations(Max-Wins, WV, PC and DDAG) and combinations of OVA and

Database	OVA	A&O	Max-Wins	WV	PC	DDAG	OVA+OVO	NOV@
Abalone	16.553	22.993	**26.598**	26.589	26.426	25.171	**26.598**	26.560
Annealing	98.062	97.929	98.196	98.196	98.129	98.085	98.307	**98.396**
Arrhythmia	63.850	65.265	69.292	69.292	71.150	68.407	71.018	**72.124**
Balance	90.128	89.840	90.096	90.929	90.865	90.641	**91.058**	90.994
Car	96.273	95.914	95.787	95.984	95.741	95.845	96.296	**96.563**
Cmc	48.350	49.220	52.831	52.492	52.465	51.840	53.646	**53.863**
Dermatology	95.519	97.158	97.541	**97.596**	**97.596**	97.541	97.432	97.377
Ecoli	82.202	83.452	83.095	83.988	83.869	83.631	83.810	**84.643**
Flare	54.540	58.668	**60.094**	59.756	59.587	59.606	**60.094**	59.794
Glass	64.206	63.458	64.112	64.486	65.327	63.458	65.140	**67.009**
Iris	95.333	94.800	94.800	94.800	94.800	94.800	94.933	**95.467**
Nursery	99.336	99.384	99.451	99.477	99.452	99.431	99.653	**99.671**
Optdigits	98.456	**98.480**	98.466	98.463	98.470	98.395	98.473	98.452
Page-blocks	96.707	96.799	96.751	96.828	96.667	96.777	96.897	**97.003**
Pendigits	**99.250**	99.221	99.210	99.214	99.207	99.143	99.238	99.221
Satimage	89.958	90.126	90.051	90.058	90.098	89.961	**90.256**	90.030
Segment	95.844	96.017	96.104	96.277	96.329	95.983	96.476	**96.667**
Vehicle	80.567	79.551	78.842	79.362	79.433	79.102	79.905	**80.922**
Waveform	84.152	86.436	**86.636**	**86.636**	**86.636**	**86.636**	86.276	85.948
Wine	95.955	96.404	96.404	96.292	96.404	96.404	**96.742**	96.629
WineRed	57.949	57.423	57.711	57.761	57.811	57.386	58.111	**58.487**
WineWhite	52.748	52.523	54.332	54.075	54.067	53.050	55.039	**55.300**
Yeast	56.631	56.536	57.615	57.655	57.251	57.049	57.803	**58.679**
Zoo	92.875	92.282	**93.267**	92.875	92.678	93.071	92.875	**93.267**
Mean	79.394	79.995	80.720	80.795	80.852	80.475	81.087	**81.378**
Rank	6.2	6.0	4.6	4.0	4.5	5.8	2.5	**2.3**

Table 6: Accuracy using the seven compared methods and different base classifiers for each sub-problem

OVO (A&O, OVA+OVO and NOV@). It can be seen that the combinations of OVA and OVO need more training time. On the other hand, the classification time needed by OVA is slightly longer than OVO aggregations for problems with few classes. Although OVA uses less sub-problems than OVO, the size of the sub-problems is higher in OVA and that is why the time required to train the classifier is longer. However, it can be noted that when the number of classes is high, the training time of OVA is shorter than in OVO.

Table 9 shows the testing time of each strategy for the different databases. It can be seen that NOV@ tends to be one of the fastest among the 8 methods. Only DDAG and OVA tend to be faster closely followed by A&O. Moreover except for in Abalone database, NOV@ spends a bit more time than OVA. Immediately below them on the table are OVO and WV that need similar time, while PC needs slightly more time since it follows an iterative procedure. Finally, OVA+OVO tends to be the slowest method.

Table 10 shows the mean number of classifiers needed in each strategy for the different databases. The obtained results tend to be similar to those found in Table 9. However, this time NOV@ needs slightly more classifiers than A&O.

Viewing the results obtained in Tables 8, 9 and 10, although NOV@ is one of the slowest strategies to train, it is one of the fastest strategies at classification time, only outperformed by OVA and DDAG. On the other hand, OVA+OVO is the slowest strategy.

Hypothesis	p-value
NOV@ vs OVA	+(1.1904E-006)
OVA+OVO vs OVA	+(3.2931E-006)
NOV@ vs A&O	+(5.2983E-006)
NOV@ vs DDAG	+(1.5605E-005)
OVA+OVO vs A&O	+(1.8148E-005)
OVA+OVO vs DDAG	+(5.0998E-005)
NOV@ vs Max-Wins	+(0.0250)
WV vs OVA	+(0.0376)
NOV@ vs PC	+(0.0376)
OVA+OVO vs Max-Wins	+(0.0424)
OVA+OVO vs PC	+(0.0682)
WV vs A&O	+(0.0820)
WV vs DDAG	=(0.1524)
PC vs OVA	=(0.2552)
NOV@ vs WV	=(0.2552)
Max-Wins vs OVA	=(0.3269)
OVA+OVO vs WV	=(0.4067)
PC vs A&O	=(0.4309)
Max-Wins vs A&O	=(0.5548)
PC vs DDAG	=(0.5709)
Max-Wins vs DDAG	=(0.6998)

Table 7: Shaffer test

6. Conclusion

This paper has presented a new approach to combine pairwise classifiers which aims to improve classification accuracy in supervised classification multi-class problems. Starting from a single combination of two well-known approaches (One-vs-All and One-vs-One), a new procedure to make a classifier combination is presented, NOV@, in which both OVA and OVO are combined in a different way than found in the rest of literature. The results obtained by the new approach on different datasets are subjected to in-depth analyses and compared with those of the most used state-of-the-art methods. From the comparison, it is shown that the results are very competitive, ranking in the first position from the accuracy point of view, and among the best in classification time; this last due to a low number of classifiers.

It has also been shown that OVA and OVO strategies are compatible and can be combined with each other, even when different base classifiers are used for each sub-problem. This is possible because each sub-problem has been tackled as an independent one, and hence it is treated as a new classification problem in which two classes are to be discriminated. The proposed methods –the single one, OVA+OVO, and NOV@– have been implemented and tested over 20 databases from the UCI repository, obtaining significant improvements over other state-of-the-art strategies. In addition to this, the two methods maintain the simplicity that has made of OVA and OVO the most used class-binarization methods. Furthermore, comparing our methods with other state-of-the-art algorithms, we have made an empirical study in order to analyse how different class-binarization strategies work when different base classifiers are applied in each sub-problem.

A further analysis of the computational load of the used approaches has shown that the proposed approach –NOV@– has competitive classification times compared with the state-of-the-art approaches, and that it has a lower computational cost with respect to the most powerful classifiers. When training time is compared, though, the results were worse than those of other approaches. However, this was expected given that all the sub-problems decomposed by OVA and OVO have to be trained.

As future works, we are planning to apply more single classifiers in the best classifier selection phase as well as other approaches to combine the different results. In this sense, the method proposed by K. Polat [36] seems to set the right direction for future experiments.

On the other hand, real applications of the proposed model are to be analysed. We are going to test the approach with real problems related data, for example we are trying to obtain new data related to our previous work [11] in order to carry on with the experiments. Other application of the proposed approach can be website phishing, which is considered one of the crucial security challenges for the online community due to the massive numbers of online transactions performed on a daily basis. We are also planning to perform an experiment similar to the one proposed by Abdelhamid et al. [1].

Database	OVA	A&O	OVO	WV	PC	DDAG	OVA+OVO	NOV@
Abalone	752,004	2,00,0680	1,248,676	1,248,676	1,248,676	1,248,676	2,000,680	2,000,680
Annealing	168,841	304,570	135,729	135,729	135,729	135,729	304,570	304,570
Arrhythmia	5,401,826	10,538,305	5,136,479	5,136,479	5,136,479	5,136,479	10,538,305	10,538,305
Balance	3,782	6,611	2,829	2,829	2,829	2,829	6,611	6,611
Car	12,948	22,779	9,831	9,831	9,831	9,831	22,779	22,779
Cmc	15,569	25,635	10,066	10,066	10,066	10,066	25,635	25,635
Dermatology	46,394	86,553	40,159	40,159	40,159	40,159	86,553	86,553
Ecoli	6,182	13,078	6,896	6,896	6,896	6,896	13,078	13,078
Flare	34,020	63,916	29,896	29,896	29,896	29,896	63,916	63,916
Glass	3,889	7,656	3,767	3,767	3,767	3,767	7,656	7,656
Iris	777	1,362	585	585	585	585	1,362	1,362
Nursery	516,714	857,053	340,339	340,339	340,339	340,339	857,053	857,053
Optdigits	3,734,410	7,035,312	3,300,902	3,300,902	3,300,902	3,300,902	7,035,312	7,035,312
Page-blocks	286,433	534,925	248,492	248,492	248,492	248,492	534,925	534,925
Pendigits	1,731,714	3,190,942	1,459,228	1,459,228	1,459,228	1,459,228	3,190,942	3,190,942
Satimage	1,622,867	2,789,917	1,167,050	1,167,050	1,167,050	1,167,050	2,789,917	2,789,917
Segment	203,956	371,821	167,865	167,865	167,865	167,865	371,821	371,821
Vehicle	33,051	57,489	24,438	24,438	24,438	24,438	57,489	57,489
Waveform	298,047	470,827	172,780	172,780	172,780	172,780	470,827	470,827
Wine	2,563	4,298	1,735	1,735	1,735	1,735	4,298	4,298
WineRed	48,824	91,505	42,681	42,681	42,681	42,681	91,505	91,505
WineWhite	323,583	591,057	267,474	267,474	267,474	267,474	591,057	591,057
Yeast	51,559	103,720	52,161	52,161	52,161	52,161	103,720	103,720
Zoo	3,501	7,313	3,812	3,812	3,812	3,812	7,313	7,313
Mean	637,643.92	1,215,721.83	578,077.92	578,077.92	578,077.92	578,077.92	1,215,721.83	1,215,721.83

Table 8: Comparison of the training time of 8 methods (in milliseconds)

Acknowledgements

The work described in this paper was partially conducted within the Basque Government Research Team grant and the University of the Basque Country UPV/EHU and under grant UFI11/45 (BAILab). I. Mendialdua holds a grant from Basque Government.

References

[1] Abdelhamid, N., Ayesh, A., Thabtah, F., 2014. Phishing detection based associative classification data mining. Expert Systems with Applications.

[2] Aha, D. W., Kibler, D., Albert, M. K., 1991. Instance-based learning algorithms. Machine learning 6 (1), 37–66.

[3] Anand, R., Mehrotra, K., Mohan, C. K., Ranka, S., 1995. Efficient classification for multiclass problems using modular neural networks. Neural Networks, IEEE Transactions on 6 (1), 117–124.

[4] Bache, K., Lichman, M., 2013. Uci machine learning repository. Irvine, CA: University of California, School of Information and Computer Science.

[5] Bagheri, M. A., Gao, Q., Escalera, S., 2012. Efficient pairwise classification using local cross off strategy. In: Advances in Artificial Intelligence. Springer, pp. 25–36.

[6] Chen, J., Wang, C., Wang, R., 2009. Adaptive binary tree for fast svm multiclass classification. Neurocomputing 72 (13), 3370–3375.

[7] Cohen, W. W., 1995. Fast effective rule induction. In: In Proceedings of the Twelfth International Conference on Machine Learning. Morgan Kaufmann, pp. 115–123.

[8] Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research 7, 1–30.

[9] Dietterich, T. G., Bakiri, G., 1995. Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research 2.

[10] Fei, B., Liu, J., 2006. Binary tree of svm: a new fast multiclass training and classification algorithm. Neural Networks, IEEE Transactions on 17 (3), 696–704.

[11] Ferreiro, S., Arnaiz, A., Sierra, B., Irigoien, I., 2012. Application of bayesian networks in prognostics for a new integrated vehicle health management concept. Expert Systems with Applications 39 (7), 6402–6418.

[12] Friedman, J., 1996. Another approach to polychotomous classifcation. Tech. rep., Technical report, Stanford University, Department of Statistics.

[13] Fürnkranz, J., 2002. Round robin classification. The Journal of Machine Learning Research 2, 721–747.

[14] Galar, M., Fernández, A., Barrenechea, E., Bustince, H., Herrera, F., 2011. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. Pattern Recognition 44 (8), 1761–1776.

[15] Galar, M., Fernández, A., Barrenechea, E., Bustince, H., Herrera, F., 2013. Dynamic classifier selection for one-vs-one strategy: Avoiding non-competent classifiers. Pattern Recognition 46 (12), 3412 – 3424.

[16] García, S., Fernández, A., Luengo, J., Herrera, F., 2010. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Information Sciences 180 (10), 2044–2064.

Database	OVA	A&O	OVO	WV	PC	DDAG	OVA+OVO	NOV@
Abalone	2,168	108,513	25,320	29,089	167,479	2,950	25,897	132,295
Annealing	570	670	1065	1068	1225	437	1633	579
Arrhythmia	5,886	6,354	19,193	19,255	20,736	3,806	25,061	9,873
Balance	24	44	34	35	127	42	50	24
Car	48	94	352	357	547	220	399	68
Cmc	28	58	51	51	140	31	76	43
Dermatology	256	394	338	342	440	110	588	257
Ecoli	182	234	289	294	484	83	470	203
Flare	139	275	386	392	676	125	498	315
Glass	35	44	45	47	103	25	77	46
Iris	10	17	12	12	55	14	21	12
Nursery	804	1,023	1,379	1,409	3,686	616	2162	817
Optdigits	579,849	1,204,315	646,695	646,861	652,546	107,932	1,226,422	581,075
Page-blocks	1,805	2,030	3,705	3,718	4,675	2,971	5,503	1,856
Pendigits	403,145	673,507	556,751	557,102	568,207	84,669	959,782	404,812
Satimage	149,632	241,427	138,126	138,159	139,908	51,400	287,481	151,562
Segment	2,248	2,381	3,312	3,331	4,233	1,046	5,546	2,322
Vehicle	71	98	129	129	219	64	198	85
Waveform	372	567	309	317	823	225	661	397
Wine	39	57	40	41	54	22	76	35
WineRed	149	230	513	531	1,186	236	656	314
WineWhite	5,148	5,322	5,521	5,563	7,524	1,658	10,650	6,971
Yeast	1,028	1,082	1,859	1,899	3,386	478	2,871	1,892
Zoo	46	50	105	108	149	23	148	48
Mean	48,070.08	93,699.42	58,563.71	58,754.58	65,775.33	10,799.29	106,538.58	53,995.88

Table 9: Comparison of the testing time of 8 methods (in milliseconds)

[17] García-Pedrajas, N., Ortiz-Boyer, D., 2006. Improving multiclass pattern recognition by the combination of two strategies. Pattern Analysis and Machine Intelligence, IEEE Transactions on 28 (6), 1001–1006.

[18] García-Pedrajas, N., Ortiz-Boyer, D., 2011. An empirical study of binary classifier fusion methods for multiclass classification. Information Fusion 12 (2), 111–130.

[19] Ghaffari, H. R., Yazdi, H. S., 2013. Multiclass classifier based on boundary complexity. Neural Computing and Applications, 1–9.

[20] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H., 2009. The weka data mining software: an update. ACM SIGKDD Explorations Newsletter 11 (1), 10–18.

[21] Hansen, L. K., Salamon, P., 1990. Neural network ensembles. Pattern Analysis and Machine Intelligence, IEEE Transactions on 12 (10), 993–1001.

[22] Hastie, T., Tibshirani, R., 1998. Classification by pairwise coupling. The annals of statistics 26 (2), 451–471.

[23] Hsu, C.-W., Lin, C.-J., 2002. A comparison of methods for multiclass support vector machines. Neural Networks, IEEE Transactions on 13 (2), 415–425.

[24] John, G. H., Langley, P., 1995. Estimating continuous distributions in bayesian classifiers. In: Proceedings of the Eleventh conference on Uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc., pp. 338–345.

[25] Ko, J., Byun, H., 2003. Binary classifier fusion based on the basic decomposition methods. In: Proceedings of the 4th international conference on Multiple classifier systems. Springer, pp. 146–155.

[26] Kuncheva, L. I., Whitaker, C. J., 2003. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Machine learning 51 (2), 181–207.

[27] Lebrun, G., Lezoray, O., Charrier, C., Cardot, H., 2007. An ea multi-model selection for svm multiclass schemes. In: Proceedings of the 9th international work conference on Artificial neural networks (IWANN07). Springer, pp. 260–267.

[28] Liepert, M., 2003. Topological fields chunking for german with svm's: Optimizing svm-parameters with ga's. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing.

[29] Liu, B., Hao, Z., Yang, X., 2007. Nesting algorithm for multi-classification problems. Soft Computing 11 (4), 383–389.

[30] Lorena, A. C., de Carvalho, A. C., 2010. Building binary-tree-based multiclass classifiers using separability measures. Neurocomputing 73 (16–18), 2837 – 2845.

[31] Lorena, A. C., de Carvalho, A. C., Gama, J. M., 2008. A review on the combination of binary classifiers in multiclass problems. Artificial Intelligence Review 30 (1-4), 19–37.

[32] Madzarov, G., Gjorgjevikj, D., 2009. Multi-class classification using support vector machines in decision tree architecture. In: EUROCON 2009, EUROCON'09. IEEE. IEEE, pp. 288–295.

[33] Moreira, M., Mayoraz, E., 1998. Improved pairwise coupling classification with correcting classifiers. In: Proceedings of the 10th European Conference on Machine Learning (ECML-98). Springer, pp. 160–171.

[34] Platt, J. C., 1999. Fast training of support vector machines using sequential minimal optimization. In: Schölkopf, B., Burges, C. J. C., Smola, A. J. (Eds.), Advances in kernel methods. MIT Press, pp. 185–208.

[35] Platt, J. C., Cristianini, N., Shawe-Taylor, J., 2000. Large margin dags for multiclass classification. Advances in neural information processing systems 12 (3), 547–553.

[36] Polat, K., 2013. Data weighting method on the basis of binary encoded output to solve multi-class pattern classification problems. Expert Systems with Applications 40 (11), 4637–4647.

[37] Polat, K., Güneş, S., 2009. A novel hybrid intelligent method based on c4.5 decision tree classifier and one-against-all approach for multi-class

Database	OVA	A&O	OVO	WV	PC	DDAG	OVA+OVO	NOV@
Abalone	28	320.153	378	378	378	27	406	389.936
Annealing	5	6	10	10	10	4	15	5.119
Arrhythmia	13	14.221	78	78	78	12	91	27.596
Balance	3	4.314	3	3	3	2	6	3.298
Car	4	5.063	6	6	6	3	10	4.250
Cmc	3	4.015	3	3	3	2	6	4.386
Dermatology	6	11.986	15	15	15	5	21	6.404
Ecoli	8	11.390	28	28	28	7	36	9.746
Flare	6	11.003	15	15	15	5	21	13.798
Glass	6	7.133	15	15	15	5	21	9.222
Iris	3	4.317	3	3	3	2	6	3.015
Nursery	5	6.013	10	10	10	4	15	5.109
Optdigits	10	53.345	45	45	45	9	55	10.090
Page-blocks	5	6.001	10	10	10	4	15	5.147
Pendigits	10	32.019	45	45	45	9	55	10.147
Satimage	6	15.733	15	15	15	5	21	6.248
Segment	7	8.085	21	21	21	6	28	7.551
Vehicle	4	5.001	6	6	6	3	10	4.792
Waveform	3	4.518	3	3	3	2	6	3.303
Wine	3	4.825	3	3	3	2	6	3.056
WineRed	6	7.802	15	15	15	5	21	9.574
WineWhite	7	8	21	21	21	6	28	12.795
Yeast	10	11	45	45	45	9	55	27.313
Zoo	7	9.277	21	21	21	6	28	8.570
Mean	7.052	23.801	33.288	33.288	33.288	5.833	40.288	24.603

Table 10: Number of classifiers used by different methods

classification problems. Expert Systems with Applications 36 (2, Part 1), 1587 – 1592.

[38] Pujol, O., Radeva, P., Vitria, J., Jun. 2006. Discriminant ecoc: A heuristic method for application dependent design of error correcting output codes. IEEE Trans. Pattern Anal. Mach. Intell. 28 (6), 1007–1012.

[39] Quinlan, J. R., 1993. C4. 5: programs for machine learning. Vol. 1. Morgan kaufmann.

[40] Reid, S. R., 2010. Model combination in multiclass classification. Ph.D. thesis, University of Colorado.

[41] Rifkin, R., Klautau, A., 2004. In defense of one-vs-all classification. The Journal of Machine Learning Research 5, 101–141.

[42] Rumelhart, D. E., Hinton, G. E., Williams, R. J., 1985. Learning internal representations by error propagation. Tech. rep., DTIC Document.

[43] Szepannek, G., Bischl, B., Weihs, C., 2009. On the combination of locally optimal pairwise classifiers. Engineering Applications of Artificial Intelligence 22 (1), 79–85.

[44] Tang, E., Suganthan, P., Yao, X., 2006. An analysis of diversity measures. Machine Learning 65 (1), 247–271.