Dynamic selection of the best base classifier in One versus One

I. Mendialdua[a,*], J. M. Martínez-Otzeta[a], I. Rodriguez[a], T. Ruiz-Vazquez[b], B. Sierra[a]

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

Abstract

Class Binarization strategies decompose the original multi-class problem into several binary sub-problems. One versus One (OVO) is one of the most popular Class Binarization techniques, which considers every pair of classes as a different sub-problem. Usually, the same classifier is applied to every sub-problem and then all the outputs are combined by some voting scheme. In this paper we present a novel idea where for each test instance we try to assign the best classifier in each sub-problem of OVO. To do so, we have used two simple Dynamic Classifier Selection (DCS) strategies that have not been used in this context. The two DCS strategies use K-NN to obtain the local region of the test-instance, and the classifier that performs the best for those instances in the local region, is selected to classify the new test instance. The difference between the two DCS strategies remains in the weight of the instance. In this paper we also have proposed a novel approach in those DCS strategies. Instead of using the K-NN method to achieve the local regions, we propose to use a version of K-NN obtained from the state-of-the-art called K-NN Equality (K-NNE). K-NNE is similar to K-NN, but it obtains the K nearest neighbors of each class. We have carried out an empirical study over several UCI databases, which shows the robustness of our proposal.

Keywords: Machine Learning, Supervised Classification, Decomposition Strategies, One against One, Classifier Combination, Dynamic Classifier Selection

1. Introduction

The objective of the Supervised Classification strategies is to classify the new unlabelled samples in their correct class. To do so, these strategies create a prediction model (also denoted as classifier) based on a training set of well labelled instances.

A classification problem with only two classes is known as a binary classification problem. A simple example of a binary classification problem are the yes/no or true/false problems. On the other hand the problems with more than two classes are known as multi class problems. However for several kind of classifiers, such as SVM, it is easier to build a classifier to distinguish only between two classes. Because of that, two general approaches have been adopted to deal with multi class problems: to create a single decision function that considers all the classes or to decompose the problem into several binary sub-problems (also known as class-binarization).

In the latest years the class-binarization strategies are getting more common in the literature. There are 3 main techniques: One versus All (OVA)[2], One versus One (OVO)[12] and Error Correcting Output Codes (ECOC)[9]. In this work we focus our attention on OVO strategy, which compares the cases belonging to two classes in each sub-problem; the remaining classes are ignored in each sub-problem.

[*]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)

OVO gives the option to consider each sub-problem as independent and to select a different base classifier in each sub-problem, which could be considered as an example of static classifier selection problem. For classification selection scheme two categories exist: static and dynamic. In the first case, regions of competence are defined during the training phase, while in the second case, they are defined during the classification phase taking into account the characteristics of the sample to be classified.

In the literature it is possible to find several works that propose the selection of different base classifiers in each sub-problem statically; however conclusions of these works are contradictory: some works obtain significant improvements, while others reject this hypothesis.

In this paper, we propose to extend this idea trying to assign dynamically the best base classifier in each sub-problem of OVO. We have called to this new approach DYNOVO. We present several variations of DYNOVO using two simple Dynamic Classifier Selection (DCS) strategies from the state-of-the-art. Those strategies select the classifier that obtains the best accuracy in a local region, which is defined by the K-Nearest Neighbor (K-NN) algorithm. In order to adapt those DCS strategies we have made several changes on the K-NN algorithm, moreover we propose the use of another K-NN version called K-Nearest Neighbor Equality (K-NNE) from the state-of-the-art which fits properly in this problem. For our experiments we have chosen several well-known classifier from the Machine Learning paradigms: SVM, C4.5, Ripper, Naive Bayes and Bayesian Network. We have carried out our experiments over 22 UCI databases. Experimental results show that DYNOVO obtains very good results.

The rest of the paper is organized as follows: In Section 2 we review the Class Binarization techniques, focusing on OVO strategy. In Section 3 we review the Dynamic Classifier Selection technique while Section 4 is devoted to related work. Section 5 describes the proposed approach and Section 6 shows the experimental results obtained. Finally, Section 7 states the conclusions of our work and future research lines.

2. Class Binarization

Several machine learning techniques, such as SVM, were designed to solve two-class problems. However many real-word problems involve the discrimination of more than two classes. In order to use those algorithms in multi-class problem the class binarization strategies divide the original problem into several two-class problems. It has been proven the benefits to use the binarization techniques in multi-class problems [15] and due to those promising results the use of these strategies has been extended to other base classifiers, such as Ripper [14] or C4.5 [9]. In the recent years the class binarization strategies are receiving more attention in the literature, and one indicative of that is that recently several reviews have been published [29] [18] [15].

The Class Binarization techniques are divided by two steps: decomposition and combination.

In the decomposition step, the multi-class problem is decomposed into several binary sub-problems. The most popular strategies consist on grouping classes into two groups in each sub-problem, in this way each binary classifier compares two groups of classes between them. The code-matrix is an easy way to represent how the classes are grouped.

In the code matrix each class takes values in the set of {+1, -1, 0}, where +1 indicates that the class is associated to the positive class, -1 indicates that the class is associated to the negative class and 0 indicates that the class is ignored in this binary sub-problem. In Figure 1 an example of a code matrix can be seen; it shows how a 5-class problem $\{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5\}$ is decomposed into a 6 binary sub-problems $\{f_1, f_2, f_3, f_4, f_5, f_6\}$. For instance, it can be seen that in the sub-problem f_1, the classifier is constructed in such manner that the cases belonging to θ_1 and θ_2 are grouped in class +1 and the cases of θ_3 and θ_5 in class -1. So this classifier compares θ_1 and θ_2 classes with θ_3 and θ_5, whereas the cases that belong to θ_4 are ignored.

Each of these sub-problems returns an output with a prediction. The combination step consists on combining these predictions to made the final decision. The simplest combination is the majority vote, where each sub-problem returns a vote and the class with the largest number of votes is predicted.

Different decomposition strategies have been developed where One Vs One (OVO) is one of the strategies that has received more attention in the literature.

2.1. One versus One (OVO)

OVO decomposition scheme decomposes a K class multiclass problem into a $K(K-1)/2$ sub-problems. Each sub-problem is responsible to differentiate one pair of classes (θ_i, θ_j), where $\theta_i \neq \theta_j$; the remaining classes are ignored.

$$
\begin{array}{ccc}
 & & \overbrace{\begin{array}{cccccc} f_1 & f_2 & f_3 & f_4 & f_5 & f_6 \end{array}}^{classifiers} \\
classes & \left\{ \begin{array}{c} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \end{array} \right. & \begin{pmatrix} +1 & 0 & -1 & -1 & 0 & +1 \\ +1 & +1 & -1 & -1 & +1 & 0 \\ -1 & +1 & +1 & -1 & 0 & 0 \\ 0 & -1 & 0 & +1 & 0 & +1 \\ -1 & -1 & 0 & -1 & -1 & -1 \end{pmatrix}
\end{array}
\qquad
\begin{array}{l}
f_1 \rightarrow \theta_1, \theta_2 \; vs \; \theta_3, \theta_5 \\
f_2 \rightarrow \theta_2, \theta_3 \; vs \; \theta_4, \theta_5 \\
f_3 \rightarrow \theta_3 \; vs \; \theta_1, \theta_2 \\
f_4 \rightarrow \theta_4 \; vs \; \theta_1, \theta_2, \theta_3, \theta_5 \\
f_5 \rightarrow \theta_2 \; vs \; \theta_5 \\
f_6 \rightarrow \theta_1, \theta_4 \; vs \; \theta_5
\end{array}
$$

Figure 1: Example of a code matrix

Figure 2 illustrates a code matrix of how a 5-class problem is decomposed in OVO: in each sub-problem one class is represented as +1 class, another one is represented as -1 and the remaining classes are represented as 0.

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

Figure 2: OVO code-matrix

There are different aggregations of combining the output predictions of the sub-problems. The simplest combination strategy is the majority vote [14] [12]. An immediate extension is the Weighted Voting, where the vote of each output is weighted based on the confidence level returned by the classifier [22]. Hastie and Tibshirani [21] propose another combination that tries to find the best approximation of the class posterior probabilities given the posterior probabilities of the pairwise sub-problems.

Although OVO requires a high number of sub-problems (specially when the number of classes is high), it is worth mentioning that each classifier is trained only with the samples from the corresponding pair of classes, hence the decision boundaries to distinguish the classes are simpler and the required time is not high. However there are several proposals that try to reduce the number of sub-problems, where most of these works are based on a hierarchical structure [32] [11].

3. Dynamic Classifier Selection (DCS)

As different classifiers usually make different error on different samples, Dynamic Classifier Selection (DCS) based methods attempt to predict the single classifier which is most likely to be correct for a given sample. To do so, the "best" classifier for each partition is determined on a validation process. For classification, an unknown sample is assigned to a partition, and the output of the best classifier for that partition is the one used to make the final decision.

The first Dynamic Classification approaches are introduced by Woods [39] and are based on K-NN algorithm. He proposes two methods: Overall Local Accuracy (OLA) and Local Class Accuracy: both methods obtain the classifiers' accuracy in local regions in the surroundings of the unknown test sample, the classifier with the best accuracy is selected to classify the unknown sample. Smith [36] proposes an immediate extension of OLA applying the Distance Weighted K-NN (DW-OLA). Giacinto and Roli [19] also extend Woods's work incorporating distance weighted and classifiers confidence levels to two new methods called A Priori and A Posteriori. On the other hand, there are also other works which are not based on the K-NN method, for instance, Liu and Yuan [28] propose to use clustering: they divide the feature space into several clusters for each base classifier. The unknown sample is assigned to a cluster for each base classifier, and the classifier of the most accurate cluster is selected to classify the unknown sample.

Recently, the DCS methods have been extended to Dynamic Ensemble Selection (DES): instead of finding the most suitable classifier, the most suitable ensemble for each sample is selected. Ko et al. [25] propose 4 new dynamic selection schemes. Those methods obtain the K nearest neighbors of the test point and the classifiers that classify correctly those neighbors, are used as ensemble to classify the test instance. On the other hand, Dos Santos et al. [10] introduce a two step DES method: in the first step, highly accurate candidates ensembles are selected; in the second step, among those ensembles, for each test sample, the ensemble with the largest confidence level is selected. In a further work Cavalin et al. [7] extend the previous work and they adapt it to Dynamic Multistage Organization strategy.

4. Related Works

In a classification problem, the classical way is to select the optimal base classifier for the database and all the sub-problems are classified with this classifier. As in binarization strategies there are too many sub-problems, it is possible that this base classifier could have difficulties to deal with all the sub-problems appeared, returning the wrong result in some of them. This raises the question - should the same base classifier be used on all sub-problems? or should sub-problems be tuned independently?

In the literature there are several works that treat the sub-problems independently. But to our knowledge excepting the introduced by Arruti et al [3] and Bautista et al [6] there are not more works that present an algorithm specifically for the cases that the sub-problems are treated independently. The majority of the approaches propose a method that try to select the best classifier or best parameters of the classifier for each sub-problem and they compare the new proposal with the results obtained without tuning.

On the one hand, some proposals focus on attempting to select the best base classifier in each sub-problem [38]. On the other hand, other approaches try to select the best hyper-parameters of SVM in each sub-problem. Because of the high number of possible values of the hyper-parameters, most of these works use evolutionary algorithms. Lebrun et al [26] and Liepert [27] propose the use of Genetic Algorithms while Souza et al [37] propose the use of Particle Swarm Optimization. The results obtained by these four works are contradictory since two of them consider that the independent tune of the sub-problems is better while the other two consider that there is no significant difference.

Lorena and Carvalho [30] consider that none of the mentioned works perform a rigorous statistical analysis. Thus, they investigate the use of Genetic Algorithms to automatically tune the parameters of each binary SVM. They conclude that the use of same parameter values in all binary SVM is sufficient to obtain good results.

In his Phd Thesis Reid [34] also deals with this problem and he concludes that it is better to tune the classifiers when the decision boundaries of sub-problems have different shape, otherwise, he concludes that it is better the same base classifier.

In the literature we have found an algorithm, proposed by Galar et al. [16] and Bagheri et al. [5] independently, that combine OVO with DCS strategies. Their main idea is to reduce the number of classifiers in OVO avoiding the no competent pairwise comparisons. The K nearest neighbors of a new instance are obtained and OVO is applied only considering those classes which are in the neighborhood.

On the other hand, there is also another work proposed by Kapp et al. [24] that selects the hyper-parameters of SVM dynamically. But this work does not use the DCS strategies and does not treat each sub-problem independently; it is oriented to data-streaming and similar problems. The authors consider that when knowledge about the environment is updated with new observations, the previously parametrized models need to be re-evaluated. To do so, they use the Particle Swarm Optimization.

5. Proposed approach: Dynamic Classifier Selection in OVO (DYNOVO)

Most of the works mentioned in Section 4 follow a similar procedure: they tune statically the classifier of each pairwise sub-problem. The hypothesis that the previous works follow is that the boundaries that distinguish the different sub-problems vary depending on the classes. We extend this idea and we consider that the shape of the boundaries between two classes can vary also, hence the use of the different base classifiers can be appropriate. Because of that, we propose a new method, called DYNOVO, that tries to select the best base classifier dynamically for each test pattern in each binary sub-problem: basically our method combines OVO with Dynamic Classifier Selection (DCS) strategies.

The structure of DYNOVO is similar to those most common DCS strategies and it is divided into two levels: validation and classification. The only difference is that the method is adapted to the pairwise decomposition strategy format.

The aim of the validation step is to see with which base classifier each training instance obtains correct or incorrect results by the different sub-problems. Each training sample is classified by different base classifiers for the different pairwise sub-problems. But instead of classifying it for every sub-problem, it is classified only in those sub-problems where the class the training sample belongs to is distinguished, since the remaining sub-problems can not return the correct result: if the training set belongs to class θ_i, the sub-problem that distinguish θ_j and θ_k ($\theta_i \neq \theta_j, \theta_i \neq \theta_k$) never will return the correct class. Hence, these sub-problems don't need to be treated and computational load is saved in the validation phase.

In the classification step, when an instance to be classified is arrived, our method tries to select the best base classifier for each sub-problem. To do so each sub-problem is treated independently. In each sub-problem the surrounding training samples of the new instance are obtained and the base classifier that obtains the best results for these instances in the validation phase is selected to classify it. In order to make this selection we have chosen the following DCS strategies:

- Overall Local Accuracy (OLA) [39]: When an instance to be classified is arrived, its surrounding region is selected obtaining its K-Nearest Neighbors. It calculates the local accuracy of each base classifier for these K neighbors. The base classifier with the highest local accuracy is selected to classify the test sample.

- Distance Weighted - Overall Local Accuracy (DW-OLA) [36]: This method is an immediate extension of OLA. When the local accuracy is calculated, each K neighbor receives a weight depending on their distance to the test sample, where the closer ones receive a higher weight.

Both strategies use K-NN method to delimit the local region. As we have mentioned previously, our approach has to be adapted to the OVO strategy. Because of that we have made a little change in the K-NN algorithm when the local region is obtained. Moreover we also propose to use a K-NN variation presented in the state-of-the-art called K-NN Equality (K-NNE) [35].

- K Nearest Neighbor (K-NN) [1]: K-NN is one of the most popular machine learning algorithms. When a new instance to be classified is arrived, the K most similar training instances are obtained and the most represented class among those K neighbors is assigned to the new instance. In order to measure the similarity, it is necessary to use a metric, being the euclidean distance one of the most common.

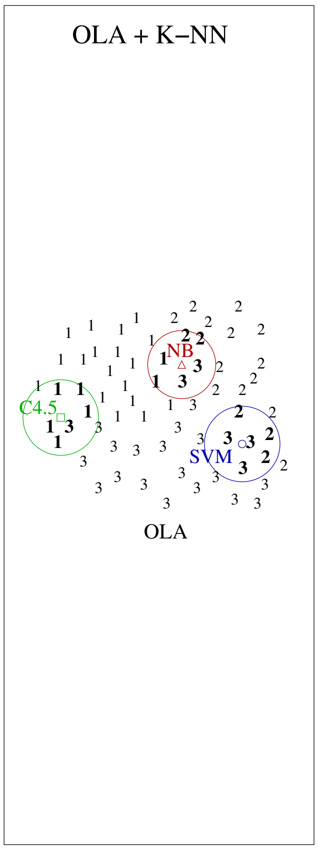
  It is worth mentioning that in this case K-NN is not used to classify a new unlabelled instance, but to delimit the local region of it. Our method tries to select the best base classifier for each sub-problem; because of that each sub-problem is treated independently. Therefore instead of taking into account all the training instances, for each sub-problem it only finds the K nearest neighbors of the test sample that belong to the classes of the sub-problem.

- K Nearest Neighbor Equality (K-NNE) [35]: K-NNE is an extension of K-NN in which the classes are treated independently: it searches in each class the K nearest neighbors and assigns the class whose K neighbors have the minimal mean distance to the sample test. In this way all the classes take part in the final decision.

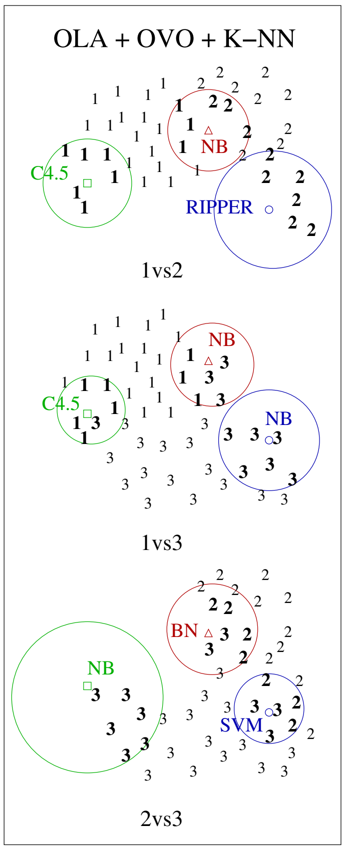
5.1. Example: obtaining the local regions

Figure 3 illustrates how the local-regions are obtained for 3 new cases in a 3-class problem with different strategies: OLA and our proposal applying K-NN and K-NNE. In the figures the 3 new cases are represented as □, △ and ∘. We want to emphasize that the local regions are not used to classify the unlabelled instances as in K-NN, indeed they are used to select the classifier which will be used to classify the unlabelled instances.

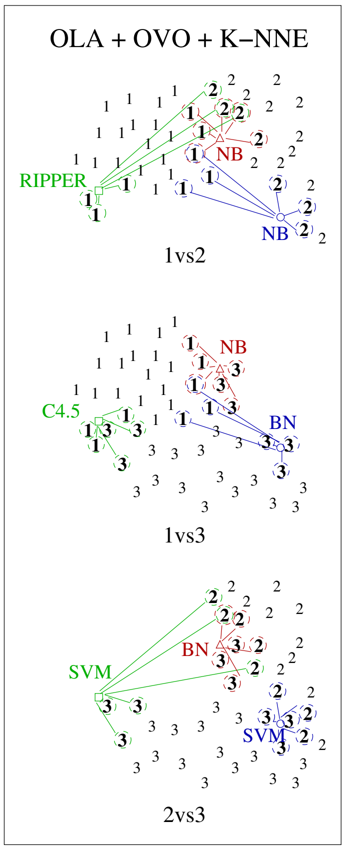
In Figure 3(a) it is shown an example of how OLA method obtains the local region applying K-NN method for the 3 unknown samples; in this example the K parameter is given a value of 6. The circle around the new case and with the same color represents its local region, and the 6 nearest neighbors are highlighted in bold. It can be seen that a different base classifier is selected for each of those samples.



(a) Obtaining the local regions of OLA



(b) Obtaining the local regions of OLA in OVO using K-NN



(c) Obtaining the local regions of OLA in OVO using K-NNE

Figure 3: Example of how the local regions are obtained for each strategy.

Figure 3(b) shows the extension of OLA to OVO. It can be seen that in each sub-problem different samples take part in the decision of the base classifier, hence, in some cases different base classifier are selected to classify the same unlabelled instance in each sub-problem.

Figure 3(c) illustrates an extension of the previous figure using in this case K-NNE to select the local regions, in this approach the value of K is 3. The way to represent the local region of each new case varies: they are formed by the 3 nearest neighbors of each class. The instances that correspond to the local region of each new case are connected by a line of their color and they are highlighted in bold with a circle around them. Comparing with the previous example more distant instances take part in the classifier selection decision, but both classes are in equal conditions.

6. Experiments

In this section we explain the experimental setup. Moreover we carry out an empirical study in order to analyzed the usefulness of DYNOVO. To do so we compare the proposed variations of DYNOVO with the state-of-the-art methods.

6.1. Datasets

We have selected 22 databases from the UCI repository [4] to perform the experiments. A summary of these databases is shown in Table 1.

Table 1: Characteristics of the databases

Database	#Cases	#Atributes	#Classes
Annealing	798	38	5
Balance-scale	625	4	3
Car	1728	6	4
Cmc	1473	9	3
Dermatology	366	33	6
Ecoli	336	7	8
Glass	214	9	7
Image Segmentation	2310	19	7
Iris	150	4	3
Lymph	148	18	4
Nursery	12960	8	5
Optdigits	5620	64	10
Page-blocks	5473	10	5
Pendigits	10992	16	10
Satimage	6435	36	6
Solar-flare-1	323	12	6
Solar-flare-2	1066	12	6
Vehicle	846	18	4
Vowel	990	13	11
Waveform	5000	21	3
Wine	178	13	3
Zoo	101	17	7

6.2. Base Classifiers

To carry out the experiments we have chosen 5 different base classifiers from a software package for Machine Learning Called WEKA [20]. 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 treated the classifiers as black boxes and we have used their WEKA package default parameters.

- J48 (C4.5 clone) [33], decision tree algorithm. It makes a post-pruning phase, based on error based pruning algorithm. The parameters used are the following:
  - Confidence Factor = 0.25.
  - Minimum number of instances = 2.
  - Unpruned = False.

- SM0 (SVM clone) [31], kernel methods. It creates a hyperplane where the categories are divided by a clear gap that is as wide as possible. The parameters used are the following:
  - Fit logistic models = False.
  - C = 1.0.
  - Epsilon = 1.0E-12.
  - Kernel = Polynomial kernel.
  - Tolerance parameter = 0.001.
- JRip (Ripper clone) [8], rule induction classifier. It builds a rule-set by repeatedly adding rules to an empty rule-set until all positive examples are covered. The parameters used are the following:
  - Check error rate: True.
  - Minimal weights of instances: 2.0.
  - Number of runs of optimizations: 2.
  - Prune: True.
- Naive Bayes [23], 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.
- Bayesian Network, [13] statistical learning algorithm. It is a probabilistic graphical model that represents a set of random variables and their conditional independences via a directed acyclic graph. The parameters used are the following:
  - Estimator: Simple Estimator.
  - Search Algorithm: K2.
  - ADTree: False.

6.3. Experimental setup

The classification performance is obtained by means of a stratified 10-fold cross-validation. Some of the compared algorithms need a validation process which consists of a 5-fold cross-validation made for each training fold independently.

The DCS methods that we have selected in our proposal, use K-NN algorithm to define the local region, and depending on the K value the results vary. Because of that we have run these methods over several K values: 6,12,18,24, 30 when K-NN is used and 3,6,9,12,15 when K-NNE is used. It is worth mentioning that as in each sub-problem there are two classes and K-NNE obtains the K nearest neighbors of each class, the number of neighbors that take part in K-NN and K-NNE are the same.

6.4. Obtained results

In this sub-section we show the results obtained by 4 different variations of DYNOVO.

Table 2 shows the results obtained by DYNOVO when K-NN is used to obtain the local regions. This table is separated into two sections: in the left side are shown the results obtained when OLA is used as DCS method (DYNOVO-OLA-KNN), whereas in the right side are shown those obtained when DW-OLA is used (DYNOVO-DW-OLA-KNN).

Table 3 shows the results obtained by DYNOVO when K-NNE is used to obtain the local regions. As in the previous table, this time also the table is divided into two parts: in the left side are shown the results obtained when OLA is used as DCS method (DYNOVO-OLA-KNNE) and in the right side the results obtained when DW-OLA is used (DYNOVO-DW-OLA-KNNE).

For each DYNOVO variation the average of the best K is remarked in bold. These K values are used in the next sub-section to compare DYNOVO's variations with other state of the art methods.

	DYNOVO-OLA-KNN					DYNOVO-DW-OLA-KNN				
DB	K=6	K=12	K=18	K=24	K=30	K=6	K=12	K=18	K=24	K=30
anneal	98.998	98.886	98.886	98.664	98.775	99.220	99.109	99.332	99.220	99.109
balance-scale	89.600	89.600	89.760	89.920	89.760	89.600	89.760	89.600	89.760	89.760
car	96.065	96.181	96.296	96.296	96.296	96.065	96.007	96.123	96.123	96.123
cmc	54.175	54.039	52.682	54.311	54.175	53.700	54.447	53.225	54.107	54.039
dermatology	96.721	97.268	97.541	97.541	97.541	96.721	96.721	96.448	97.268	97.268
ecoli	86.905	87.202	87.798	87.500	87.202	86.905	87.798	87.202	87.202	87.202
glass	68.224	71.495	70.093	70.093	69.626	68.692	71.495	70.561	71.028	71.028
iris	95.333	97.333	96.667	95.333	95.333	96.000	96.000	95.333	95.333	95.333
imgsegment	97.229	97.359	97.489	97.273	97.229	97.229	97.229	97.532	97.229	97.229
lymph	87.838	87.162	85.135	83.784	85.135	87.838	88.514	85.811	83.784	84.459
nursery	98.526	98.526	98.549	98.573	98.573	98.526	98.526	98.611	98.634	98.634
optdigits	98.310	98.203	98.132	98.132	98.132	98.238	98.149	98.043	98.043	98.060
page-blocks	97.278	97.058	97.077	97.040	97.131	97.223	97.150	97.223	97.186	97.278
pendigits	98.781	98.817	98.763	98.799	98.754	98.772	98.790	98.790	98.817	98.790
satimg	89.464	89.448	89.510	89.371	89.355	89.510	89.588	89.588	89.448	89.542
solar-flare1	70.279	70.279	69.969	69.969	69.969	71.827	71.827	71.517	71.207	71.827
solar-flare2	75.328	75.235	75.141	75.141	75.141	75.235	75.235	75.141	75.047	75.141
vehicle	73.995	74.823	74.586	74.232	74.586	73.404	74.941	73.759	73.759	74.941
vowel	90.909	89.192	89.495	89.495	89.293	91.818	90.909	90.808	90.909	90.909
waveform-5000	84.520	84.520	85.180	85.260	85.560	84.020	83.860	84.600	84.800	85.040
wine	95.506	95.506	95.506	95.506	95.506	96.629	96.629	96.629	96.629	96.629
zoo	97.030	97.030	97.030	97.030	97.030	97.030	97.030	97.030	97.030	97.030
Mean	88.228	**88.416**	88.240	88.148	88.186	88.373	**88.623**	88.314	88.298	88.426

Table 2: Classification accuracies of DYNOVO when K-NN is used to obtain the local region.

	DYNOVO-OLA-KNNE					DYNOVO-DW-OLA-KNNE				
DB	K=3	K=6	K=9	K=12	K=15	K=3	K=6	K=9	K=12	K=15
anneal	98.998	98.775	98.886	98.886	98.664	99.332	99.220	99.443	99.443	99.443
balance-scale	88.640	89.600	89.600	89.440	89.760	89.440	90.720	90.720	90.560	90.880
car	96.065	95.428	96.181	96.296	96.238	96.817	96.181	96.470	96.470	96.470
cmc	53.836	54.243	53.225	53.496	53.632	61.371	62.322	62.322	63.069	62.865
dermatology	96.995	97.541	96.995	97.268	97.541	96.995	96.995	96.721	97.268	97.541
ecoli	86.310	86.905	86.905	87.798	87.500	87.798	88.095	89.583	89.583	90.179
glass	72.897	71.495	71.963	71.495	71.028	75.701	76.636	75.234	75.701	75.234
iris	96.000	96.667	96.000	96.000	96.000	95.333	96.000	96.000	96.000	95.333
imgsegment	97.186	97.143	96.883	97.013	96.797	97.532	97.749	97.706	97.532	97.403
lymph	87.838	87.162	87.162	86.486	86.486	87.162	87.838	86.486	87.162	86.486
nursery	98.511	98.349	98.156	98.526	98.573	98.634	98.495	98.387	98.696	98.719
optdigits	98.149	98.132	98.096	98.025	97.954	98.096	98.096	98.060	98.007	97.883
page-blocks	96.638	96.821	96.675	96.656	96.620	97.625	97.625	97.533	97.552	97.460
pendigits	98.362	98.353	98.435	98.444	98.444	98.408	98.344	98.372	98.444	98.490
satimg	88.594	88.205	88.625	88.656	88.485	89.899	89.806	89.930	90.070	90.023
solar-flare1	70.279	69.969	69.969	69.659	69.659	73.994	73.994	73.994	73.684	73.375
solar-flare2	74.953	75.235	75.235	74.953	75.047	76.735	77.486	77.486	76.923	76.923
vehicle	74.823	74.704	73.641	75.296	75.414	80.733	82.151	82.388	83.097	83.688
vowel	90.404	90.000	89.293	89.293	88.889	91.818	91.616	90.909	90.707	91.616
waveform-5000	84.000	83.920	84.160	84.540	84.540	84.380	84.320	84.400	84.980	85.160
wine	95.506	95.506	95.506	95.506	95.506	96.067	96.629	96.629	96.629	96.629
zoo	97.030	97.030	97.030	97.030	97.030	97.030	97.030	97.030	97.030	97.030
Mean	**88.273**	88.236	88.119	88.217	88.173	89.586	89.879	89.809	89.937	**89.947**

Table 3: Classification accuracies of DYNOVO when K-NNE is used to obtain the local region.

6.5. Comparing the results

In this sub-section we compare our proposals with other state-of-the-art methods. We have divided the experiments into two parts: in the first one OLA is applied in those methods that select the classifiers dynamically, while in the second one DW-OLA is applied. We show the results of each part on Tables 4 and 5. Following, we briefly describe the strategies that correspond to each column of the tables.

- Best Single (OVO-BS) [12]: Each database is classified with every classifier defined in sub-section 6.2 applying OVO decomposition strategy. The result of the best base classifier is shown in each database.
- Galar et al. (Galar) [16]: It finds the K nearest neighbors of the test instance and it applies OVO only considering those classes in the neighborhood. The K value is established to 3 times the number of classes. The result of the best base classifier is shown in each database.
- Static selection (OVO-ST) [38]: For each sub-problem it is selected independently the base classifier that obtains the best result after a validation process.
- DCS methods (OLA [39] or DW-OLA [36]): Depending on the table the DCS strategy that is used vary. In the first table is OLA the strategy that is compared, while in the second table is DW-OLA. As it has been commented before, the DCS strategies are run over several K values, in the tables the results of the K value with the highest mean are shown.
- Dynamic selection of the base classifier in each sub-problem with K-NN (DYNOVO-OLA-KNN or DYNOVO-DW-OLA-KNN): It is tried to select the best base classifier in each sub-problem independently and dynamically. K-NN is used to obtain the local region in DCS strategies. The results obtained by the best K value in Table 2 are shown. In Figure 3(b) it can be seen a graphical example of how DYNOVO-OLA-KNN and DYNOVO-DW-OLA-KNN obtain the local regions.
- Dynamic selection of the base classifier in each sub-problem with K-NNE (DYNOVO-OLA-KNNE or DYNOVO-DW-OLA-KNNE): Similar to the previous one with the difference that it uses K-NNE instead of K-NN. The results obtained by the best K value in Table 3 are shown. In Figure 3(c) it can be seen a graphical example of how DYNOVO-OLA-KNNE and DYNOVO-DW-OLA-KNNE obtain the local regions.

Table 4 shows the results obtained when OLA is used in the methods that select dynamically the base classifiers. It could be seen that our proposal DYNOVO-OLA-KNN shows the best result in the majority of the cases: it reaches the best result in 8 of the databases. Moreover it achieves the best mean and rank values also. Our other proposal, DYNOVO-OLA-KNNE, obtains the best results in 4 of the databases and it gets the third best mean.

Table 5 shows the results obtained when DW-OLA is used in the methods that select dynamically the base classifiers. Our proposal DYNOVO-DW-OLA-KNNE shows the best result in 15 of the databases and it reaches also the best mean and rank. This time our other proposal, DYNOVO-DW-OLA-KNN, also gets interesting results since it obtains the second best mean and rank.

These results, show that methods which select the base classifiers dynamically in OVO obtain promising results. However, we can not obtain any meaningful conclusion without using a statistical test. Hence, in the next sub-section, we carry out an statistical analysis in order to find whether significant differences among the results obtained exists or not.

6.6. Statistical analysis

As we have several methods to compare, according to García et al. [17], we have used the Iman-Davenport test to detect statistical differences among the different strategies. If the difference exists, we apply the Shaffer post-hoc test in order to find out which algorithms are distinctive among them. We show the most relevant p-values obtained in the pairwise comparisons in tables, where ”+” symbol implies that the first algorithm is statistically better than the confronting one, whereas ”=” means that there are not significant differences between them.

With respect to OLA the results of the statistical analysis reject the null hypothesis that all the methods are equivalent, since the p-value (0.0200) returned by the Iman-Davenport test is lower than our α-value (0.1). In Table 6

DB	OVO-BS	Galar	OVO-ST	OLA	DYNOVO-OLA-KNN	DYNOVO-OLA-KNNE
anneal	98.552	98.552	**98.998**	98.664	98.886	**98.998**
balance-scale	**90.400**	**90.400**	89.120	89.440	89.600	88.640
car	93.866	93.866	93.692	95.833	**96.181**	96.065
cmc	**54.582**	54.447	53.089	51.663	54.039	53.836
dermatology	97.541	**98.361**	96.995	95.902	97.268	96.995
glass	**73.832**	**73.832**	70.561	71.495	71.495	72.897
ecoli	86.607	86.905	85.417	86.607	**87.202**	86.310
imgsegment	97.186	97.013	97.143	97.143	**97.359**	97.186
iris	96.667	96.667	96.667	94.667	**97.333**	96.000
lymph	87.162	87.162	86.486	86.486	87.162	**87.838**
nursery	97.238	97.130	97.824	98.071	**98.526**	98.511
optdigits	98.292	**98.523**	98.256	98.256	98.203	98.149
page-blocks	**97.223**	97.168	97.003	97.003	97.058	96.638
pendigits	98.026	98.299	98.426	98.690	**98.817**	98.362
satimg	88.283	88.361	88.454	88.858	**89.448**	88.594
solar-flare1	70.279	70.588	69.969	**71.517**	70.279	70.279
solar-flare2	**75.516**	**75.516**	75.141	74.672	75.235	74.953
vehicle	75.414	75.887	**76.123**	74.941	74.823	74.823
vowel	82.828	83.636	84.949	83.232	89.192	**90.404**
waveform-5000	86.700	**86.720**	86.680	84.500	84.520	84.000
wine	**98.876**	**98.876**	96.067	98.315	95.506	95.506
zoo	96.040	96.040	95.050	95.050	**97.030**	**97.030**
Mean	88.232	88.361	87.823	87.773	**88.416**	88.273
Rank	3.16	2.93	4.20	4.16	**2.72**	3.82

Table 4: Classification accuracies of different methods. In those approaches that select the classifiers dynamically, OLA method is used.

we show the most relevant p-values obtained with Shaffer post-hoc test. Although there are not statistical differences in each pairwise comparisons, DYNOVO-OLA-KNN is close to outperform statistically OVO-ST and OLA, since the p-value is low. Because of that, and taking into account that the results obtained in Table 4, we consider that DYNOVO-OLA-KNN performs better than the other methods.

Considering DW-OLA, the Iman-Davenport test also returns p-value (0.0002) lower than α-value, so we execute the Shafer post-hoc test. The achieved p-values could be seen in Table 7. The results show that DYNOVO-DW-OLA-KNNE is the most robust strategy since it outperforms significantly OVO-BS, OVO-ST and DW-OLA.

6.7. Computational complexity

In order to provide a more complete study, we analyze the time and space complexity of our proposal.

The computational load of building the model is pretty big, since it involves to classify every training instance over every classifiers for every pair of classes. Those results are stored on a table, hence, this task only needs to be executed once. At classification time the information of the tables is retrieved from the table.

To analyse the computational and spatial complexity of classifying a new instance, let us examine the process that such instance undergoes.

- *For every pair of classes in the dataset, a vote is cast:* The number of pair of classes is $O(C^2)$, where C is the number of classes.
  - *Search the K nearest neighbors:* K-NN using kd-tree has a search time of $O(K \log(I_{TR}))$, where I_{TR} the number of instances in the training set from where the model has been built and K is the number of nearest neighbours.
  - *Search the classifier that best classifies the neighbors:* This is achieved by a search in a table that stores if a classifier type classified correctly an instance in the sub-problem associated to a pair of classes. The table has the pair of classes, the training instances and the classifier types as keys and a boolean as value. If implemented as a hash table, the searching time is $O(1)$ in the average.
  - *The instance is classified according to the best classifier:* It is clear that this depends of the classifier, but being K-NN a lazy algorithm, and thus a slow one, it looks sensible to assume $O(K \log I_{TR})$ is an upper bound in the execution time.
- *The instance is assigned the class with the majority of votes:* It takes $O(C^2)$ time to tally all the votes.

DB	OVO-BS	Galar	OVO-ST	DW-OLA	DYNOVO-DW-OLA-KNN	DYNOVO-DW-OLA-KNNE
anneal	98.552	98.552	98.998	99.220	99.109	**99.443**
balance-scale	90.400	90.400	89.120	89.120	89.760	**90.880**
car	93.866	93.866	93.692	95.833	96.007	**96.470**
cmc	54.582	54.447	53.089	51.663	54.447	**62.865**
dermatology	97.541	**98.361**	96.995	95.082	96.721	97.541
glass	73.832	73.832	70.561	71.495	71.495	**75.234**
ecoli	86.607	86.905	85.417	84.821	87.798	**90.179**
imgsegment	97.186	97.013	97.143	96.926	97.229	**97.403**
iris	**96.667**	**96.667**	**96.667**	94.667	96.000	95.333
lymph	87.162	87.162	86.486	**88.514**	**88.514**	86.486
nursery	97.238	97.130	97.824	98.071	98.526	**98.719**
optdigits	98.292	**98.523**	98.256	98.185	98.149	97.883
page-blocks	97.223	97.168	97.003	96.766	97.150	**97.460**
pendigits	98.026	98.299	98.426	98.717	**98.790**	98.490
satimg	88.283	88.361	88.454	88.827	89.588	**90.023**
solar-flare1	70.279	70.588	69.969	72.136	71.827	**73.375**
solar-flare2	75.516	75.516	75.141	74.578	75.235	**76.923**
vehicle	75.414	75.887	76.123	75.650	74.941	**83.688**
vowel	82.828	83.636	84.949	85.455	90.909	**91.616**
waveform-5000	86.700	**86.720**	86.680	83.920	83.860	85.160
wine	**98.876**	**98.876**	96.067	98.315	96.629	96.629
zoo	96.040	96.040	95.050	96.040	**97.030**	**97.030**
Mean	88.232	88.361	87.823	87.909	88.623	**89.947**
Rank	3.59	3.36	4.50	4.16	3.30	**2.09**

Table 5: Classification accuracies of different methods. In those approaches that select the classifiers dynamically, DW-OLA method is used.

Table 6: Shaffer test results when OLA is used

Hypothesis	p-value
DYNOVO-OLA-KNN vs OVO-ST	=(0.1323)
DYNOVO-OLA-KNNvs OLA	=(0.1323)
Galar vs OVO-ST	=(0.2405)
Galar vs OLA	=(0.2958)
DYNOVO-OLA-KNN vs DYNOVO-OLA-KNNE	=(0.5312)
OVO-BS vs OVO-ST	=(0.6383)
OVO-BS vs OLA	=(0.6383)
Galar vs DYNOVO-OLA-KNNE	=(0.8127)

Table 7: Shaffer test results when DW-OLA is used

Hypothesis	p-value
DYNOVO-DW-OLA-KNNE vs OVO-ST	**+(2.9E-4)**
DYNOVO-DW-OLA-KNNEvs DW-OLA	**+(0.0025)**
DYNOVO-DW-OLA-KNNE vs OVO-BS	**+(0.0783)**
DYNOVO-DW-OLA-KNNE vs Galar	=(0.2405)
DYNOVO-DW-OLA-KNNE vs DYNOVO-DW-OLA-KNN	=(0.3273)
DYNOVO-DW-OLA-KNN vs OVO-ST	=(0.3273)
Galar vs OVO-ST	=(0.3273)
OVO-BS vs OVO-ST	=(0.7493)
DYNOVO-DW-OLA-KNN vs OLA	=(0.8803)
Galar vs OLA	=(0.9509)

Within these assumptions the average execution time of all the process is $O(K\log(I_{TR})C^2)$. Let us note that I_{TR} is different for every pair of classes in the C^2 sub-problems, but in average will be $(N/C) * 2$. If N is the number of instances in the original database, the average execution time will be $O(K\log(N)C^2)$, so the classification time is logarithmic in the number of instances in the original dataset and quadratic in the number of classes, provided reasonable classification complexity of the classifiers used.

With respect to space complexity, the approach would request $O(C^2 I_{TR} T)$ space, that, as stated above, amounts to $O(NCT)$, with N the number of instances in the original database. Storage of the classifier models should never be bigger than $O(KN)$, even with lazy paradigms using kd-trees structures or similar.

Compared with other OVO versions, our proposal has a bigger space complexity, due to the need of storing big tables. About time complexity, only the search for the K neighbours and the lookup in the hash table are not made in other OVO versions. As the comparisons with other methods are considerably in favour of DYNOVO-DW-OLA-

KNNE, and the differences with other OVO versions are mostly in space requirements, we consider that its good performance compensates this extra computational cost.

6.8. Discussion

After all these experiments, considering only the state-of-the-art methods, the first conclusion that we have obtained is that selecting different base classifier for each sub-problem statically in OVO (OVO-ST), does not outperform the best single classifier in OVO (OVO-BS). These results coincide with those found in the state-of-the-art [27] [37]. On the other hand, it is worth mentioning that the algorithm proposed by Galar et al. [16] obtains interesting result and although it uses less sub-problems than OVO, it shows the best performance among the state-of-the-art methods.

On the other hand it can be seen that the proposed approach obtains promising results. It gets better mean and rank than the compared methods with almost all the variations (the exception is DYNOVO-OLA-KNNE strategy). Moreover the statistical tests show the good performance of our proposal.

Finally DYNOVO-DW-OLA-KNNE is which shows the best performance. It obtains the best mean with a significant difference and the statistical test shows its solidity. Furthermore, it can be seen in Table 3 that all the averages obtained with the different K values overcome the averages obtained by state-of-the-art methods. The combination of DW-OLA and K-NNE gives some advantages which result beneficial to select the appropriate base classifier. Let us consider that we are trying to select the appropriate base classifier to classify a new unknown instance for the sub-problem that distinguishes between θ_i and θ_j classes. Also consider that all its K nearest nehighbors belong to θ_i class. Under these circumstances, it is more likely to select a base classifier that tends to return θ_i class. But if the new unknown sample belongs to θ_j, it is more likely to predict the wrong class. This problem can be minimized using K-NNE algorithm, since it gives the chance to participate to all the classes. In this manner the selected base classifier should be able to differentiate both classes. However, it is possible to be a significant difference in the distance to the new unknown sample between the K nearest neighbors of θ_i and θ_j. Therefore it is not completely adequate that all the neighbors have the same influence when the base classifier is selected. So one possibility is to assign different weights to each neighbor depending in their distance to the new sample, in other words, apply DW-OLA.

7. Conclusion

In this paper we present a new proposal called DYNOVO which aims to improve classification accuracy in supervised classification multi-class problems. Among several base classifiers, the approach attempts to select the best base classifier in OVO dynamically for each test patterns. To do so we have chosen several well-known classifiers from different Machine Learning paradigms: SVM, C4.5 Decision Tree, Ripper, Bayes Networks and Naive Bayes. We have presented 4 different variations of our proposal which have been tested over 22 databases from the UCI repository.

The novel procedure proposed has shown its usefulness due to the competitive results obtained. We have shown the positive synergy existing between OVO and DCS strategies, specially when K-NNE is utilized to obtain the local region and the instances of the local region are weighted by the distance to the new unknown case.

This fact open doors for future combinations of OVO and DCS using more complex DCS strategies or to extend it to Dynamic Ensemble Selection strategies. Furthermore, it would be interesting to introduce these strategies in the more general ECOC framework.

Acknowledgements

The work described in this paper was partially conducted within the Basque Government Research Team Grant IT313-10 and the University of the Basque Country UPV/EHU. I. Mendialdua holds a Grant from Basque Government.

References

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

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

[3] Arruti, A., Mendialdua, I., Sierra, B., Lazkano, E., Jauregi, E., 2014. New one versus allone method: Nov@. Expert Systems with Applications 41, 6251–6260.

[4] Bache, K., Lichman, M., 2013. UCI machine learning repository. URL: `http://archive.ics.uci.edu/ml`.

[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] Bautista, M.Á., Escalera, S., Baró, X., Radeva, P., Vitriá, J., Pujol, O., 2012. Minimal design of error-correcting output codes. Pattern Recognition Letters 33, 693 – 702.

[7] Cavalin, P.R., Sabourin, R., Suen, C.Y., 2013. Dynamic selection approaches for multiple classifier systems. Neural Computing and Applications 22, 673–688.

[8] Cohen, W.W., 1995. Fast effective rule induction, in: Proceedings of the Twelfth international conference on machine learning, pp. 115–123.

[9] Dietterich, T.G., Bakiri, G., 1995. Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research 2, 263–286.

[10] Dos Santos, E.M., Sabourin, R., Maupin, P., 2008. A dynamic overproduce-and-choose strategy for the selection of classifier ensembles. Pattern Recognition 41, 2993–3009.

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

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

[13] Friedman, N., Geiger, D., Goldszmidt, M., 1997. Bayesian network classifiers. Machine learning 29, 131–163.

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

[15] 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, 1761–1776.

[16] 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, 3412 – 3424.

[17] 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, 2044–2064.

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

[19] Giacinto, G., Roli, F., 1999. Methods for dynamic classifier selection, in: International Conference on Image Analysis and Processing, IEEE Computer Society. pp. 659–659.

[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, 10–18.

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

[22] Hüllermeier, E., Vanderlooy, S., 2010. Combining predictions in pairwise classification: An optimal adaptive voting strategy and its relation to weighted voting. Pattern Recognition 43, 128–142.

[23] 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.

[24] Kapp, M.N., Sabourin, R., Maupin, P., 2012. A dynamic model selection strategy for support vector machine classifiers. Applied Soft Computing 12, 2550–2565.

[25] Ko, A.H., Sabourin, R., Britto Jr, A.S., 2008. From dynamic classifier selection to dynamic ensemble selection. Pattern Recognition 41, 1718–1731.

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

[27] 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.

[28] Liu, R., Yuan, B., 2001. Multiple classifiers combination by clustering and selection. Information Fusion 2, 163–168.

[29] 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, 19–37.

[30] Lorena, A.C., De Carvalho, A.C., 2008. Evolutionary tuning of svm parameter values in multiclass problems. Neurocomputing 71, 3326–3334.

[31] 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.

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

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

[34] Reid, S.R., 2010. Model Combination in Multiclass Classification. Ph.D. thesis. University of Colorado.

[35] Sierra, B., Lazkano, E., Irigoien, I., Jauregi, E., Mendialdua, I., 2011. K nearest neighbor equality: Giving equal chance to all existing classes. Information Sciences 181, 5158–5168.

[36] Smits, P.C., 2002. Multiple classifier systems for supervised remote sensing image classification based on dynamic classifier selection. IEEE Transactions on Geoscience and Remote Sensing 40, 801–813.

[37] de Souza, B.F., de Carvalho, A., Calvo, R., Ishii, R.P., 2006. Multiclass svm model selection using particle swarm optimization, in: Proceedings of the Sixth International Conference on Hybrid Intelligent Systems, IEEE. pp. 31–31.

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

[39] Woods, K., 1997. Combination of multiple classifiers using local accuracy estimates. IEEE Transactions on Pattern Analysis and Machine

Intelligence 19, 405–410.