MC^{2}ESVM: Multiclass Classification based on Cooperative Evolution of Support Vector Machines

Alejandro Rosales-Pérez and Hugo Terashima-Marin,
School of Engineering and Sciences, Tecnologico de Monterrey, Mexico
Salvador García and Francisco Herrera,
Department of Computer Science and Artificial Intelligence, University of Granada, Spain
Carlos A. Coello Coello,
Department of Computer Science, CINVESTAV-IPN, Mexico

Abstract

Support vector machines (SVMs) are one of the most powerful learning algorithms for solving classification problems. However, in their original formulation, they only deal with binary classification. Traditional extensions of the binary SVMs for multiclass problems are based either on decomposing the problem into a number of binary classification problems, which are then independently solved, or on reformulating the objective function by solving larger optimization problems. In this paper, we propose MC^{2}ESVM, an approach for multiclass classification based on the cooperative evolution of SVMs. Cooperative evolution allows us to decompose an M-class problem into M subproblems, which are simultaneously optimized in a cooperative fashion. We have reformulated the optimization problem such that it focuses on learning the support vectors for each class at the time that it takes into account the information from other classes. A comprehensive experimental study using common benchmark datasets is carried out to validate MC^{2}ESVM. The experimental results, supported by statistical tests, show the effectiveness of MC^{2}ESVM for solving multiclass classification problems, while keeping a reasonable number of support vectors.

I. Introduction

Support Vector Machines (SVMs) [1] are powerful supervised learning algorithms with strong theoretical foundations that have shown a high performance over a wide range of problems [2]–[4]. The main idea behind SVMs is to find the hyperplane that maximizes the separation between two classes, which is defined through the so-called *support vectors*. In spite of the effectiveness of SVMs in solving binary classification problems, real world problems often require discriminating among more than two classes.

Over the last years, there has been interest in extending SVMs to multiclass problems. These approaches can be differentiated into two major groups: (1) decomposition strategies

Corresponding Author: Alejandro Rosales-Pérez (Email: arosalesp85@gmail.com)

and (2) single machine methods. The first type of approach is based on decomposing the M-class problem into several binary classification problems. The most well-known decomposition techniques are the one-vs-one (OVO) and the one-vs-all (OVA) methods. These have been found to be quite effective in solving multiclass problems [5], [6]. However, they assume that each binary classification problem to be solved is independent of the rest.

On the other hand, single machine approaches are based on modifying the optimization problem, such that the multiclass SVM classifier is constructed based on solving a single optimization problem [7]. Nonetheless, they have the shortcoming of dealing with a more complex and larger optimization problem.

Evolutionary algorithms (EAs) encompass a family of algorithms that aim at solving complex optimization problems. EAs have been applied with success to the solution of different machine learning problems [8]–[10]. In recent years, several studies that hybridize EAs with SVMs have been reported [11]–[13]. Most of them deal with the hyper-parameter optimization problem. There are only a few attempts to deal with the parameter optimization problem, such as those reported in [14], [15]. They have, however, only focused on the classical binary classification problems.

This paper introduces MC^{2}ESVM (Multiclass Classification based on the Cooperative Evolution of SVMs). MC^{2}ESVM aims at taking advantage of the benefits of both decomposition and single machine approaches, by decomposing the multiclass problem and solving the resulting problems as single-objective optimization problems, optimizing the support vector for each class. This can be approached in a natural fashion with cooperative coevolutionary algorithms. Moreover, the inherent advantages of evolutionary algorithms allow MC^{2}ESVM to handle non-positive semidefinite kernels[1]. The main contributions of this paper are the following:

- The decomposition of the multiclass problem via coevolutionary optimization. This allows SVMs to be able to learn multiclass classifiers in a single optimization run by simultaneously solving a set of simpler problems. To the best of the authors' knowledge, this is the first attempt to combine coevolutionary algorithms with SVMs for multiclass problems.

[1]Non-positive semidefinite kernels can lead to a non-convex optimization for SVMs.

- A derivation of the optimization problem that learns the class-specific support vectors, considering the information from other classes.

The performance of MC^{2}ESVM is assessed using a suite of 25 multiclass classification datasets. We first compare it with state-of-the-art SVMs extensions in terms of the prediction performance and common learning algorithms. Second, we compare with respect to the support vectors. Afterwards, we assess its scalability as either the number of instances or classes are increased. Finally, we assess the stability of the algorithm and the evolutionary parameters. Our experimental results show the effectiveness of MC^{2}ESVM for solving the classification task, while keeping a reasonable number of support vectors. These findings are supported by a set of non-parametric tests.

The remainder of this paper is organized as follows. Section II describes some preliminary concepts related to the main extension to multiclass SVM and coevolutionary optimization. Section III describes in detail our proposed MC^{2}ESVM. Next, Section IV outlines the experimental settings for our study, while Section V presents the experimental results and the statistical validation. Finally, Section VI provides our general conclusions.

II. Preliminaries

This section discusses the main preliminaries in which our contribution is based. Section II-A describes the main extensions proposed to solve multiclass problems using SVMs. Next, in Section II-B, we describe the main characteristics of coevolutionary algorithms.

A. Multiclass Extensions for SVMs

A number of approaches for extending SVMs so that they can handle multiclass problems have been proposed. Fig. 1 shows the proposed methods to approach multiclass problems with SVMs. They are briefly discussed next.

1) Decomposition strategies

These approaches follow the idea of dividing the multiclass problems into several binary classification problems. The most common decomposition methods for multiclass SVMs are the following:

- ***One-vs-All*** (OVA) [16]: OVA decomposes the M-class problem into M subproblems. M-binary SVMs are constructed for each subproblem, such that the ith SVM is trained using the samples belonging to the ith class as positive samples and the remaining are

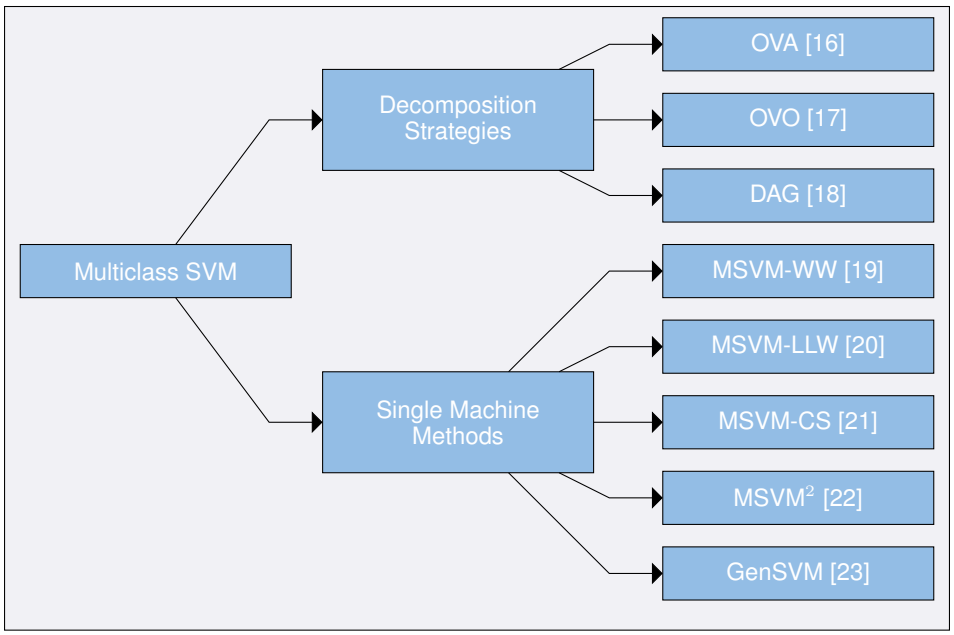


Fig. 1: SVMs extensions for handling multiclass problems.

treated as negative samples. A new sample is assigned to the class with the largest activation value. OVA introduces an artificial imbalance during the training. Thus, the higher the value of M, the higher the imbalance rate.

- ***One-vs-One*** (OVO) [17]: In OVO, an SVM is trained for each possible pair of classes, resulting in a total of $M(M-1)/2$ SVMs. This number is usually larger than the one of the OVA approach. In the prediction phase, a new sample is classified for each SVM and the class with the majority vote wins. The main criticism of OVO is that when M is large, the evaluation of the $M(M-1)/2$ SVMs can slow down the prediction stage of the resulting OVO.
- ***Directed Acyclic Graph*** (DAG) [18]: The training phase is similar to OVO, resulting in $M(M-1)/2$ SVMs constructed for each pair of classes. The difference relies on the prediction stage. DAG starts at the root, where an SVM is used to classify the test sample, and it moves either to the left or to the right path, depending on the predicted class given by the SVM. This process is repeated until a leaf node is reached, which indicates the predicted class. Note that, however, the performance of DAG depends on the SVM at the root node.

A comparison between these three strategies is performed in [24], finding that their

accuracy is quite similar, with no statistical difference.

2) Single Machine Methods

These methods aim at solving directly the multiclass problem during the training phase. This is attained by modifying the SVM objective function, such that it simultaneously allows computing the multiclass classifier. For instance, in [19], authors propose MSVM-WW, where the single objective formulation for the multiclass SVM is given as follows:

$$\begin{aligned} \min_{\mathbf{w}_r,\xi^r,b_r} \frac{1}{2}\sum_{r=1}^{M}\|\mathbf{w}_r\|^2 + \frac{C}{N}\sum_{i=1}^{N}\sum_{q}\xi_i^q \\ \text{subject to } \langle \mathbf{w}_r, \mathbf{x}_i\rangle + b_r \geq \langle \mathbf{w}_q, \mathbf{x}_i\rangle + b_q + 2 - \xi_i^q, \quad \xi_i^r \geq 0. \end{aligned} \tag{1}$$

where $q = \{1, \ldots, M\}\backslash r$, N is the number of training samples, and C is a penalty parameter that controls the trade-off between accuracy and complexity.

This formulation, however, has to deal with a large number of slack variables. Other formulations of the objective function are MSVM-LLW [20], which reduces the dimensionality of the problem by means of a sum-to-zero constraint; MSVM-CS [21], which only takes into account the largest activation and the bias term is not considered during the training; MSVM^2 [22], which adds a quadratic function to the slack variables; and GenSVM [23], which uses a simplex encoding to reduce the dimensionality of the problem. These approaches have reported similar performance to those obtained by either OVA or OVO. Nonetheless, these methods have the disadvantage of dealing with larger optimization problems.

B. Coevolutionary Optimization

A coevolutionary algorithm is an evolutionary algorithm which is able to manage two or more populations simultaneously [25]. An important characteristic of these algorithms is that they allow to split the problem into different parts and assign a different population to each subproblem. Each population focuses its efforts on solving one specific part of the problem. Two different kinds of coevolutionary algorithms can be described:

- **Competitive coevolutionary algorithms** [26]. The individuals of each population compete against each other, such that the fitness value of an individual decreases as the result of an increment in the fitness value of its adversaries. Competitive coevolution is normally adopted for game-like problems.

- **Cooperative coevolutionary algorithms** [27]. Each population evolves individuals representing a part of the solution. A complete solution is composed by joining individuals from all the populations. Therefore, the fitness value of an individual is the result of its collaboration with other individuals from other populations.

In this work, our focus is on cooperative coevolution, due to the fact that it allows us to decompose the multiclass classification problem in a natural fashion, by assigning to each subproblem the task of learning the set of support vectors for each class.

III. MC^{2}ESVM: Multiclass Classification based on Cooperative Evolution of Support Vector Machines

The proposed MC^{2}ESVM aims at training a multiclass SVM in a single step. The multiclass classifier is defined by the set of support vectors of each class. MC^{2}ESVM is based on the cooperative coevolutionary algorithm, in which each subpopulation optimizes the support vectors for each class at the same time that it considers the other subpopulations for solving the multiclass problem. Algorithm 1 describes MC^{2}ESVM. Generally, it follows these steps:

Algorithm 1 MC^{2}ESVM

Require: $\mathcal{X}$, the set of samples,
$\mathcal{Y}$, the set of classes labels,
C, the regularization term,
P, the population size,
E, maximum number of evaluations.
Ensure: The set of support vectors

```
Generate randomly an initial population, P_y for each class y ∈ 𝒴
for each y ∈ 𝒴 do
    Select randomly an individual from each class y' ∈ 𝒴 \ y
    Construct full solutions by combining the selected individuals of each class
    Evaluate the full solutions using the fitness function
end for
while a stopping criterion is not met do
    for each y ∈ 𝒴 do
        Select the best individual from each population y' ∈ 𝒴 \ y
        for each individual in the current class do
            Apply evolutionary operators to create an offspring
            Evaluate the offspring with the fitness function
            Add the offspring to the next generation if it improves its parents
        end for
    end for
end while

Construct the final solution based on the best individuals of each population
```

1) In line 1, for each class, a population is randomly created. The number of variables for each population depends on the number of samples in the training set for the given class.
2) In lines from 3 to 5, the fitness value is assigned for each individual of each class (population). For doing so, an individual from other classes is randomly selected to build the multiclass classifier. This is part of the cooperative coevolution.
3) Lines from 7 to 16 are the evolutionary process, as follows:
   a) Line 9 selects the best individual from other classes and the evolutionary operators are applied to create an offspring, in line 11.
   b) Line 12 computes the fitness value of the offspring solution by concatenating it with the best solutions from other classes.
   c) In line 13, the best solutions among the parents and offspring are selected to be included in the next generation.
4) Once the evolutionary process is over, the final solutions, i.e., the support vectors for the multiclass problem are obtained from the best individuals of each class; this is done in line 17.

The details of MC^{2}ESVM are given in the remainder of this section. First, in Section III-A, we explain the optimization problem for finding the support vectors for a multiclass problem. Next, Section III-B presents the representation adopted in the evolutionary optimization as well as the evolutionary operators. Section III-C describes the final steps to construct the multiclass SVM and in Section III-D, we discuss the extension to learn nonlinear functions.

A. Fitness Functions: Optimization Problem

In MC^{2}ESVM, each subproblem aims at learning the support vector for a given class label. Therefore, the optimization problem for the rth class is formulated as follows:

$$\begin{aligned} &\min_{\mathbf{w}_r, \xi^r} \frac{1}{2} \|\mathbf{w}_r\|^2 + C \sum_{i=1}^{n_r} \xi_i^r \\ &\text{subject to } \langle \mathbf{w}_r, \mathbf{x}_i \rangle \geq 1 - \xi_i^r, \quad \xi_i^r \geq 0 \end{aligned} \tag{2}$$

where n_r is the number of samples for the rth class.

For the sake of simplicity, we have omitted the bias term in our formulation. Moreover, since a population is focused on learning the support vectors for a single class, the class

label is not part of Equation 2. This leads to a simpler dual problem, with no additional constraints. For the dual formulation, the constraint is incorporated in the objective function by using the Lagrange multipliers:

$$\begin{aligned}\mathcal{L}\left(\mathbf{w}, \xi^r, \alpha_i^r, \beta_i^r\right) &= \frac{1}{2}\left\|\mathbf{w}_r\right\|^2 + C\sum_{i=1}^{n_r}\xi_i^r - \sum_{i=1}^{n_r}\alpha_i^r\left[\left\langle\mathbf{w}_r, \mathbf{x}_i\right\rangle - 1 + \xi_i^r\right] - \sum_{i=1}^{n_r}\beta_i^r\xi_i^r \\ &\text{subject to } \alpha_i^r, \beta_i^r \geq 0.\end{aligned} \tag{3}$$

Setting the partial derivatives to zero gives:

$$\frac{\partial\mathcal{L}}{\partial\mathbf{w}_r} = 0 \rightarrow \mathbf{w}_r = \sum_{i=1}^{n_r}\alpha_i^r\mathbf{x}_i \tag{4}$$

$$\frac{\partial\mathcal{L}}{\partial\xi_i^r} = 0 \rightarrow \beta_i^r = C - \alpha_i^r. \tag{5}$$

Substituting Equations 4 and 5 in Equation 3, gives the following dual optimization problem:

$$\begin{aligned}&\min_{\alpha_i^r}\frac{1}{2}\sum_{i,j=1}^{n_r}\alpha_i^r\alpha_j^r\left\langle\mathbf{x}_i, \mathbf{x}_j\right\rangle - \sum_{i=1}^{n_r}\alpha_i^r \\ &\text{subject to } 0 \leq \alpha_i^r \leq C.\end{aligned} \tag{6}$$

At this point, Equation 6 learns the support vectors for a single class label. Since the goal of MC^{2}ESVM is to gain benefit from the cooperative evolution of each class label, an additional term is added to Equation 6 that considers the information from other classes. Thus, the optimization problem in cooperative evolution is stated as:

$$\begin{aligned}&\min_{\alpha_i^r}\frac{1}{2}\sum_{i,j=1}^{n_r}\alpha_i^r\alpha_j^r\left\langle\mathbf{x}_i, \mathbf{x}_j\right\rangle - \sum_{i=1}^{n_r}\alpha_i^r + \frac{1}{n_r}\sum_{i=1}^{n_r}\mathcal{P}\left(\mathbf{x}_i\right) \\ &\text{subject to } 0 \leq \alpha_i^r \leq C\end{aligned} \tag{7}$$

where

$$\mathcal{P}\left(\mathbf{x}\right) = \begin{cases} Z & \text{if } Z > 0 \\ 0 & \text{if } Z \leq 0 \end{cases} : Z = 2 + \sum_{i=1}^{n_q}\alpha_i^q\left\langle\mathbf{x}_i, \mathbf{x}\right\rangle - \sum_{i=1}^{n_r}\alpha_i^r\left\langle\mathbf{x}_i, \mathbf{x}\right\rangle \tag{8}$$

where $q \in \{1, \ldots, M\} \setminus r$ is the index class with the largest activation.

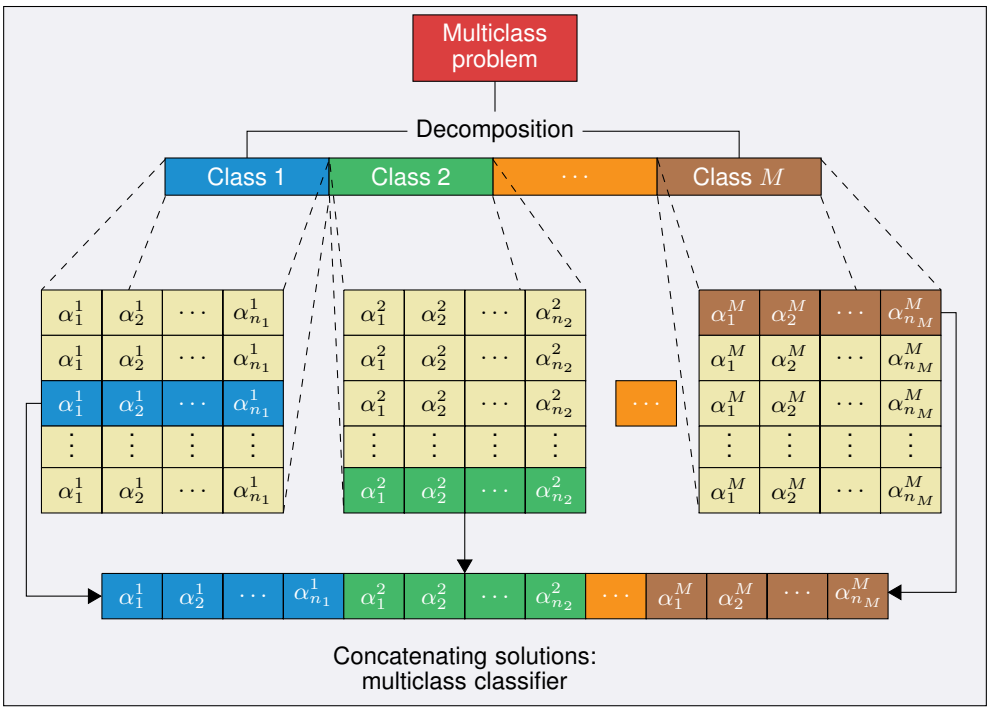


Fig. 2: Population scheme used in MC^{2}ESVM.

Thus, by adding this term, we penalize the errors that occur in the multiclass classification. In other words, MC^{2}ESVM punishes those solutions that do not work well together.

In the next section, we describe the representation of the solutions and the evolutionary operators used in MC^{2}ESVM.

B. Representation

In MC^{2}ESVM, each class works with the others in a cooperative fashion. As we have previously mentioned, each population manages the instances for a specific class. A representation scheme of the population is shown in Fig. 2.

Each population consists of P individuals. All populations share the same individual representation. Since the goal is to optimize the α vector from Equation 7, a real-valued representation is adopted. Moreover, the number of variables of each population depends on the number of instances available in the training set for each class[2]. By using this representation, the number of variables is not increased in the optimization task, as usually

[2]For example, in a three-class problem, with 50 instances of class 1, 80 instances of class 2, and 100 instances of class 3; the number of variables to optimize in subpopulations 1, 2, and 3 are equal to 50, 80, and 100, respectively.

happens with other methods. Moreover, all populations are evolved simultaneously and each of them deals with simpler problems.

The α vector of each individual in each population is randomly initialized. For doing so, each variable of an individual has a probability of 0.5 to take a value in the range $(0, C]$; otherwise, it takes a value of 0.

The individuals in each population are evolved. This is attained by using the differential evolution operator [28], which generates a new child solution as follows:

$$\overline{\alpha}_i^{r(s)} = \begin{cases} \alpha_i^{r(t)} + F \times \left(\alpha_i^{r(u)} - \alpha_i^{r(v)}\right) & \text{with prob. } CR, \\ \alpha_i^{r(s)} & \text{Otherwise} \end{cases} \tag{9}$$

where CR and F are two control parameters and s, t, u and v are the indexes for the current individual, which acts as the parent solution, and three randomly selected individuals from the rth population.

Finally, the child solution is added to the population for the next generation in the evolutionary process if and only if it improves the sth parent; otherwise, the current parent is kept.

C. Building the Multiclass SVM

Once the coevolutionary process is over, the next step is to build the multiclass classifier. This is done by selecting from each population the member that gets the highest score in the objective function and by concatenating the solutions. This can also be shown in Fig. 2.

It is worth noting that solutions with $\alpha_i^r > 0$ are considered the support vectors. These solutions represent the multiclass classifier learned by MC^{2}ESVM. A new instance $\mathbf{x}_t$ is classified as follows:

$$y_t = \underset{r}{\operatorname{argmax}} \sum_{i \in SV_r} \alpha_i^r \langle \mathbf{x}_i, \mathbf{x}_t \rangle \tag{10}$$

where SV_r represents the set of support vectors from the rth class.

D. Learning Nonlinear SVMs

The optimization problem presented in Equation 7 learns a linear function from the training data. For learning nonlinear functions, the so-called *kernel trick* is used with SVMs.

By using the kernel trick in MC^{2}ESVM, the inner product, $\langle \mathbf{x}_i, \mathbf{x} \rangle$, in Equations 7 and 10, is replaced by a Kernel function, $K(\mathbf{x}_i, \mathbf{x})$. Some commonly used kernel functions are the following [29]:

- Linear kernel: $K(\mathbf{x}_i, \mathbf{x}) = \langle \mathbf{x}_i, \mathbf{x} \rangle$
- Polynomial kernel: $K(\mathbf{x}_i, \mathbf{x}) = (\langle \mathbf{x}_i, \mathbf{x} \rangle + 1)^d$
- Radial basis function kernel: $K(\mathbf{x}_i, \mathbf{x}) = \mathrm{e}^{-\gamma \|\mathbf{x}_i - \mathbf{x}\|^2}$

where d, γ are adjustable parameters for the above kernel functions.

IV. Experimental Setup

In this section, we describe the experimental settings in our study. In Section IV-A, we present the datasets used in our experimental study. Section IV-B describes the algorithms that are used to compare the performance of MC^{2}ESVM. Finally, Section IV-C presents the performance measures and statistical tests used to assess each algorithm.

A. Datasets

A set of 25 datasets available in the KEEL repository [30], [31] are used in our experimental study. Since we want to assess the behavior of the proposed MC^{2}ESVM in multiclass problems, the datasets have been chosen based on the number of classes, i.e., those with more than two classes. Table I shows some characteristics of these datasets, such as the number of instances, the number of features, the imbalance rate (IR)[3] and the number of classes.

These datasets have been partitioned into 10 training/test subsets by using the k-fold cross validation technique. Furthermore, features have been pre-processed in order to have zero mean and unit standard deviation.

B. Considered Algorithms

Several multiclass extensions are used to compare the performance of MC^{2}ESVM with respect to them. The selection of these extensions is based on their availability on public frameworks, such as KEEL and MSVMpack [7]. Concretely, these extensions are the following:

[3]The IR is computed as the average of the IR of all pairwise classes.

TABLE I: Description of the datasets used in our study. For each dataset, we show the number of instances, the number of attributes, and the number of classes.

ID	Dataset	Atts.	Insts.	IR	Classes	ID	Dataset	Atts.	Insts.	IR	Classes
1	Automobile	25	203	5.69	6	14	Newthyroid	5	215	3.48	3
2	Balance	4	625	4.25	3	15	Penbased	16	10,992	1.05	10
3	Cleveland	13	303	3.87	5	16	Satimage	36	6,435	1.73	6
4	Contraceptive	9	1,473	1.55	3	17	Segment	19	2,310	1.00	7
5	Dermatology	34	366	2.17	6	18	Splice	60	3,190	1.77	3
6	Ecoli	7	336	15.27	8	19	Tae	5	151	1.04	3
7	Glass	9	214	3.60	6	20	Texture	40	5,500	1.00	11
8	Hayes-roth	4	132	1.47	3	21	Vehicle	18	846	1.05	4
9	Iris	4	150	1.00	3	22	Vowel	13	990	1.00	11
10	Led7digit	7	500	1.16	10	23	Wine	13	178	1.30	3
11	Lymphography	18	148	18.30	4	24	Yeast	8	1,484	11.65	10
12	Marketing	13	8,993	1.48	9	25	Zoo	16	101	3.20	7
13	Movement Libras	90	360	1.00	15						

- Decomposition strategies using the sequential minimal optimization (SMO) [32] implemented in KEEL:
  - OVO
  - OVA
- Single machine methods available at MSVMpack:
  - $MSVM^2$
  - MSVM-CS
  - MSVM-LLW
  - MSVM-WW

For all cases, a radial basis function (RBF) kernel is used, because it is one of the most popular and effective kernels used with SVMs [33]. Regarding the hyper-parameter configuration of each method, and for the sake of allowing a fair comparison, the values of the hyper-parameters are tuned for each method in each dataset, as it is suggested in [33], [34]. For doing so, we used a random search for the hyper-parameters optimization [35], by randomly sampling 100 points for the kernel's parameter in the range $\gamma = [2^{-10}, 2^2]$, and for the regularization parameter $C = [2^{-4}, 2^{10}]$ for all methods. Each configuration is tested and the one with the best score in accuracy is chosen for each dataset.

It is worth noting that during the experiments, some configurations for the single machine methods available at MSVMpack were unable to converge to an optimal solution. Thus, we have modified the code in order to allow a maximum number of evaluations of the objective function, which was fixed to 200,000. This limit is also set for all methods. Furthermore, in the case of MC^2ESVM, the convergence criterion is defined as having an improvement in the best solution after 10 iterations lower than 0.001. MC^2ESVM also requires some additional parameters, which are common in evolutionary algorithms. These parameters were set as

follows:

- Population size = 20
- Crossover Rate (CR) = 0.8
- Differential weight (F) = 1.6

We have further considered the set of common learning algorithms: Random Forest (RF), Multi-Layer Perceptron (MLP), K-Nearest Neighbor (KNN), and Naïve Bayes (NB), for comparison. To allow a fair comparison, the hyper-parameters of these methods are tuned. In RF, the number of trees is adjusted in the range $[1, 100]$; in MLP, the learning rate is tuned in the range $[0, 1]$ and the number of neurons $[1, 100]$; KNN, the neighborhood size is determined in $[1, 10]$.

C. Performance Measures

We have considered two different metrics for measuring the performance of MC^{2}ESVM and the reference methods. The set of metrics are accuracy and Cohen's kappa, which are described next:

- **Accuracy** (Acc) is a common metric for assessing the performance of supervised learning algorithms. It indicates the ratio of samples that are correctly classified, i.e.,

$$Acc = \frac{1}{N} \sum_{i=1}^{M} TP_i \tag{11}$$

  where TP_i is the number of correctly classified samples from class i.

- **Cohen's kappa** ($\mathcal{K}$) measures the degree of agreement between two observations: the predicted class and the correct one. An easy way of computing Cohen's kappa is as follows:

$$\mathcal{K} = \frac{N \sum_{i=1}^{M} TP_i - \sum_{i=1}^{M} P_i T_i}{N^2 - \sum_{i=1}^{M} P_i T_i} \tag{12}$$

  where P_i is the number of predicted samples as class i and T_i is the number of samples from class i.

  Cohen's kappa ranges from -1, indicating total disagreement, through 0 (random classification), to 1, which indicates a perfect agreement.

In order to support the comparisons, a set of non-parametric statistical tests is used. Non-parametric tests are widely recommended for a safe and robust comparison of multiple classifiers over multiple datasets by [36]–[38]. In this study, we have used the Friedman

TABLE II: Comparison between MC^{2}ESVM and the SVM formulations for multiclass problems.

Method	Acc	$\mathcal{K}$
MC^{2}ESVM	$\mathbf{0.8167 \pm 0.1787}$	$\mathbf{0.7424 \pm 0.2437}$
OVA	0.7920 ± 0.1897	0.7044 ± 0.2683
OVO	0.8003 ± 0.1868	0.7031 ± 0.2880
MSVM^2	0.7682 ± 0.1886	0.6648 ± 0.2697
MSVM-CS	0.7958 ± 0.1970	0.7157 ± 0.2651
MSVM-LLW	0.7795 ± 0.1967	0.6715 ± 0.2988
MSVM-WW	0.7808 ± 0.1949	0.6875 ± 0.2705

Aligned Ranks test to compare among multiple algorithms, and the Holm's procedure is used to find out which algorithms are distinctive. In all cases, the significance level is set to $\alpha = 0.05$. A description of these tests can be found in [37], [38].

V. Experimental Results and Their Analysis

This section presents the results obtained in our experimental study and analyzes them. In Section V-A, we report the results of MC^{2}ESVM and the reference methods using the set of multiclass datasets. Section V-B analyzes and compares the support vectors found by each method. Section V-C, we assess the scalability of the proposed MC^{2}ESVM with respect to the number of classes and the number of samples. Finally, in Section V-D we analyze the stability of MC^{2}ESVM.

A. Classification Performance

The aim of this study is twofold. First, comparing the performance of MC^{2}ESVM with respect to other SVM formulations for multiclass problems. Second, comparing against standard learning algorithms.

1) Comparing with SVMs Multiclass Extensions

In this first part of our study, we assess the performance of MC^{2}ESVM when it is compared with several extensions of SVMs for multiclass problems. Table II[4] shows the average results obtained from the 25 datasets.

Table III shows the ranking obtained by Friedman's Aligned Ranks both with accuracy score (Acc) and Cohen's Kappa ($\mathcal{K}$). We further show the adjusted p-value with the Holm's

[4]The detailed results on each dataset and the hyper-parameters values of each method are provided as supplementary material. The supplementary material and source code are available at http://ccc.inaoep.mx/~arosales/resources/MC2ESVM.tar.gz.

TABLE III: Average rankings of the methods computed with Friedman Aligned Ranks (FAR) and Holm's adjusted p-values (p_{Holm}).

Method	Acc		$\mathcal{K}$	
	FAR	p_{Holm}	FAR	p_{Holm}
MC^{2}ESVM	50.28	—	51.20	—
OVO	70.36	0.1611	75.30	0.1128
MSVM-CS	83.82	**0.0385**	78.54	0.1128
OVA	92.78	**0.0091**	94.56	**0.0074**
MSVM^2	103.44	**0.0008**	102.94	**0.0015**
MSVM-WW	105.04	**0.0007**	101.12	**0.0020**
MSVM-LLW	110.28	**0.0002**	112.34	**0.0001**

test (p_{Holm}). Note that MC^{2}ESVM is set as a control method because the purpose of our study is to compare the performance of our proposal against the rest.

Observing the results in Tables II and III, we can highlight the following:

- The worst performance of MC^{2}ESVM is obtained in the Balance and the Vehicle datasets, which have three and four classes, respectively. On the other hand, its best performance is shown in the Cleveland, Ecoli, and Zoo datasets, which have at least five classes each.
- It is worth noting that, in general, MC^{2}ESVM significantly performs better than reference methods in datasets with imbalance rates greater than 1.5 and with five or more classes. This may be explained due to the fact that OVO with a large number of classes, significantly increases the number of binary classifiers, leading to an ensemble with a more complex decision function. OVA, on the other hand, artificially makes higher this imbalance.
- On well-balanced problems, the performance of MC^{2}ESVM and reference methods are quite similar, regardless of the number of classes.
- MC^{2}ESVM statistically outperforms most of the SVM formulations for handling multiclass problems. In fact, it is statistically better in five out of six methods for the accuracy score and in four out of six methods for Cohen's Kappa statistic, under the considered level of $\alpha = 0.05$.
- MSVM-CS and the OVO decomposition are clearly the most competitive multiclass SVMs for the proposed MC^{2}ESVM. These competitive performances can also be noted in the lack of a statistically significant difference when Cohen's Kappa is considered.
- The difference between MC^{2}ESVM and OVO in accuracy is marginal, but in Cohen's

TABLE IV: Average Friedman Aligned Ranks (FAR) and Holm's adjusted p-values (p_{Holm}) for the best methods.

Method	Acc		$\mathcal{K}$	
	FAR	p_{Holm}	FAR	p_{Holm}
MC^{2}ESVM	27.16	—	27.96	—
OVO	39.92	**0.0385**	41.74	**0.0254**
MSVM-CS	46.92	**0.0027**	44.33	**0.0161**

Kappa, MC^{2}ESVM clearly outperforms OVO. This is an interesting point to observe because the hyper-parameters for each method were done by considering the accuracy as the main criterion. Thus, MC^{2}ESVM is not overfitted to this criterion.

- Except for MSVM-CS, the rest of the multiclass SVMs based on modifying the objective function showed a low performance. This may be due to the fact that they have to deal with a larger optimization problem than those based on decomposition.
- OVO can be highlighted as the best method based on decomposing the problem into multiple binary classification problems, while MSVM-CS is an outstanding method from those based on modifying the optimization problem.

The Holm's test has reported no statistically significant difference between MC^{2}ESVM and OVO neither between MC^{2}ESVM and MSVM-CS, when the multiple comparison is done by considering all methods. This may be due to the number of algorithms in the comparison and the fact that those algorithms have influence on the rank computation and also in the post-hoc [39]. Therefore, we have thoroughly inspected these three algorithms by comparing them. With this aim, Table IV shows the statistical comparison when considering these methods. MC^{2}ESVM is again considered as a control method in the test. Based on this more focused test, we can note that indeed the differences between MC^{2}ESVM and the reference methods (OVO and MSVM-CS) are statistically significant at the considered level.

Fig. 3 graphically depicts a comparison of the computational time for each method. This figure represents the probability that a given method learns the multiclass SVM in a given amount of time. From it, we can note:

- Both OVO and OVA are the best ones, and have virtually the same performance.
- MC^{2}ESVM is the second best one, requiring at most, 30 seconds for solving each benchmark problem.
- Single machine methods are clearly the slowest ones. This is due to the fact that they

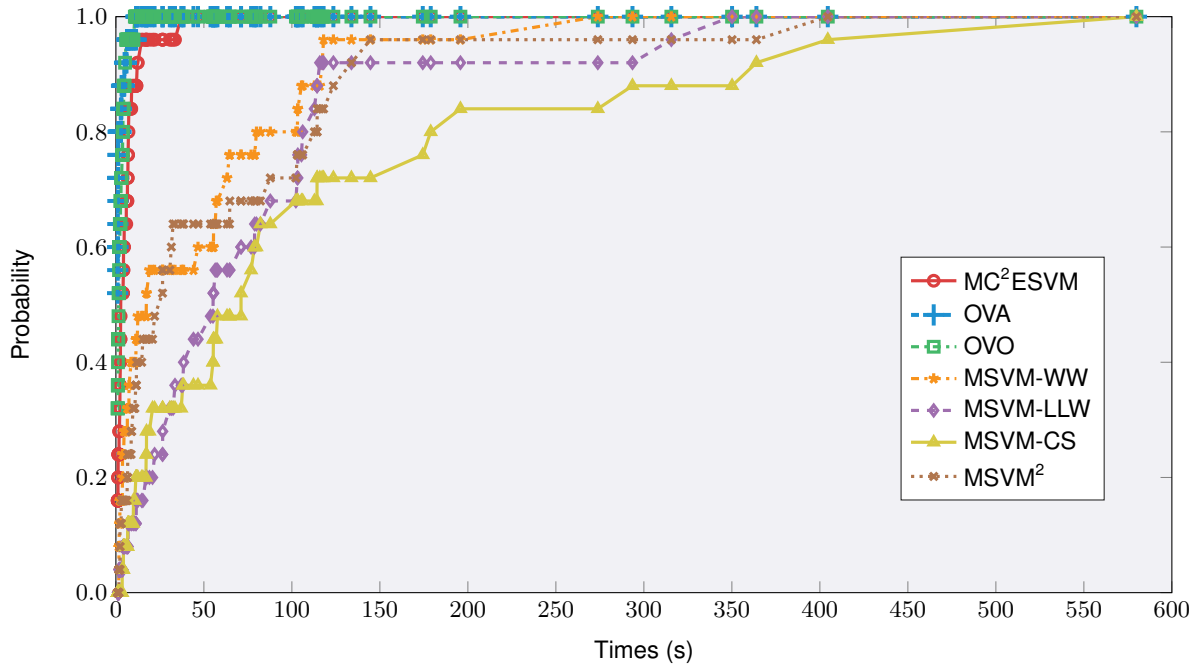


Fig. 3: Probability of each method to learn a Multiclass SVM classifier in a given amount of time.

TABLE V: Comparison between MC^{2}ESVM and RF, MLP, KNN, and NB.

Method	Acc	$\mathcal{K}$
MC^{2}ESVM	$\mathbf{0.8167 \pm 0.1787}$	$\mathbf{0.7424 \pm 0.2437}$
RF	0.8048 ± 0.1742	0.7229 ± 0.2417
MLP	0.7932 ± 0.1837	0.7164 ± 0.2511
KNN	0.7607 ± 0.2020	0.6638 ± 0.2873
NB	0.7238 ± 0.1934	0.6319 ± 0.2577

deal with a larger optimization problem.

- In the best case, single machine methods required around 280 seconds to ensure solving each dataset.
- Among all SVMs multiclass extensions, MSVM-CS is the worst one in terms of computational time.

2) Comparing with other Learning Algorithms

In this section, our goal is to contrast the performance of MC^{2}ESVM with common learning algorithms. To this end, RF, MLP, KNN, and NB are chosen as reference methods.

Table V shows the reported results obtained by each method on each dataset and Table VI shows the ranking for each algorithm computed with the Friedman's Aligned Ranks method

TABLE VI: Average rankings of the methods computed with Friedman Aligned Ranks (FAR) and Holm's adjusted p-values (p_{Holm}).

Method	Acc		$\mathcal{K}$	
	FAR	p_{Holm}	FAR	p_{Holm}
MC^{2}ESVM	35.28	—	37.60	—
RF	53.76	0.0713	56.62	0.0634
MLP	61.04	**0.0239**	59.26	0.0634
KNN	77.22	$<$ **0.010**	78.42	$<$ **0.010**
NB	87.70	$<$ **0.010**	83.10	$<$ **0.010**

and the adjusted p-values with the Holm's test. Observing these results, we stress the following:

- MC^{2}ESVM achieves, on average, the highest scores both on the accuracy and Kappa statistics metrics.
- Among the reference methods, RF is the most competitive, since it has no statistically significant difference with respect to MC^{2}ESVM.
- NB is generally the worst one. The exception is on the Splice dataset, where it achieves the best performance. This may be explained due to the inductive bias of this algorithm, which assumes independence between attributes.

B. Analyzing the Support Vectors

The aim of this study is twofold. First, by comparing the number of support vectors generated by each method, this can give insights about the complexity of the learned model. Second, we show the effect in the decision function for each method.

For the first case, Fig. 4 graphically depicts the distribution of the number of support vectors resulting from each method. Based on it, we can stress the following for this first part:

- MC^{2}ESVM, OVO, OVA, MSVM-WW, and MSVM-LLW are quite similar in terms of the number of learned support vectors.
- MSVM-CS and MSVM^2 are the worst performers under this criterion. This can be explained due to the fact that these methods deal with a large number of variables during the optimization process. Furthermore, MSVM-CS does not consider the bias in its formulation, which can also explain this behavior.

The second goal of this study is to contrast the decision boundaries learned by each method. For illustrative purposes, we have generated an artificial dataset with two features

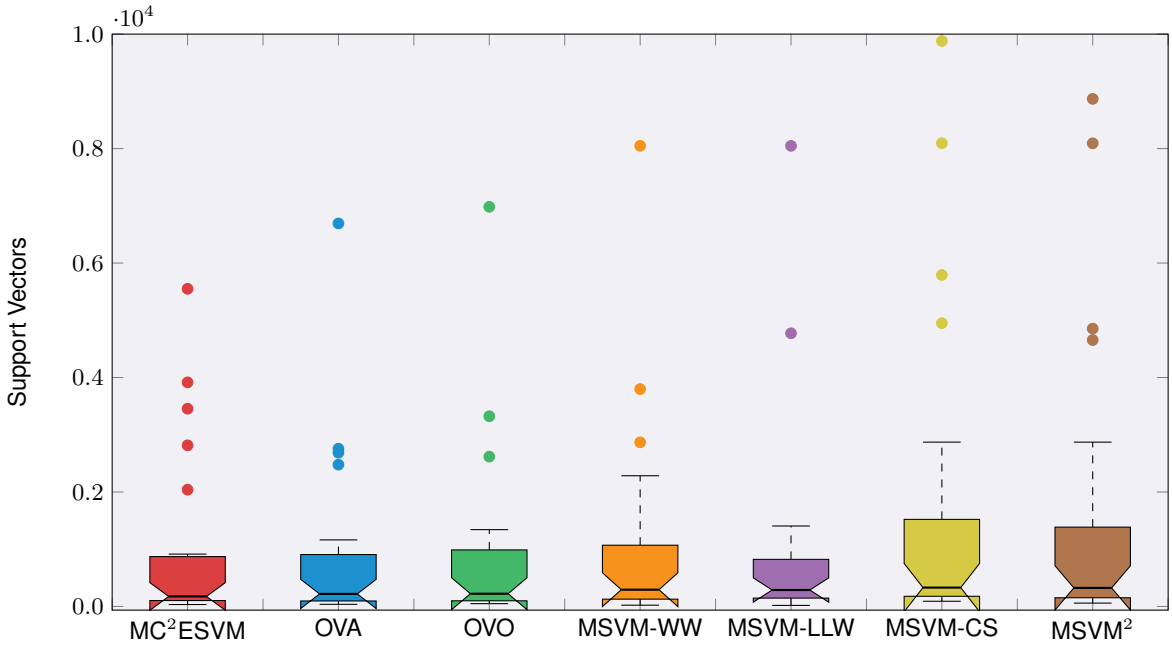
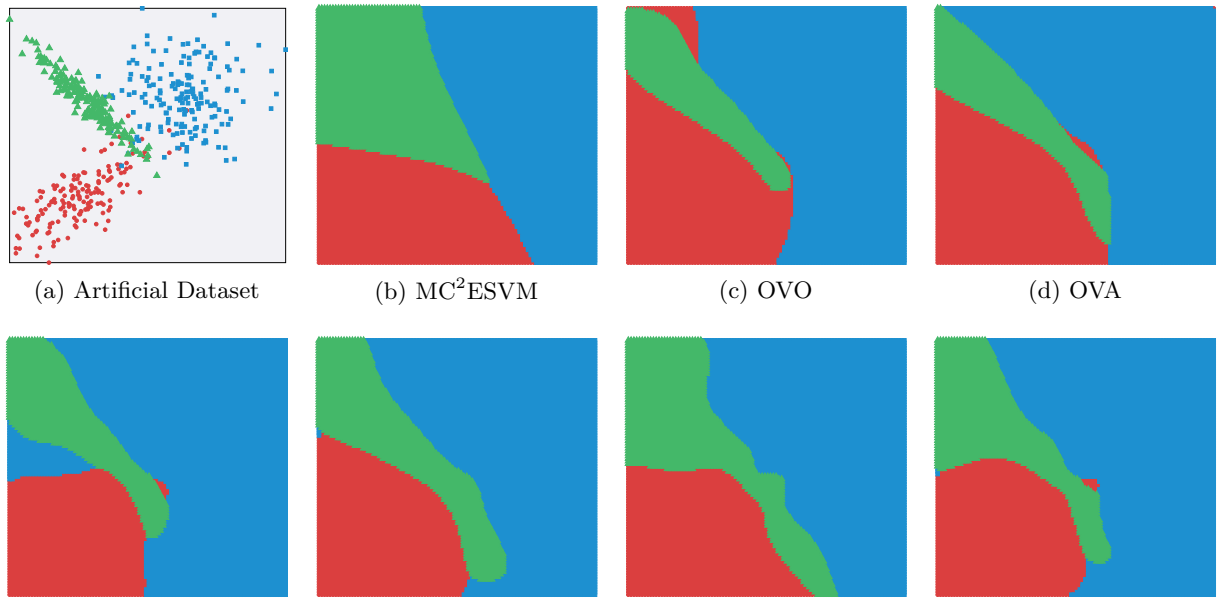


Fig. 4: Boxplot for the number of support vectors generated by each method.

and three classes. This artificial dataset is best known as the Madelon dataset and it is generated following the methodology proposed in [40]. A noise level of 10% is induced in this dataset.

Fig. 5 graphically depicts the decision regions that are learned by each method. Fig. 5a shows the training points used to fit the model's parameters. Figs. 5b to 5h show the region for each class generated by each method. From this, the following can be noted:

- MC^{2}ESVM seems to better capture the regions of each class, by exhibiting well-defined regions for each. Moreover, MC^{2}ESVM is able to learn a simpler function, which does not show a wiggle shape.
- OVO and OVA show overlapped regions. This may be explained due to the fact that both OVO and OVA work by solving several binary classification problems, independently, which leads to regions of uncertainty.
- In general, MSVM-CS and MSVM^2 generate more complex decision boundaries than the other methods. This is also consistent with the fact that they are the ones with the highest number of support vectors, which can be considered a measure of the model's complexity.



(a) Artificial Dataset (b) MC^{2}ESVM (c) OVO (d) OVA

(e) MSVM-WW (f) MSVM-LLW (g) MSVM-CS (h) MSVM^2

Fig. 5: Decision boundary for each of the SVM multiclass extensions.

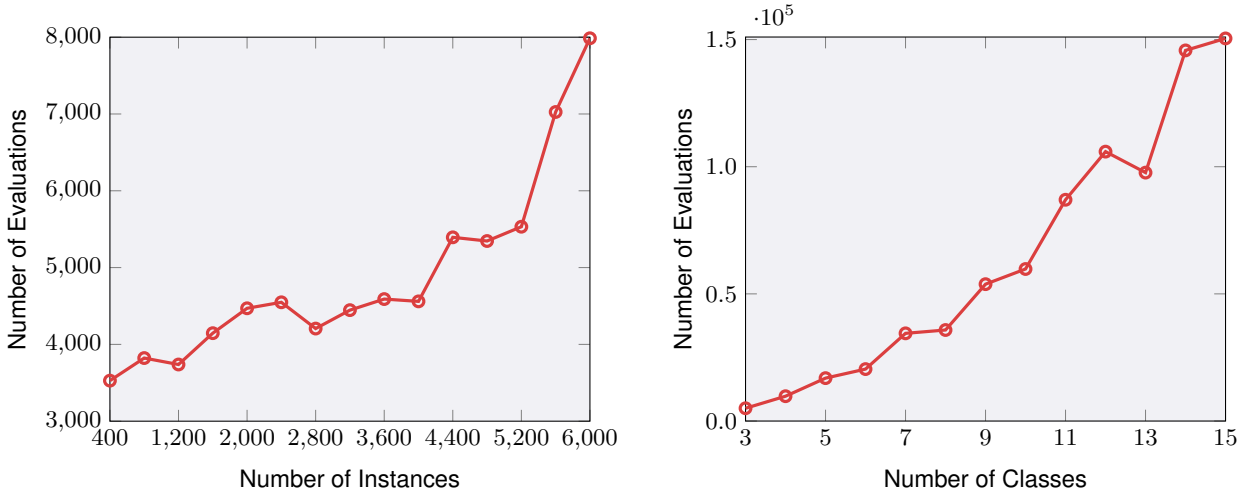
- MSVM-WW has the highest overlapped area. This is also consistent with its low performance in the classification task.

C. Analyzing the Scalability

The aim of this study is to assess the scalability of MC^{2}ESVM with respect to the number of classes and the number of training instances. In order to perform these studies, we have generated a set of artificial datasets, following the methodology proposed in [40], as in the previous study, but with the following considerations:

- For assessing the scalability with respect to the number of classes, the datasets are generated by varying the number of classes from 3 to 15. In all cases, a set of 25 features and 1,500 samples are fixed.
- For assessing the scalability with respect to the number of instances, the number of classes is fixed in 3 and the number of features is kept in 25. The number of instances for each dataset ranges from 800 to 6,000, with a step size of 400.

Fig. 6 depicts the behavior of MC^{2}ESVM under these conditions. Based on these figures, we can remark the following:



(a) Instances

(b) Classes

Fig. 6: Scalability when either the number of classes or instances increases.

- The number of evaluations increases as either the number of instances or the number of classes is increased. This is an expected result, since MC^{2}ESVM deals with more complex problems.
- The number of classes seems to be more harmful in the scalability of MC^{2}ESVM. This may be due to the fact that, as the number of classes grows, the complexity of the decision boundaries is increased, making harder the recognition. Another factor that also contributes to this is the cooperative approach, which decomposes the problem, having a population for each class. Thus, the number of evaluations of the objective function increases, at least, by a factor of the number classes. However, since each population works with the instances of the corresponding class, the computation requires a lower computational cost.

D. Analyzing the Stability

Due to the stochastic nature of the cooperative evolutionary algorithm in MC^{2}ESVM, another interesting issue is concerned with its stability. Thus, in order to assess it, we have carried out two analyzes. On the one hand, the classification performance over different replications is determined. On the other hand, we examine the parameters of the evolutionary algorithm.

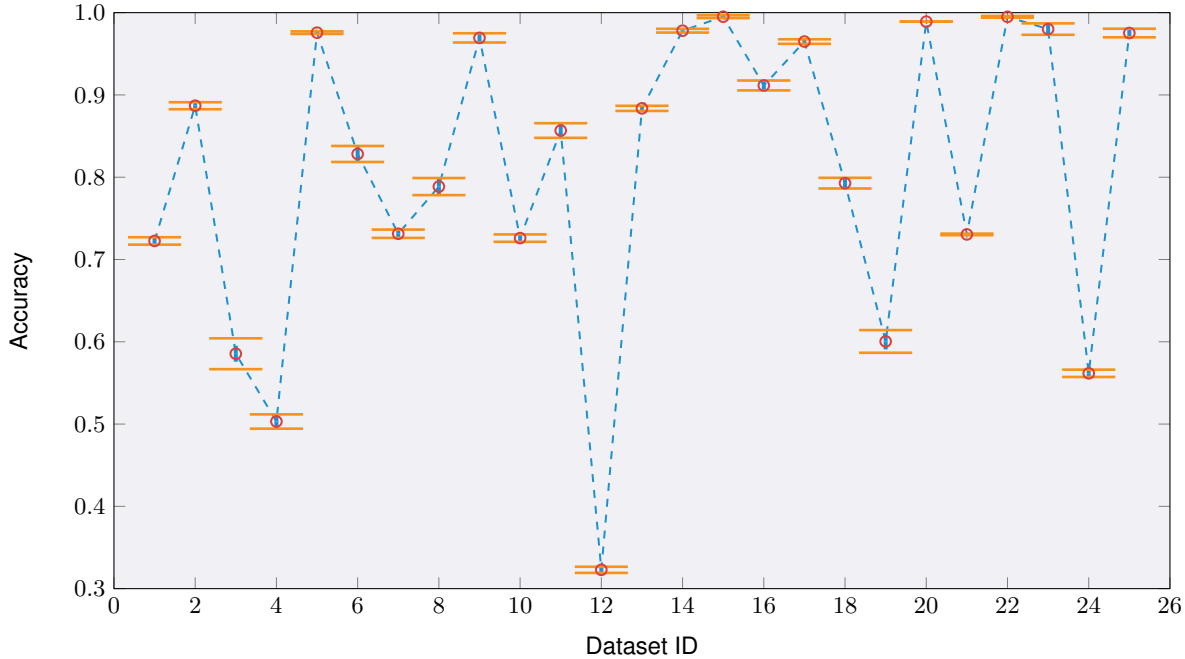


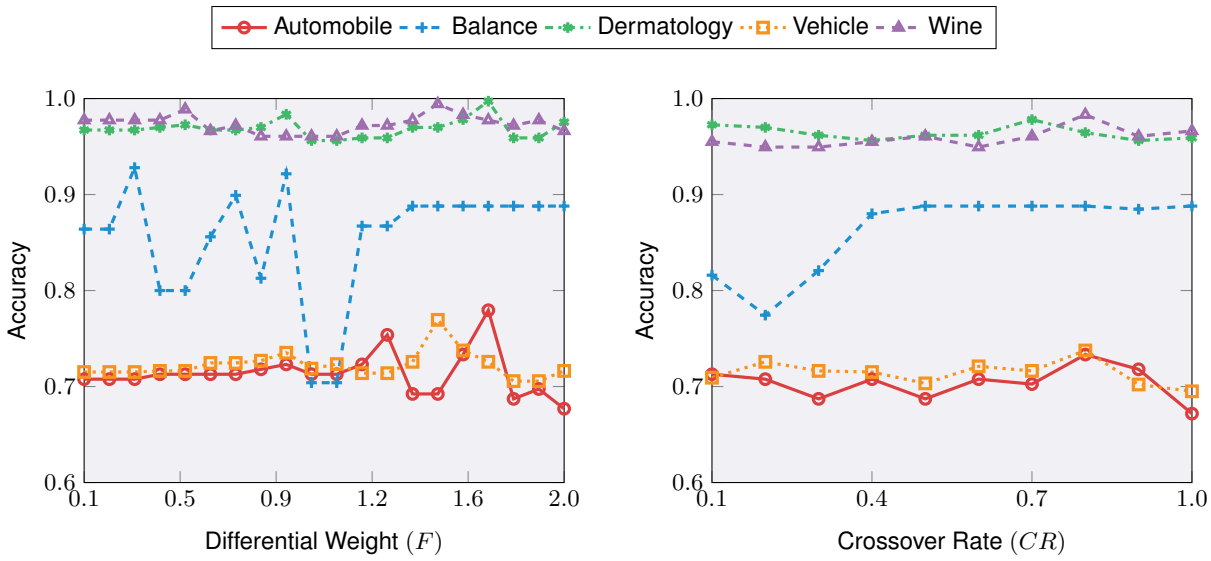
Fig. 7: Mean and standard deviation of the 15 replications of MC^{2}ESVM for each dataset.

For the first part, we have performed 15 replications of MC^{2}ESVM over each dataset. Fig. 7 shows the mean and standard deviation. From it, we can note how MC^{2}ESVM is able to get virtually the same performance in all replications. Thus, MC^{2}ESVM exhibits a stable behavior, showing a low variance in the classification performance.

The remaining issue to analyze is the evolutionary parameters. Since MC^{2}ESVM uses differential evolution operators, we have tested it by varying the value of the differential weight (F) and the crossover rate (CR). Fig. 8 shows the performance on five representative datasets when these parameters are varied. Fig. 8a shows the effect of varying F in the range [0.1,2.0] and Fig. 8b for CR in the range [0.1,1.0]. Based on these results, a value in the range [1.5,1.7] for the F parameter and in the range [0.6,0.9] for the CR parameter, are good recommendations.

VI. Conclusions

This paper introduced a multiclass classification approach based on the cooperative evolution of support vector machines, called MC^{2}ESVM. The method is general in the sense that it subsumes the decomposition-based and the single-machine methods. MC^{2}ESVM is intuitive and takes advantages of the cooperative evolution to decompose the classification



(a) Differential Weight (F)

(b) Crossover Rate (CR)

Fig. 8: Performance of MC^{2}ESVM under different parameters values.

problem, such that each subproblem faces the learning of the support vectors to the specific class, but acting in a cooperative fashion by considering the information of the other classes.

Unlike decomposition-based extensions, MC^{2}ESVM is able to capture in a single model the multiclass classification problem, leading to a simpler decision function than the one obtained by fusing multiple binary classifiers. In contrast with single machine methods, MC^{2}ESVM does not increase the number of variables to be optimized, deriving in a simpler optimization problem with no additional constraints. These features make the proposed extension a more flexible approach.

The experimental evaluation over a set of 25 common benchmark datasets shows that MC^{2}ESVM is able to outperform most of the multiclass SVM extensions. This claim is supported by a set of non-parametric tests with a level of significance of $\alpha = 0.05$. The experimental results have also revealed that MC^{2}ESVM performs better on problems with an imbalance rate higher than 1.5 and with a number of classes greater than five. MSVM-CS excels as the most competitive among the single machine methods. OVO stands as the best method from those based on binary decomposition. This finding is also confirmed in [5]. A focused analysis of these two prominent methods and MC^{2}ESVM revealed that the latter is able to statistically perform better. MC^{2}ESVM does not seem to be overfitted

to the hyper-parameters optimized by the accuracy criterion. In fact, it shows a greater improvement on Cohen's Kappa when it is compared to the other methods. Furthermore, the computational time required by MC^{2}ESVM is not as high as that of existing single machine methods. In fact, it requires a reasonable time for learning a model, similar to that required by decomposition-based methods. Thus, it has benefitted from the advantages of each approach.

Another interesting conclusion is that MC^{2}ESVM showed to be able to learn simpler functions than most of the methods based on adapting the optimization problem. OVO and OVA, on the other hand, suffer from generating regions whit high uncertainty. Finally, the most important criterion for scalability of MC^{2}ESVM is the number of classes, which increases the number of evaluations in the optimization.

An advantage of EAs is that they will allow SVM to handle non-positive semidefinite kernels. This type of kernel leads to a non-convex optimization problem, narrowing their applicability to the quadratic solver of SVMs. As part of our future work, we will analyze the performance of MC^{2}ESVM on non-positive semidefinite kernels. Another interesting path for future research is to deepen into the interaction scheme of solutions of different populations. Finally, exploiting the parallelizable nature of EAs to handle the so-called Big Data problems is another interesting topic for future research.

Acknowledgments

This research has been supported by FEDER and by the Spanish National Research Project TIN2017-89517-P, by the Regional Andalusian project P11-TIC-7765 and by CONACyT under projects 221551 and 241461.

References

[1] V. N. Vapnik, *The Nature of Statistical Learning Theory.* New York, NY, USA: Springer-Verlag New York, Inc., 1995.

[2] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems?" *J. Mach. Learn. Res.*, vol. 15, pp. 3133–3181, Oct. 2014.

[3] A. Demontis, M. Melis, B. Biggio, G. Fumera, and F. Roli, "Super-sparse learning in similarity spaces," *IEEE Comput. Intell. M.*, vol. 11, no. 4, pp. 36–45, Nov. 2016.

[4] N. I. Sapankevych and R. Sankar, "Time series prediction using support vector machines: A survey," *IEEE Comput. Intell. M.*, vol. 4, no. 2, pp. 24–38, May. 2009.

[5] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes," *Pattern Recogn.*, vol. 44, no. 8, pp. 1761–1776, Aug. 2011.

[6] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," *J. Mach. Learn. Res.*, vol. 5, pp. 101–141, Jan. 2004.

[7] F. Lauer and Y. Guermeur, "MSVMpack: A multi-class support vector machine package," *J. Mach. Learn. Res.*, vol. 12, pp. 2293–2296, Jul. 2011.

[8] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE T. Syst. Man Cy. C*, vol. 38, no. 3, pp. 397–415, May. 2008.

[9] A. Mukhopadhyay, U. Maulik, S. Bandyopadhyay, and C. A. Coello Coello, "A survey of multiobjective evolutionary algorithms for data mining: Part I," *IEEE T. Evol. Comput.*, vol. 18, no. 1, pp. 4–19, Feb. 2014.

[10] ——, "Survey of multiobjective evolutionary algorithms for data mining: Part II," *IEEE T. Evol. Comput.*, vol. 18, no. 1, pp. 20–35, Feb. 2014.

[11] C. Chatelain, S. Adam, Y. Lecourtier, L. Heutte, and T. Paquet, "A multi-model selection framework for unknown and/or evolutive misclassification cost problems," *Pattern Recogn.*, vol. 43, no. 3, pp. 815 – 823, Mar. 2010.

[12] A. Rosales-Pérez, S. García, J. A. Gonzalez, C. A. Coello Coello, and F. Herrera, "An evolutionary multiobjective model and instance selection for support vector machines with pareto-based ensembles," *IEEE T. Evol. Comput.*, vol. 21, no. 6, pp. 863–877, Dec. 2017.

[13] N. Verbiest, J. Derrac, C. Cornelis, S. García, and F. Herrera, "Evolutionary wrapper approaches for training set selection as preprocessing mechanism for support vector machines: Experimental evaluation and support vector analysis," *Appl. Soft. Comput.*, vol. 38, pp. 10 – 22, Jan. 2016.

[14] M. L. D. Dias and A. R. R. Neto, "Evolutionary support vector machines: A dual approach," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 2185–2192.

[15] I. Mierswa, "Evolutionary learning with kernels: A generic solution for large margin problems," in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '06, 2006, pp. 1553–1560.

[16] L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, L. D. Jackel, Y. LeCun, U. A. Muller, E. Sackinger, P. Simard, and V. Vapnik, "Comparison of classifier methods: A case study in handwritten digit recognition," in *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, vol. 2, 1994, pp. 77–82.

[17] U. H. G. Kreßel, "Pairwise classification and support vector machines," in *Advances in Kernel Methods*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA, USA: MIT Press, 1999, pp. 255–268.

[18] J. C. Platt, N. Cristianini, and J. Shawe-Taylor, "Large margin dags for multiclass classification," in *Advances in Neural Information Processing Systems 12*, S. A. Solla, T. K. Leen, and K. Müller, Eds. MIT Press, 2000, pp. 547–553.

[19] J. Weston and C. Watkins, "Multi-class support vector machines," Royal Holloway, University of London, Tech. Rep. CSD-TR-98-04, 1998.

[20] Y. Lee, Y. Lin, and G. Wahba, "Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data," *J. Am. Stat. Assoc.*, vol. 99, pp. 67–81, Jun. 2004.

[21] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *J. Mach. Learn. Res.*, vol. 2, pp. 265–292, Mar. 2002.

[22] Y. Guermeur and E. Monfrini, "A quadratic loss multi-class SVM for which a radius-margin bound applies," *Informatica*, vol. 22, no. 1, pp. 73–96, Jan. 2011.

[23] G. J. J. Van Den Burg and P. J. F. Groenen, "GenSVM: A generalized multiclass support vector machine," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 7964–8005, Jan. 2016.

[24] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE T. Neural Network.*, vol. 13, no. 2, pp. 415–425, Mar. 2002.

[25] R. P. Wiegand, "An analysis of cooperative coevolutionary algorithms," Ph.D. dissertation, George Mason University, Fairfax, VA, USA, 2004.

[26] C. D. Rosin and R. K. Belew, "New methods for competitive coevolution," *Evol. Comput.*, vol. 5, no. 1, pp. 1–29, Mar. 1997.

[27] M. A. Potter and K. A. De Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evol. Comput.*, vol. 8, no. 1, pp. 1–29, Mar. 2000.

[28] K. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[29] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

[30] J. Alcalá, L. Sánchez, S. García, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas, J. C. Fernández, and F. Herrera, "KEEL: A software tool to assess evolutionary algorithms for data mining problems," *Soft Comput.*, vol. 13, no. 3, pp. 307–318, Oct. 2008.

[31] J. Alcalá, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Mult.-Valued Log. S.*, vol. 17, no. 2-3, pp. 255–287, Jun. 2011.

[32] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA, USA: MIT Press, 1999, pp. 185–208.

[33] A. J. Bagnall and G. C. Cawley, "On the use of default parameter settings in the empirical evaluation of classification algorithms," *CoRR*, vol. abs/1703.06777, Mar. 2017.

[34] J. Wainer and G. Cawley, "Empirical evaluation of resampling procedures for optimising SVM hyperparameters," *J. Mach. Learn. Res.*, vol. 18, no. 15, pp. 1–35, Jan. 2017.

[35] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Feb. 2012.

[36] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[37] S. García and F. Herrera, "An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons," *J. Mach. Learn. Res.*, vol. 9, pp. 2677–2694, Dec. 2008.

[38] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inform. Sciences*, vol. 180, no. 10, pp. 2044–2064, May. 2010.

[39] A. Benavoli, G. Corani, and F. Mangili, "Should we really use post-hoc tests based on mean-ranks?" *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 152–161, Jan. 2016.

[40] I. Guyon, S. Gunn, A. B. Hur, and G. Dror, *Design and Analysis of the NIPS2003 Challenge.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 237–263.