Cooperative Multi-Objective Evolutionary Support Vector Machines for Multiclass Problems

Alejandro Rosales-Pérez
Tecnologico de Monterey
Monterrey, Mexico
arosalesp85@gmail.com

Andres E. Gutierrez-Rodríguez
Tecnologico de Monterey
Monterrey, Mexico
aegr82@itesm.mx

Salvador García
University of Granada
Granada, Spain
salvagl@decsai.ugr.es

Hugo Terashima-Marín
Tecnologico de Monterey
Monterrey, Mexico
terashima@itesm.mx

Carlos A. Coello Coello
CINVESTAV-IPN
Mexico City, Mexico
ccoello@cs.cinvestav.mx

Francisco Herrera
University of Granada
Granada, Spain
herrera@decsai.ugr.es

ABSTRACT

In recent years, evolutionary algorithms have been found to be effective and efficient techniques to train support vector machines (SVMs) for binary classification problems while multiclass problems have been neglected. This paper proposes CMOE-SVM: Cooperative Multi-Objective Evolutionary SVMs for multiclass problems. CMOE-SVM enables SVMs to handle multiclass problems via coevolutionary optimization, by breaking down the original M-class problem into M simpler ones, which are optimized simultaneously in a cooperative manner. Furthermore, CMOE-SVM can explicitly maximize the margin and reduce the training error (the two components of the SVM optimization), by means of multi-objective optimization. Through a comprehensive experimental evaluation using a suite of benchmark datasets, we validate the performance of CMOE-SVM. The experimental results, supported by statistical tests, give evidence of the effectiveness of the proposed approach for solving multiclass classification problems.

CCS CONCEPTS

• **Theory of computation → Support vector machines**;

KEYWORDS

Support Vector Machines, Cooperative Coevolutionary Algorithms, Multi-Objective Optimization

ACM Reference Format:
Alejandro Rosales-Pérez, Andres E. Gutierrez-Rodríguez, Salvador García, Hugo Terashima-Marín, Carlos A. Coello Coello, and Francisco Herrera. 2018. Cooperative Multi-Objective Evolutionary Support Vector Machines for Multiclass Problems. In *GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205524

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5618-3/18/07…$15.00
https://doi.org/10.1145/3205455.3205524

1 INTRODUCTION

Support Vector Machines (SVMs) [24] are supervised learning algorithms with strong theoretical foundations. The ultimate goal of an SVM is to find the hyperplane that maximizes the separation margin between two classes, described by the so-called *support vectors*. In spite of the effectiveness of SVMs in binary classification, real-world problems often require discriminating among more than two classes.

Common extensions to enable SVMs to work with multiclass problems are categorized into decomposition strategies and single machine methods. Decomposition strategies break down the multiclass problem into multiple binary ones, which are solved independently without considering the interactions with others. Single machine methods, on the other hand, reformulate the function to be optimized, allowing to capture in a single model the multiclass decision function. Nonetheless, single machine methods result in larger and more complex optimization problems.

Recent studies have successfully integrated evolutionary algorithms (EAs) into SVMs [4, 20, 21, 25]. One integration consists in replacing the quadratic programming solver with an EA to train SVMs [7, 15], allowing us to work with non-positive semidefinite kernels[1]. Despite these advances, they have focused on the binary classification problem, neglecting the multiclass ones.

We propose here CMOE-SVM: Cooperative Multi-Objective Evolutionary SVMs for multiclass problems. CMOE-SVM aims at putting forward at the intersection of EAs and SVMs, by taking advantage of the benefits of cooperative coevolutionary algorithms to decompose the multiclass problem into several subproblems and optimizing them in a cooperative fashion. Thus, each subproblem interacts with others and the multiclass problem is captured in a single model. The additional benefits of EAs for solving multi-objective problems can lead to enhancements by explicitly and simultaneously optimizing the margin and controlling the over-fitting during the training stage. The main contributions of this paper are the following:

- An extension to SVMs, which has enabled them to learn multiclass classifiers via coevolutionary optimization.
- A derivation of the multi-objective optimization problem that learns the class-specific support vectors, by considering the information from other classes. The multi-objective formulation has the inherent advantage of managing different

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

costs for the penalty term without requiring to explicitly specify it.

We assess the performance of CMOE-SVM with a set of 20 multiclass classification datasets. We first compare it with decomposition strategies. Second, we compare with respect to single machine methods. Third, we contrast the performance of the proposed method against standard learning algorithms. Finally, we compare in terms of the computational time with respect to the multiclass SVMs extensions. Our experimental results show its effectiveness in solving the classification task. These findings are supported by a set of non-parametric statistical tests.

The remainder of this paper is organized as follows. Section 2 describes some preliminary concepts related to the main extension to multiclass SVMs, coevolutionary optimization, and multi-objective optimization. Section 3 describes in detail our proposed CMOE-SVM. Next, Section 4 outlines the experimental settings for our study, while Section 5 presents our experimental results and their corresponding statistical validation. Finally, Section 6 provides our general conclusions.

2 PRELIMINARIES

This section discusses the main preliminaries on which our contribution is based. Section 2.1 describes the main extensions proposed to solve multiclass problems using SVMs. Next, in Section 2.2, we describe the main characteristics of coevolutionary algorithms. Finally, Section 2.3 presents the basic concepts regarding multi-objective optimization.

2.1 Multiclass Extensions for SVMs

In this section, we revise the main extensions of SVMs for multiclass problems. These can be categorized into two major groups: Decomposition strategies and Single Machine Methods, which are briefly discussed next.

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

- ***One-vs-All*** (OVA) [3]: OVA decomposes the M-class problem into M subproblems and learns an SVM for each subproblem. An instance is classified in the class with the highest activation value. One disadvantage of OVA is that it introduces an artificial imbalance during training. Therefore, the larger the number of classes, the greater the rate of imbalance.
- ***One-vs-One*** (OVO) [12]: OVO trains a binary SVM for each possible pair of classes. An instance is assigned to the class with a majority vote. Nonetheless, OVO can lead to a large number of binary SVMs, slowing down the prediction stage.
- ***Directed Acyclic Graph*** (DAG) [17]: The training phase is similar to OVO, but differing in the prediction stage. DAG follows a tree structure, starting from a root node and moving until it reaches a leaf node, which indicates the class label. Note that, however, the performance of DAG depends on the SVM in the root node.

A comparison between these three strategies is performed in [11], finding that their accuracy is quite similar, with no statistical difference.

2.1.2 Single Machine Methods. These methods modify the objective function to solve directly the multiclass problem. Remarkable single machine methods are the following:

- ***Multiclass SVM Weston & Watkins*** (MSVM-WW) [27]: This formulation adds slack variables to each class and the constraints consider that the activation value of each sample in each class is at least twice that of the others.
- ***Multiclass SVM Lee, Lin & Wahba*** (MSVM-LLW) [14]: It is similar to MSVM-WW, but it reduces the dimensionality of the problem by means of a sum-to-zero constraint.
- ***Multiclass SVM Crammer & Singer*** (MSVM-CS) [5]: This formulation removes the bias term. Moreover, the constraints only consider that the activation value of each sample in each class is greater than the largest activation from other classes.
- ***Multiclass SVM Square*** (MSVM^2) [10]: It adds a quadratic function to the slack variables and also adopts the squared hinge loss function.

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.

2.2 Coevolutionary Optimization

Coevolutionary algorithms are able to manage two or more populations (species) simultaneously [28]. They allow splitting the problem into different parts and assign a different population to each subproblem, such that each focuses its efforts on solving one specific part of the problem. Two different kinds of coevolutionary algorithms can be described:

- **Competitive coevolutionary algorithms** [22]. The individuals of each population compete against each other. 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** [18]. 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.

2.3 Multi-Objective Optimization

A general Multi-Objective Optimization Problem (MOP) can be stated as follows:

$$\begin{aligned} &\min \mathbf{f}(\alpha) = [f_1(\alpha), \ldots, f_l(\alpha)]^T \\ &\text{subject to } \alpha \in \mathcal{A} \end{aligned} \tag{1}$$

where $\alpha \in \mathbb{R}^n$ is a decision variables vector, $f_h(\alpha)$, $h = 1, \ldots, l$, are the l-objective functions, and $\mathcal{A}$ is the set of feasible solutions.

The objectives in a MOP are conflicting, and therefore a single best solution for all of them does not exist. In these cases, we use Pareto optimality. We say that a solution α^1 dominates a solution α^2 (denoted by $\alpha^1 \preceq \alpha^2$) if and only if α^1 is better than α^2 at least in one objective and is not worse in the rest, i.e.,

$$\forall h : f_h\left(\alpha^1\right) \leq f_h\left(\alpha^2\right) \wedge \exists h : f_h\left(\alpha^1\right) < f_h\left(\alpha^2\right) \tag{2}$$

A solution α^* is a Pareto optimal solution if another solution $\alpha' \in \mathcal{A}$ does not exist such that $\alpha' \preceq \alpha^*$. One should note that this definition does not produce a single solution, but a set of trade-off solutions; this set is called Pareto optimal set, and the image of this set in objective function space is referred to as the Pareto Front.

3 CMOE-SVM: COOPERATIVE MULTI-OBJECTIVE EVOLUTIONARY SUPPORT VECTOR MACHINES

The proposed CMOE-SVM aims at training a multiclass SVM in a single step, by finding the set of support vectors of each class. It follows a cooperative approach, having a subpopulation optimizing the support vectors for a specific class while considering the other subpopulations (classes) to solve the multiclass problem. Algorithm 1 describes CMOE-SVM.

Algorithm 1 CMOE-SVM

Require: $\mathcal{X}$, the set of samples,
$\mathcal{Y}$, the set of classes labels,
P, the population size,
CR, F, the differential evolution parameters,
E, maximum number of evaluations
ω, the set of spread weight vectors associated to each individual
Ensure: The set of support vectors
1: Generate randomly an initial population, $\mathcal{P}_y$ for each class $y \in \mathcal{Y}$
2: **for** each $y \in \mathcal{Y}$ **do**
3: Select randomly an individual from each class $y' \in \mathcal{Y} \setminus y$
4: Construct full solutions by combining the selected individuals of each class
5: Evaluate the full solutions using the fitness functions, f
6: Save the nondominated solutions in an external archive, $EP(y)$
7: **end for**
8: **while** a stopping criterion is not met **do**
9: **for** each $y \in \mathcal{Y}$ **do**
10: Select an individual randomly from the nondominated set, $EP(y') \; \forall y' \in \mathcal{Y} \setminus y$
11: **for** each individual in the current class **do**
12: Apply evolutionary operators to create an offspring, α'
13: Evaluate the offspring with the fitness functions, f
14: Replace the current individual, α, if $\max(\omega_i |f(\alpha')|) < \max(\omega_i |f(\alpha)|)$
15: Update the nondominated solutions in $EP(y)$
16: **end for**
17: **end for**
18: **end while**
19: Construct the final solution based on the individuals in the external archive

(1) The algorithm starts creating a population at random. In each population, the number of variables is directly related with the number of samples in the training set for the given class. This is done in step 1.
(2) Steps 3 to 5 assess the fitness of each individual in a subproblem by randomly selecting individuals from other subproblems to form a complete solution. This is part of the cooperative coevolution.
(3) The best nondominated solutions are kept in an external archive in step 6.
(4) From steps 8 to 18, the evolutionary process takes place:
   (a) Step 10 randomly selects an individual of the nondominated set from other classes and the evolutionary operators are applied to create an offspring, in step 12.
   (b) Step 13 computes the fitness value of the offspring solution by forming a complete solution with the nondominated individuals from other populations.
   (c) A child solution replaces the current individual if it improves the current solution. To determine this, objectives are normalized and the aggregated fitness function is computed with the weighted Tchebycheff distance shown in step 14.
   (d) In step 15, the nondominated solutions in the external archive are updated by adding the child solution if no solution in the archive dominates it. Solutions that are dominated by the child are also removed.
(5) The multiclass SVM is constructed by selecting from each subproblem a solution from the external archive (see section 3.3).

The remainder of this section details the proposed method.

3.1 Fitness Functions: Optimization Problem

In the proposed approach, the multiclass problem is split into multiple subproblems, which are solved simultaneously in a cooperative fashion. Each subproblem has the task to learn the support vectors of a specific class. Thus, the multi-objective optimization problem for the r^{th} class is formulated as follows:

$$\begin{aligned} \min \; \mathbf{f}(\mathbf{w}_r, \xi^r) &= \begin{bmatrix} \frac{1}{2}\|\mathbf{w}_r\|^2 \\ \sum_{i=1}^{n_r} \xi_i^r \end{bmatrix} \\ \text{subject to } \langle \mathbf{w}_r, \mathbf{x}_i \rangle &\geq 1 - \xi_i^r, \quad \xi_i^r \geq 0 \end{aligned} \tag{3}$$

where n_r is the number of samples for the r^{th} class.

The first objective function aims at maximizing the separation margin while the second one seeks for controlling the over-fitting during the training. For the dual formulation, constraints are incorporated in the objective functions by using the Lagrange multipliers:

$$\begin{aligned}\mathcal{L}_1\left(\mathbf{w},\xi^r,\alpha^r,\beta^r\right) &= \frac{1}{2}\left\|\mathbf{w}_r\right\|^2 - \sum_{i=1}^{n_r}\alpha_i^r\left[\langle\mathbf{w}_r,\mathbf{x}_i\rangle - 1\right] \\ &\quad - \sum_{i=1}^{n_r}\alpha_i^r\xi_i^r - \sum_{i=1}^{n_r}\beta_i^r\xi_i^r \\ \mathcal{L}_2\left(\mathbf{w},\xi^r,\alpha^r,\beta^r\right) &= \sum_{i=1}^{n_r}\xi_i^r - \sum_{i=1}^{n_r}\alpha_i^r\left[\langle\mathbf{w}_r,\mathbf{x}_i\rangle - 1\right] \\ &\quad - \sum_{i=1}^{n_r}\alpha_i^r\xi_i^r - \sum_{i=1}^{n_r}\beta_i^r\xi_i^r \\ \text{subject to } \alpha_i^r,\beta_i^r &\geq 0.\end{aligned} \tag{4}$$

Setting the partial derivatives to zero, replacing them in Equation (4) and simplifying, we get the following equivalent optimization problem:

$$\begin{aligned}\min\ f'\left(\alpha^r\right) &= \begin{bmatrix} \frac{1}{2}\sum_{i,j=1}^{n_r}\alpha_i^r\alpha_j^r\left\langle\mathbf{x}_i,\mathbf{x}_j\right\rangle \\ -\sum_{i=1}^{n_r}\alpha_i^r \end{bmatrix} \\ \text{subject to } \alpha_i^r &\geq 0\end{aligned} \tag{5}$$

Equation (5) learns the support vectors for a single class label. In order to benefit from the cooperative evolution of each class label, a third objective is considered, which takes into account the interaction with other subproblems (classes). The multi-objective optimization problem in cooperative evolution is stated as:

$$\begin{aligned}\min\ f'\left(\alpha^r\right) &= \begin{bmatrix} \frac{1}{2}\sum_{i,j=1}^{n_r}\alpha_i^r\alpha_j^r\left\langle\mathbf{x}_i,\mathbf{x}_j\right\rangle \\ -\sum_{i=1}^{n_r}\alpha_i^r \\ 2 + \frac{1}{n_r}\sum_{i=1}^{n_r}\left(\sum_{j=1}^{n_q}\alpha_i^q\left\langle\mathbf{x}_i,\mathbf{x}_j\right\rangle - \right. \\ \quad \left.\sum_{j=1}^{n_r}\alpha_i^r\left\langle\mathbf{x}_i,\mathbf{x}_j\right\rangle\right) \end{bmatrix} \\ \text{subject to } \alpha_i^r &\geq 0\end{aligned} \tag{6}$$

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

Next, we describe the representation of the solutions and the evolutionary operators used in CMOE-SVM.

3.2 Representation

Each population consists of P individuals. The populations of each subproblem share the same individual representation. All individuals are encoded as real-valued vectors. The number of variables of each population is equal to the number of instances available in the training set for each class. Thus, the number of variables is not increased in the optimization task, as usually happens with other methods. Moreover, all populations are evolved simultaneously and each of them deals with simpler problems.

The α vector is randomly initialized, by assigning to each variable 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 by using the differential evolution operator [19], which generates a new child solution as follows:

$$\overline{\alpha}_i^{r(s)} = \begin{cases} \alpha_i^{r(t)} + F \times \left(\alpha_i^{r(u)} - \alpha_i^{r(v)}\right) & \text{rnd} \leq CR, \\ \alpha_i^{r(s)} & \text{Otherwise} \end{cases} \tag{7}$$

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 r^{th} 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 s^{th} parent in the weighted Tchebycheff function, as it is shown in line 14 from Algorithm 1.

3.3 Building the Multiclass SVM

Once the coevolutionary process is over, the multiclass classifier is built by selecting from each population the member that gets the minimal distance to an ideal vector, $\mathbf{z}^r$. The ideal vector is defined as:

$$\mathbf{z}^r = \left[z_h : z_h = \min f_h'\left(\alpha^r\right)\right]\ \forall h \in \{1,2,3\} \tag{8}$$

The solutions with $\alpha_i^r > 0$ are the support vectors and represent the multiclass classifier. A new instance $\mathbf{x}_t$ is classified as follows:

$$y_t = \underset{r}{\text{argmax}} \sum_{i \in SV_r}\alpha_i^r\left\langle\mathbf{x}_i,\mathbf{x}_t\right\rangle \tag{9}$$

where SV_r represents the set of support vectors from the r^{th} class.

3.4 Learning Nonlinear SVMs

The optimization problem presented in Equation (6) learns a linear function from the training data. The *kernel trick* can be used in CMOE-SVM by replacing the inner product, $\langle\mathbf{x}_i,\mathbf{x}\rangle$, in Equations (6) and (9), with a Kernel function, $K\left(\mathbf{x}_i,\mathbf{x}\right)$. Some commonly used kernel functions are the following [23]:

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

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

4 EXPERIMENTAL SETUP

In this section, we describe the experimental settings adopted in our study. In Section 4.1, we present the datasets used in our experimental study. Section 4.2 describes the algorithms that are used to compare the performance of CMOE-SVM. Finally, Section 4.3 presents the performance measures and statistical tests adopted to assess the results produced by each algorithm.

4.1 Datasets

We used a set of 20 datasets available in the KEEL repository [1] to experimentally validate the performance of CMOE-SVM. Table 1 shows some characteristics of these datasets, such as the number of instances, the number of features, the imbalance rate (IR)[2] and the number of classes. These datasets were chosen based on those having three o more classes.

These datasets have been partitioned into 10 training/test subsets by using the k-fold cross validation technique. Furthermore,

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

Table 1: 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
1	Automobile	25	203	5.69	6
2	Cleveland	13	303	3.87	5
3	Contraceptive	9	1,473	1.55	3
4	Dermatology	34	366	2.17	6
5	Ecoli	7	336	15.27	8
6	Glass	9	214	3.60	6
7	Iris	4	150	1.00	3
8	Led7digit	7	500	1.16	10
9	Lymphography	18	148	18.30	4
10	Movement Libras	90	360	1.00	15
11	Newthyroid	5	215	3.48	3
12	Penbased	16	10,992	1.05	10
13	Satimage	36	6,435	1.73	6
14	Segment	19	2,310	1.00	7
15	Splice	60	3,190	1.77	3
16	Texture	40	5,500	1.00	11
17	Vowel	13	990	1.00	11
18	Wine	13	178	1.30	3
19	Yeast	8	1,484	11.65	10
20	Zoo	16	101	3.20	7

features have been pre-processed in order to have zero mean and unit standard deviation.

4.2 Considered Algorithms

We compared the performance of CMOE-SVM against decomposition strategies and single machine methods. To this end, the following reference methods were considered:

- Decomposition strategies using the sequential minimal optimization (SMO) [16] implemented in KEEL:
  - OVO
  - OVA
- Single machine methods available at MSVMpack [13]:
  - MSVM^2
  - MSVM-CS
  - MSVM-LLW
  - MSVM-WW

All methods use the radial basis function (RBF) kernel. In order to allow a fair comparison, the values of the hyper-parameters for each method are tuned for each dataset, as it is suggested in [26]. Thus, we randomly sample 100 configurations following the methodology described in [2]. We have specifically considered the ranges of $CR = (0, 1]$, $F = (0, 2]$. The population size in all cases is set to 20, to avoid increasing the computational cost in CMOE-SVM. Moreover, the range $\gamma = \left[2^{-10}, 2^{2}\right]$ for the kernel's parameter is set for all methods. Reference methods further require to specify the regularization parameter, which is tuned in the range $C = \left[2^{-4}, 2^{10}\right]$. Each of the 100 configuration is tested and the one with the best score in accuracy is chosen for each dataset.

We have fixed to 200,000 the maximum number of evaluations of the objective functions for all methods. The convergence criterion of CMOE-SVM is defined as having an improvement in the distance from the nondominated front to the reference vector after 25 iterations lower than 0.001.

CMOE-SVM is also compared with respect to common learning algorithms. The set of learning algorithms are Random Forest (RF), K-Nearest Neighbor (KNN), and Naïve Bayes (NB). The hyper-parameters of these methods are tuned. In RF, the number of trees is adjusted in the range $[1, 100]$ while in KNN the neighborhood size is adjusted in the range $[1, 10]$.

4.3 Performance Measures

Accuracy and Cohen's kappa metrics are used to assess the performance of all methods. These metrics are computed as follows:

- **Accuracy** (Acc) indicates the ratio of samples that are correctly classified, i.e.,

$$Acc = \frac{1}{N} \sum_{i=1}^{M} TP_i \tag{10}$$

  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. Cohen's kappa is computed as follows:

$$\mathcal{K} = \frac{N \sum_{i=1}^{M} TP_i - \sum_{i=1}^{M} P_i T_i}{N^2 - \sum_{i=1}^{M} P_i T_i} \tag{11}$$

  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.

Non-parametric statistical tests were used to contrast the difference among methods, since this is widely recommended for comparing multiple classifiers over multiple datasets by [6, 8, 9]. We used the Friedman Aligned Ranks test to compare among CMOE-SVM and reference methods, and Holm's procedure was used to find out which algorithms are distinctive. In all cases, the significance level was set to $\alpha = 0.05$.

5 EXPERIMENTAL RESULTS

This section presents the results of our experimental study. First, in Section 5.1, we compare with respect to decomposition strategies. Next, Section 5.2 contrasts the proposed method against single machine methods. Section 5.3 shows a comparison between CMOE-SVM and standard learning algorithms. Finally, Section 5.4 compares the running time of each method.

5.1 Comparing with Decomposition Strategies

The aim of this experimental study is to assess the performance of CMOE-SVM and the decomposition strategies: OVA and OVO. Table 2 shows the average results obtained from the 20 datasets. The detailed results on each dataset and the hyper-parameters values of each method are provided as supplementary material. To allow

Table 2: Comparison between CMOE-SVM and the decomposition strategies SVM.

Method	*Acc*	$\mathcal{K}$
CMOE-SVM	**0.8363 ± 0.1577**	**0.7636 ± 0.2397**
OVO	0.8239 ± 0.1636	0.7258 ± 0.2876
OVA	0.8174 ± 0.1647	0.7334 ± 0.2577

Table 3: 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}
CMOE-SVM	21.28	—	21.35	—
OVO	31.15	0.0738	31.60	0.0635
OVA	39.08	0.0025	38.55	0.0037

reproducibility, the supplementary material and source code are available at ccc.inaoep.mx/~arosales/resources/CMOESVM.tar.gz.

Table 3 shows the ranking obtained by Friedman's Aligned Ranks both with accuracy score (*Acc*) and Cohen's Kappa ($\mathcal{K}$). It also shows the adjusted p-value with the Holm's test (p_{Holm}).

Based on Tables 2 and 3, we stress the following:

- CMOE-SVM statistically outperforms the OVA decomposition strategy. The statistical tests have revealed significant differences in both performance measures, under the considered level of $\alpha = 0.05$.
- OVO decomposition is clearly the most competitive decomposition strategy for the proposed CMOE-SVM in both accuracy and Cohen's Kappa metrics. This is noted with the lack of a statistically significant difference between both approaches.
- The most significant difference between CMOE-SVM with OVO and OVA is in Cohen's Kappa. This is interesting because the hyper-parameters for each method were set by considering accuracy as the main criterion. Thus, CMOE-SVM is not overfitted to this criterion.

5.2 Comparing with Single Machine Methods

In this section, our goal is to contrast the performance of CMOE-SVM with respect to single machine methods. Table 4 shows the average results obtained by each method and Table 5 shows the ranking for each algorithm computed with Friedman's Aligned Ranks method and the adjusted p-values with the Holm's test. Based on these, the following is highlighted:

- The difference between CMOE-SVM and single machine methods is statistically significant in most cases.
- MSVM-CS excels as the most prominent single machine method. This is consistent with the fact that there is no significant difference between CMOE-SVM and MSVM-CS.
- Both CMOE-SVM and MSVM-CS have the major improvements in the Cohen's Kappa. Moreover, they do not consider the bias term in the optimization problem.

Table 4: Comparison between CMOE-SVM and single machine methods.

Method	*Acc*	$\mathcal{K}$
CMOE-SVM	**0.8363 ± 0.1577**	**0.7636 ± 0.2397**
MSVM-CS	0.8307 ± 0.1687	0.7591 ± 0.2481
MSVM-WW	0.8082 ± 0.1699	0.7178 ± 0.2628
$MSVM^2$	0.7897 ± 0.1740	0.6876 ± 0.2721
MSVM-LLW	0.8067 ± 0.1750	0.6993 ± 0.3033

Table 5: 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}
CMOE-SVM	32.88	—	33.35	—
MSVM-CS	40.10	0.4310	36.80	0.7069
MSVM-WW	59.08	0.0111	57.75	0.0156
$MSVM^2$	59.50	0.0111	61.40	0.0067
MSVM-LLW	60.95	0.0088	63.20	0.0046

Table 6: Comparison between CMOE-SVM and standard learning algorithms.

Method	*Acc*	$\mathcal{K}$
CMOE-SVM	**0.8363 ± 0.1577**	0.7636 ± 0.2397
RF	0.8362 ± 0.1558	**0.7638 ± 0.2339**
KNN	0.8102 ± 0.1647	0.7317 ± 0.2508
NB	0.7611 ± 0.1714	0.6842 ± 0.2380

Table 7: 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}
CMOE-SVM	29.00	—	29.95	—
RF	31.75	0.7672	32.58	0.7209
KNN	45.85	0.0437	46.15	0.0550
NB	55.98	0.0002	53.33	0.0044

5.3 Comparing with Standard Learning Algorithms

In this section, we aim at assessing the performance of the proposed method against well-known learning algorithms. Table 6 shows the average results over the 20 datasets and Table 7 reports the average rankings and adjusted p-values.

From these tables, we remark:

- CMOE-SVM and RF have virtually the same performance. This is an interesting result since RF is a powerful learning algorithm based on an ensemble of several decision trees.

Thus, in spite of the fact that CMOE-SVM does not consider building ensembles, it still has a highly competitive performance.
- There is no statistically significant difference between CMOE-SVM and KNN when Cohen's Kappa is considered. However, in accuracy, tests have shown a significant difference.
- NB is the worst one, both in accuracy and Cohen's Kappa metric. Statistical tests have revealed significant differences in both scores.

5.4 Comparing Running Times

Figure 1 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 approaches, and have virtually the same performance.
- CMOE-SVM is the third best method, requiring at most, 118 seconds for solving each benchmark problem.
- Single machine methods are clearly the slowest ones. This is due to the fact that they deal with a larger optimization problem.
- In the best case, single machine methods required around 280 seconds to ensure solving each dataset.
- In contrast to its outstanding performances, MSVM-CS is the worst one in terms of computational time.

6 CONCLUSIONS

This paper introduced CMOE-SVM, an approach for handling multiclass problems with SVMs via multi-objective coevolutionary optimization. The method resembles the decomposition strategies and the single-machine methods, by decomposing the problem into several subproblems and solving them via cooperative coevolution. However, unlike decomposition strategies, CMOE-SVM is able to capture in a single model the multiclass classification problem, and in contrast with single machine methods, it does not increase the number of variables to be optimized. These features make the proposed extension a more flexible approach.

The use of evolutionary algorithms has allowed us to formulate the optimization problem as a multi-objective one, by explicitly and simultaneously optimizing the margin and controlling the overfitting during the training. An inherent advantage of this is that our formulation does not require to specify the penalty parameter beforehand, but this trade-off is managed as part of the training stage.

The experimental evaluation was carried out using 20 benchmark datasets. The experimental results give evidence of the suitability of CMOE-SVM to handle multiclass problems, by outperforming most of the multiclass SVM extensions. This claim is supported by a set of non-parametric tests with a level of significance of $\alpha = 0.05$. MSVM-CS has shown to be the most competitive extension in terms of the prediction performance, but it is also the slowest one during the training process. CMOE-SVM is able to handle a better trade-off between performance and training time than MSVM-CS.

Finally, non-positive semidefinite kernels lead to a non-convex optimization problem, narrowing their applicability to the quadratic solver of SVMs. As part of our future work, we want to exploit the global search properties of EAs, so as to explore the use of non-positive semidefinite kernels with SVMs.

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 project 221551.

REFERENCES

[1] Jesús Alcalá-Fdez, Alberto Fernández, Julián Luengo, Joaquín Derrac, Salvador García, Luciano Sánchez, and Francisco Herrera. 2011. KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *J. Mult.-Valued Log. S.* 17, 2-3 (Jun. 2011), 255–287.

[2] James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-parameter Optimization. *J. Mach. Learn. Res.* 13 (Feb. 2012), 281–305.

[3] Léon Bottou, Corinna Cortes, John S. Denker, Harris Drucker, Isabelle Guyon, L. D. Jackel, Yam LeCun, Urs A. Muller, Eduard Sackinger, Patrice Simard, and Vladimir Vapnik. 1994. Comparison of classifier methods: A case study in handwritten digit recognition. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, Vol. 2. 77–82. https://doi.org/10.1109/ICPR.1994.576879

[4] Clément Chatelain, Sébastien Adam, Yves Lecourtier, Laurent Heutte, and Thierry Paquet. 2010. A multi-model selection framework for unknown and/or evolutive misclassification cost problems. *Pattern Recogn.* 43, 3 (Mar. 2010), 815 – 823. https://doi.org/10.1016/j.patcog.2009.07.006

[5] Koby Crammer and Yoram Singer. 2002. On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. *J. Mach. Learn. Res.* 2 (Mar. 2002), 265–292.

[6] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* 7 (Jan. 2006), 1–30.

[7] Madson L. Dantas Dias and Ajalmar R. Rocha Neto. 2016. Evolutionary support vector machines: A dual approach. In *2016 IEEE Congress on Evolutionary Computation*. 2185–2192. https://doi.org/10.1109/CEC.2016.7744058

[8] Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. 2010. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Inform. Sciences* 180, 10 (May. 2010), 2044–2064.

[9] Salvador García and Francisco Herrera. 2008. An Extension on "Statistical Comparisons of Classifiers over Multiple Data Sets" for all Pairwise Comparisons. *J. Mach. Learn. Res.* 9 (Dec. 2008), 2677–2694.

[10] Yann Guermeur and Emmanuel Monfrini. 2011. A Quadratic Loss Multi-Class SVM for Which a Radius-Margin Bound Applies. *Informatica* 22, 1 (Jan. 2011), 73–96.

[11] Chih-Wei Hsu and Chih-Jen Lin. 2002. A comparison of methods for multiclass support vector machines. *IEEE T. Neural Network.* 13, 2 (Mar. 2002), 415–425. https://doi.org/10.1109/72.991427

[12] Ulrich H. G. Kreßel. 1999. Pairwise Classification and Support Vector Machines. In *Advances in Kernel Methods*, Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola (Eds.). MIT Press, Cambridge, MA, USA, 255–268.

[13] Fabien Lauer and Yann Guermeur. 2011. MSVMpack: A Multi-Class Support Vector Machine Package. *J. Mach. Learn. Res.* 12 (Jul. 2011), 2293–2296.

[14] Yoonkyung Lee, Yi Lin, and Grace Wahba. 2004. Multicategory Support Vector Machines, theory, and application to the classification of microarray data and satellite radiance data. *J. Am. Stat. Assoc.* 99 (Jun. 2004), 67–81. https://doi.org/10.1198/016214504000000098

[15] Ingo Mierswa. 2006. Evolutionary Learning with Kernels: A Generic Solution for Large Margin Problems. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation (GECCO '06)*. 1553–1560. https://doi.org/10.1145/1143997.1144249

[16] John C. Platt. 1999. Fast Training of Support Vector Machines Using Sequential Minimal Optimization. In *Advances in Kernel Methods*, Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola (Eds.). MIT Press, Cambridge, MA, USA, 185–208.

[17] John C. Platt, Nello Cristianini, and John Shawe-Taylor. 2000. 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, 547–553.

[18] Mitchell A. Potter and Kenneth A. De Jong. 2000. Cooperative Coevolution: An Architecture for Evolving Coadapted Subcomponents. *Evol. Comput.* 8, 1 (Mar. 2000), 1–29. https://doi.org/10.1162/106365600568086

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

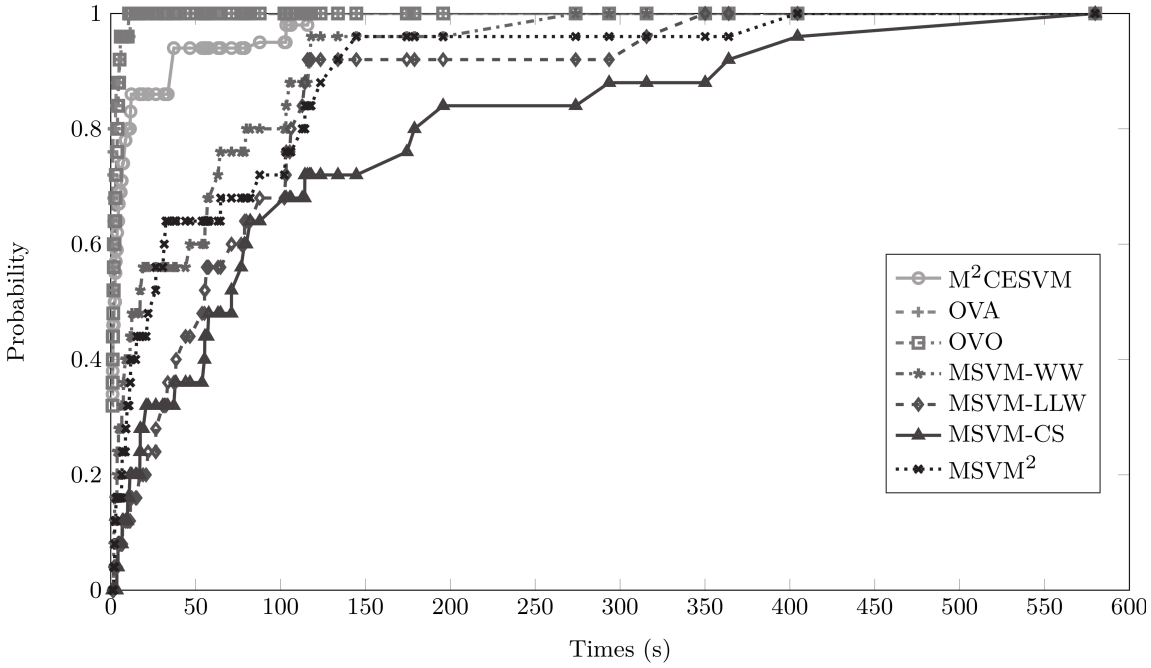

Figure 1: Probability of each method to learn a Multiclass SVM classifier in a given amount of time.

[20] Alejandro Rosales-Pérez, Salvador García, Jesus A. Gonzalez, Carlos A. Coello Coello, and Francisco Herrera. 2017. An Evolutionary Multiobjective Model and Instance Selection for Support Vector Machines With Pareto-Based Ensembles. *IEEE T. Evol. Comput.* 21, 6 (Dec. 2017), 863–877. https://doi.org/10.1109/TEVC.2017.2688863

[21] Alejandro Rosales-Pérez, Hugo Terashima-Marin, Salvador García, Francisco Herrera, and Carlos A. Coello Coello. 2018. MC^{2}ESVM: Multiclass Classification Based on Cooperative Evolution of Support Vector Machines. *IEEE Comput. Intell. M.* 13, 2 (Apr. 2018), 18–29. https://doi.org/10.1109/MCI.2018.2806997

[22] Christopher D. Rosin and Richard K. Belew. 1997. New Methods for Competitive Coevolution. *Evol. Comput.* 5, 1 (Mar. 1997), 1–29. https://doi.org/10.1162/evco.1997.5.1.1

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

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

[25] Nele Verbiest, JoaquÃŋn Derrac, Chris Cornelis, Salvador García, and Francisco Herrera. 2016. Evolutionary wrapper approaches for training set selection as preprocessing mechanism for support vector machines: Experimental evaluation and support vector analysis. *Appl. Soft. Comput.* 38 (Jan. 2016), 10 – 22. https://doi.org/10.1016/j.asoc.2015.09.006

[26] Jacques Wainer and Gavin Cawley. 2017. Empirical Evaluation of Resampling Procedures for Optimising SVM Hyperparameters. *J. Mach. Learn. Res.* 18, 15 (Jan. 2017), 1–35.

[27] Jason Weston and Christopher Watkins. 1999. Support vector machines for multi-class pattern recognition. In *7th European Symposium on Artificial Neural Networks.* 219–224.

[28] Rudolf Paul Wiegand. 2004. *An Analysis of Cooperative Coevolutionary Algorithms.* Ph.D. Dissertation. George Mason University, Fairfax, VA, USA.