Surrogate-Assisted Multi-Objective Model Selection for Support Vector Machines

Alejandro Rosales-Pérez[a,*], Jesus A. Gonzalez[a], Carlos A. Coello Coello[b], Hugo Jair Escalante[a], Carlos A. Reyes-Garcia[a]

[a]*Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), Computer Science Department, Luis Enrique Erro No.1, Santa María Tonantzintla, Puebla, 72840, Mexico*
[b]*Centro de Investigación y de Estudios Avanzados del IPN (CINVESTAV-IPN), Computer Science Department, Evolutionary Computation Group (EVOCINV), Av. IPN No. 2508, San Pedro Zacatenco, Mexico City, 07360, Mexico*

Abstract

Classification is one of the most well-known tasks in supervised learning. A vast number of algorithms for pattern classification have been proposed so far. Among these, support vector machines (SVMs) are one of the most popular approaches, due to the high performance reached by these methods in a wide number of pattern recognition applications. Nevertheless, the effectiveness of SVMs highly depends on their hyper-parameters. Besides the fine-tuning of their hyper-parameters, the way in which the features are scaled as well as the presence of non-relevant features could affect their generalization performance. This paper introduces an approach for addressing model selection for support vector machines used in classification tasks. In our formulation, a model can be composed by feature selection and pre-processing methods besides the SVM classifier. We formulate the model selection problem as a multi-objective one, aiming to minimize simultaneously two components that are closely related to the error of a model: bias and variance components, which are estimated in an experimental fashion. A surrogate-assisted evolutionary multi-objective optimization approach is adopted to explore the hyper-parameters space. We adopted this approach due to the fact that estimating the bias and variance could be computationally expensive. Therefore, by using surrogate-assisted optimization, we expect to reduce of the number

*Corresponding author. Tel.: +52 (222) 2663100 x 3413
Email address: arosales@inaoep.mx (Alejandro Rosales-Pérez)

of solutions evaluated by the fitness functions so that the computational cost would also be reduced. Experimental results conducted on benchmark data sets widely used in the literature, indicate that highly competitive models with a fewer number of fitness function evaluations are obtained by our proposal when it is compared to state of the art model selection methods.

Keywords: Model selection, Multi-objective optimization, Support vector machines, Surrogate-assisted optimization.

---

1. Introduction

Supervised classification is the task of learning a function from a labeled data set. This function is a model that is used to predict the response of future data points from the same problem by mapping a data point from the feature space to a class label. To date, there are many machine learning algorithms that can be used for constructing such model. Among these, support vector machines (SVMs) [5, 12, 40] is one of the most powerful. The popularity of SMVs relies on their theoretical background, high performance, and scalability. In spite of this, the effectiveness of SVMs depends on the fine-tuning of a set of parameters (usually called hyper-parameters), such as the kernel type and its parameters. Furthermore, other factors that can affect their performance are the way features are scaled, or the presence of irrelevant/redundant features in a data set. Therefore, it can be beneficial for the SVMs if the data are first pre-processed. This raises the issue of model selection, which is a crucial step to obtain classifiers with a good performance.

The problem of choosing the hyper-parameters values for an SVM can be seen as an optimization one, where a search engine is used to explore the corresponding hyper-parameters space. A number of methods for tackling it have been proposed for this sake, so far. Most of these methods address this problem by fixing a priori a kernel type and they perform the selection of the hyper-parameters values for that kernel. These methods could be mainly differentiated in two aspects: by the criterion used and by the search strategy adopted for this purpose. Regarding the first aspect, the studies can be differentiated in those that consider a single-criterion and those that consider multiple criteria for guiding the search. Single-criterion approaches [1, 3, 8, 9, 26] usually adopt the well-known k-fold cross validation to estimate the performance of a given configuration of hyper-parameters. Multiple criteria approaches typically consider an estimation of the model performance and a

measure of the model complexity (such as the number of support vectors) [2, 37]. Others have considered the number of features and an estimation of the generalization error [22], estimates of the bias and variance of the model [33], or the minimization of the errors in positive and negative classes [10, 25].

On the other hand, regarding the second aspect, the most commonly adopted techniques are grid search [8, 38, 39], gradient-based methods [1, 7, 33, 9], and meta-heuristics such as evolutionary algorithms [10, 22, 25, 26, 37], artificial immune systems [2], or particle swarm optimization [3].

Grid search is the simplest method for adjusting the values of the hyper-parameters. This strategy requires to discretize the search space, which is attained by the variation of each hyper-parameter with a step size through a wide range of values and the performance of each combination is typically assessed through a k-fold cross-validation technique. Such cross-validation makes grid search a computationally expensive method which is suitable only when few hyper-parameters need to be set. The way in which the search space is discretized is another crucial issue in grid search.

Gradient-based methods are highly efficient and have been successfully applied to hyper-parameter optimization for SVMs. In spite of this, they still have some drawbacks. For instance, the objective function has to be differentiable with respect to the hyper-parameters and the kernel, which also needs to be differentiable. Moreover, the effectiveness of these methods highly depends on the initial point chosen for the search, which causes that they can be susceptible to getting trapped in a local optimal solution due to the multimodality of the problem.

Several studies have adopted evolutionary algorithms to alleviate the above mentioned shortcomings, since they are more robust to local optimal solutions than gradient-based methods. Although these methods can be computationally cheaper than grid search methods, they can still be computationally expensive.

An alternative approach consists of tackling the model selection problem as a supervised learning task through meta-learning [36]. In meta-learning, a number of problems (datasets) are described by a set of features (meta-features) in conjunction with the information about the performance obtained from a set of candidate models; these constitute a meta-dataset. A meta-learner is constructed from the meta-dataset. Given a new problem, the meta-learner is used to predict a model based on its meta-features. Even when meta-learning approaches are more efficient than those based on search techniques, they have some drawbacks. The most important one is that

meta-learning depends on the quality of the meta-samples, as well as on the number of problems used for generating a meta-dataset, which could be limited. Recent studies that combine meta-learning with a search strategy have been proposed [18, 28, 29, 31]. The main idea behind these methods is to use meta-learning for obtaining an initial suggestion of potential models, which is then used to provide initial search points in the optimization step. Nonetheless, convergence in the optimization stage could be affected if the suggestions given by meta-learning are not good enough.

In spite of the considerable number of studies currently available on SVMs model selection, to the authors' best knowledge, little effort has been devoted to considering the selection of both the pre-processing method and the feature selection method in combination with defining the parameters of the SVM. In this paper, we describe a novel approach for SVM model selection through the use of multi-objective optimization. In this case, the preprocessing stage, feature selection, and the hyper-parameters tuning for a SVM are all taken into consideration in the model selection formulation. Estimates of bias and variance of a model are defined as the objectives in our multi-objective formulation. Inspired on the ideas of meta-learning, we address the optimization stage through a surrogate-assisted multi-objective optimization approach. Unlike meta-learning approaches, in which a meta-learner is constructed to obtain an initial suggestion of models, under this formulation, a surrogate is constructed aiming to approximate the objective functions. The main contribution of this paper is a novel method for performing a SVM model selection (i.e., besides hyper-parameters selection for SVMs, we aim to choose both pre-processing and feature selection methods). We assessed the performance of our proposal with a suite of benchmark data sets, widely used in the specialized literature. Our experimental results show that our proposal obtains highly competitive models in terms of generalization performance with a lower number of fitness function evaluations.

The remainder of this paper is organized as follows. In Section 2, we present the bias and variance definitions proposed for classification tasks. Section 3 describes our proposal for tackling the model selection problem for SVMs in classification problems. Next, Section 4 shows the experimental settings and experimental results that show the viability of our proposal. Finally, the main conclusions and some possible paths for future work are presented in Section 5.

2. Bias and Variance Decomposition in Classification Problems

From a statistical point of view, the expected error over a sample $x \in \mathbb{R}^n$ can be decomposed into two components: the squared bias and the variance. The bias-variance decomposition was borrowed from the field of regression, using squared-loss loss function. Based on that definition, several bias-variance decompositions have been proposed in the field of classification using the 0-1 loss function, which is commonly adopted in classification tasks. Roughly speaking, square bias is a measure of the contribution to the error of the central tendency (i.e., the class with the most votes across the multiple predictions) when a model is trained with different data sets. The variance is a measure of the deviations of the central tendency when a model is trained with different data sets [41].

In order to obtain a better generalization error, both components should be minimized. Nevertheless, reducing one of them causes an increment in the other one. This is known as the bias-variance dilemma [4, 17, 20]. It is said that a model with low bias is too flexible and has a low training error rate, but its generalization capability is poor; this is known as the *overfitting* problem. In contrast, a model with low variance is too simple, has low complexity and does not have the ability to learn the training set and its generalization performance is also poor; this is known as the *underfitting* problem. Therefore, a good model is the one which provides a good trade-off between these two components. So, here we face the model selection task as a multi-objective optimization problem. We used as objectives the estimates of bias and variance, with the aim of selecting the model with the best trade-off between both components.

In classification tasks, different ways to estimate the bias and the variance have been proposed [15, 17, 21, 23, 24, 41]. The definition proposed by Kong and Dietterich [24] measures the bias directly from the error with respect to the central tendency[1] and the variance is defined as the difference between the error and the bias. Nonetheless, this definition applies to the noise-free cases, and it could lead to negative values for the variance. Kohavi and Wolpert define [23] the bias and variance as a quadratic function of the difference between the probabilities that a sample belongs to a class and that the model is able to predict such class. The advantage is that this definition is applicable to multi-class cases. Nevertheless, it does not measure the extent

[1]The central tendency is the class with the most votes.

to which each of these contributes to the error. Friedman's definition [17] is the closest to that defined for regression tasks, but it applies only for binary classification problems. Domingos [15] proposed a unified definition of the bias and variance decomposition for an arbitrary loss function. A similar approach was proposed by James [21], who extended the notion of bias and variance for general loss functions. Finally, in the definition proposed by Webb [41], the bias measures the extent of the contribution to the central tendency of the error, and the variance measures the deviation with respect to the central tendency. The different definitions have some benefits and limitations. In spite of this, all of them have proven to give insights of the bias and variance contribution to the model error, each one at least in the context that were introduced. In our study, we adopted Webb's definition [41] to compute the bias and variance for a given model with respect to the pre-processing, feature selection methods, kernel and its hyper-parameters.

Bias and variance definitions make use of knowledge of the distribution of the target function, which is usually unknown. Typically, the only knowledge that one has about this distribution is a data set of training samples. Thus, in practical situations, the true values of bias and variance terms have to be exchanged for estimates. One way of doing so is by splitting the data set into disjoint parts: training and test parts. This can be attained by using k-fold cross validation, bootstrapping, or another sampling strategy. In consequence, N models are trained with different partitions of a data set. Then, the N models are tested using the samples that were not used during the training phase. The predictions are recorded and finally an estimation of bias and variance is computed based on a previous given definition.

3. Multi-Objective Support Vector Machines Model Selection

The final goal of our proposed method is to select a classification model that has a good generalization performance. To tackle this task, we formulate the model selection problem as a multi-objective optimization one. We used a surrogate-assisted multi-objective evolutionary algorithm for exploring the hyper-parameters space. The method takes into consideration two fitness functions for optimization. The first is the bias and the second one is the variance. In the proposed method, for computing the bias and variance, we used the well-known $n \times k$-fold cross validation. Hence, the aim of adopting a surrogate-assisted optimization approach is to approximate the fitness functions through surrogates and to reduce the computational cost in the

optimization step. Figure 1 shows the general architecture of our proposal. Our method starts by dividing the training set into two disjoint sets, called learning set and validation set. The learning set is used to fit the parameters of the model during the search phase. At the end of the search, a set of solutions that satisfies the best trade-off between the bias and variance is obtained. The next step is to choose one solution from that set. In order to avoid the selection of an underfitted or overfitted (see Figure 2) solution, the validation set is used to test each model in the Pareto optimal set. We select the solution with the lowest error rate in the validation set. Finally, the model is trained using both, the learning set and the validation set. The selected model is tested over a new data set to evaluate its performance.

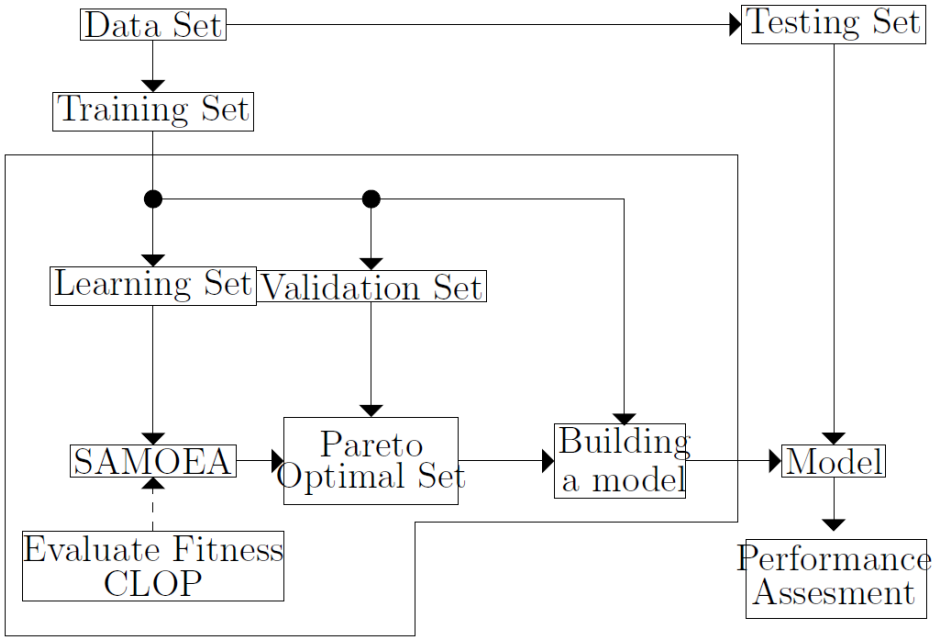

Figure 1: General architecture of the proposed model selection approach.

For the machine learning algorithms, we used the Challenge Learning Object Packet (CLOP) [34]. This Matlab toolbox contains several methods for pre-processing and feature selection, as well as several machine learning algorithms. Table 1 shows some methods available in in CLOP, which were used in our study.

In the rest of this section, we explain the proposed multi-objective formulation for approaching the SVM model selection. First, we provide a short

introduction to evolutionary multi-objective optimization and explain the adopted approach. Next, we describe the proposed technique.

3.1. Evolutionary Multi-Objective Optimization

A general multi-objective optimization problem (MOOP) can be stated as follows [11, 13, 27]:

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

where $\mathbf{x} = [x_1, \ldots, x_n]^T \in \mathbb{R}^n$ is a decision variables vector, $f_i(\mathbf{x})$, $i = 1, \ldots, l$, are the l-objective functions, and $\mathcal{X}$ is the set of feasible solutions.

In the case that the objectives in a MOOP are conflicting, usually there is not a single solution that would be the best for all these. Consequently, the notion of optimum in MOOPs differs from that in single-objective optimization. In MOOPs, the focus is on finding solutions that satisfy a good trade-off among the objectives. Pareto optimality provides a framework to determine such trade-off, which allows to deal with such cases. We say that solution $\mathbf{x}^1$ dominates a solution $\mathbf{x}^2$ ($\mathbf{x}^1 \preceq \mathbf{x}^2$) if and only if $\mathbf{x}^1$ is better than $\mathbf{x}^2$ at least in one objective and it is not worse in the rest, i.e.,

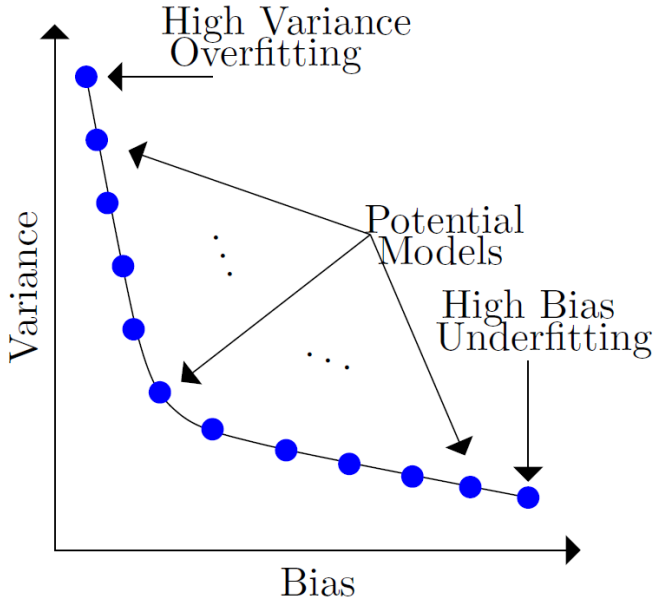

Figure 2: Models with high bias or variance in the resulting set of solutions.

Table 1: Available methods in the CLOP toolbox

Object	No. Pars.	Description
		Pre-Processing
standardize	1	Standardization of feat.
mormalize	1	Normalization of samples.
Shift-scale	2	Data are shifted and scaled.
pcextract	1	Principal Components A.
		Feature Selection
S2n	2	Signal to noise ratio.
relief	3	Relief feature ranking.
ZFilter	2	Statistical filter.
AUC fs	2	AUC criterion.
		Learning Algorithm
SVM	4	A Support Vector Machine Classifier.

$$\forall i : f_i\left(\mathbf{x}^1\right) \leq f_i\left(\mathbf{x}^2\right) \wedge \exists i : f_i\left(\mathbf{x}^1\right) < f_i\left(\mathbf{x}^2\right) \tag{2}$$

A solution $\mathbf{x}^*$ is Pareto optimal if there is not another solution $\mathbf{x}' \in \mathcal{X}$ such that $\mathbf{x}' \preceq \mathbf{x}^*$. The set of all Pareto optimal solutions is called the *Pareto optimal set*, and the image of this set in the objective space is referred to as the *Pareto Front*.

Evolutionary algorithms are stochastic search techniques inspired in Darwin's evolutionary theory. These algorithms have been successfully used for solving MOOPs, and offer several advantages with respect to mathematical programming techniques. For example, multi-objective evolutionary algorithms (MOEAs) can generate several elements of the Pareto optimal set in a single run (instead of generating only one solution at a time, as normally happens with mathematical programming techniques) and are less susceptible than mathematical programming techniques to the shape and continuity of the Pareto front [11, 13].

Since the seminal work of Schaffer [35] in the mid-1980s, an important number of MOEAs have been proposed. For instance, SPEA2 [44], NSGA-II [14], and MOEA/D [43] are some of the MOEAs currently available in the specialized literature. A comprehensive review of MOEAs can be found in [11, 13].

MOEAs have been widely used to approximate the Pareto front. Never-

theless, most of them require performing a relatively high number of fitness function evaluations to generate a reasonably good approximation. In the problem that we face, this could be a shortcoming insomuch as the computation of the bias and variance is done by performing $n \times k$-fold cross validation, which implies that a model has to be trained and tested several times to estimate such terms. To overcome this handicap, in this paper, we adopted a surrogate-assisted optimization approach. In this context, a surrogate is a cheaper approximation of the fitness function. To this extent, we expect to reduce the number of required (fitness functions) evaluations without degrading, in a significant manner, the quality of the obtained solutions. The Surrogate-Assisted Multi-Objective Evolutionary Algorithm (SAMOEA) [32] is taken up to this aim. SAMOEA is described in Algorithm 1. This algorithm keeps two external populations: the first one stores the non-dominated solutions found so far during the search, whilst the second one stores the solutions that were evaluated with the fitness function in order to be used for constructing the surrogates. SAMOEA starts by creating an initial population using a Latin hyper-cubes technique, so that the initial points are uniformly distributed on the domain search (step 1 in Algorithm 1). The initial points are evaluated with the fitness functions and they are stored in the second external archive, whilst the non-dominated solutions are stored in the first external archive (steps 2-4). After that, the solutions in the second external archive are used to construct the surrogates, which are an approximation to the fitness functions (step 5). Next, an offspring population is created by applying evolutionary operators over a parent population (step 7). The offspring population is evaluated with the surrogates, and the non-dominated solutions are selected to be evaluated with the fitness functions (steps 8-10). Both the external archives and the surrogates are updated based on the knowledge of the landscape obtained (steps 11-12). Finally, steps 7 to 12 are repeated until a stopping criterion is not reached. A detailed description of SAMOEA is out of scope of this paper, but interested readers are referred to [32].

We should highlight that each member in the evolutionary algorithm's population represents, in our case, a potential model for a given classification task. The surrogates are used to evaluate such potential models and those that are the most promising to be optimal are selected to be evaluated with the fitness functions. Thus, the required number of trained and tested models is reduced. In this manner, we aim to reduce the computational cost. Next, we explain the application of SAMOEA for addressing the model selection

Algorithm 1 SAMOEA

Require: **f**: objective functions vector,
N: population size,
G: maximum number of generations,
m: size of external archive B,
n: number of models in the ensemble.

Ensure: A set of non-dominated solutions

1: Create an initial population, P_0, using latin-hypercubes sampling
2: Evaluate initial population's members using the real fitness functions
3: Store the non-dominated solutions in external archive A
4: Store the population's members in external archive B
5: Construct surrogates from solution in external archive B.
6: **while** A stopping criterion is not satisfied **do**
7: Apply evolutionary operators (selection, crossover and mutation) over the parents and archive A populations to create an offspring population
8: Evaluate offspring's members using the surrogates
9: Determinate the non-dominated solutions from offspring population
10: Evaluate the non-dominated solutions using the real fitness functions
11: Update external archives.
12: Update surrogates from solution in external archive B.
13: **end while**

problem.

3.2. Surrogate for Addressing the Model Selection

Approaches that combine meta-learning with a search strategy typically address the problem by using meta-learning to obtain an initial suggestion of possible search points, which are the used in the search step. Therefore, meta-learning should give suggestions of potential models for the problem at hand. This is attained by constructing a model (called meta-learner), which predicts what the expected performance of a model would be, based on the features of a dataset. In consequence, the meta-learner receives as input a dataset described by a set of features and its output is the expected performance. In this paper, we address this issue in a different way by using surrogate-assisted optimization. As we previously stated, the surrogates are used to approximate the fitness values. However, these surrogates should be first constructed to make such approximation. To achieve this, instead of describing a dataset by a set of features (which could be an issue since making such a description depends on the set of features adopted), a surrogate receives as its input a potential model, which is described by its configuration, and the output is the expected value of a measure if such model is used with the problem at hand.

Summarizing, the differences between meta-learning plus search and our proposed approach are the following:

- In meta-learning plus search, meta-learning is performed first in order to predict potential models, which are then used in the search step as initial search points. During the search, the merit of each model is assessed by the objective function (i.e., the fitness function). In our proposed approach, the surrogate is used in conjunction with the search, such that, based on some criterion, the merit of some models is assessed by the fitness function and the merit of the others by the surrogate; i.e., the surrogate and the fitness function interact during the search.

- In meta-learning plus search, the learning of a meta-learned requires a meta-dataset. The construction of such meta-dataset requires to describe a number of datasets (preferably from different domains) by a set of meta-features. This makes that the quality of the initial suggestion depends on the meta-features and on a similar dataset that had

been previously learned. In our proposed approach, the meta-data set is constructed with the configurations of models and the response of such model for a given dataset (the problem at hand). This is attained by evaluating a set of models, which would be representative from the search space, with the dataset of interest. Besides, the interaction with the fitness function would allow to update the knowledge of the surrogate.

3.3. Representation

Evolutionary algorithms work with a population of solutions. In our problem, each solution represents a potential model to be used in the classification task. Therefore, under the adopted approach, we need to represent each potential model as an individual for being used by SAMOEA. Each model is encoded in a 12-dimensional vector as follows:

$$\mathbf{x}^i = \left[x^i_{mt}, x^i_{pp}, x^i_{fs}, x^i_{hp_1}, \ldots, x^i_{hp_9}\right] \tag{3}$$

where x^i_{mt} controls if a pre-processing and/or feature selection methods are applied and the order in which they are applied, x^i_{pp} is a pre-processing method, x^i_{fs} is a feature selection method, $[x^i_{hp_1}, \ldots, x^i_{hp_3}]$ represents the hyper-parameter for the feature selection method, $[x^i_{hp_4}, \ldots, x^i_{hp_5}]$ the hyper-parameters for the pre-processing method, and $[x^i_{hp_6}, \ldots, x^i_{hp_9}]$ the hyper-parameters for the SVM.

3.4. Evolutionary Operators

In general, evolutionary algorithms are population-based search techniques. The main idea behind these methods is to evolve a population of solutions. Evolutionary operators are the components that are in charge of performing the evolution of a current population. The two main evolutionary operators are crossover and mutation.

In the literature, a number of evolutionary operators have been proposed to deal with a particular type of encoding. One should note that in the encoding adopted in this work we have both discrete values and real-number values. Standard evolutionary operators are applied for each type of variables. For the discrete encoding, we used uniform crossover [42] and for the real-numbers encoding, we adopted binary simulated crossover (SBX) [13, 42]. For mutation, we generate a random value within the allowable range for the discrete encoding. For the real-numbers encoding, polynomial-based mutation [13, 42] is used.

Regarding the selection operator, which chooses the parents to be recombined, we adopted a binary tournament scheme [13, 42]. This operator randomly chooses two individuals from the population and the one with the best score is the winner of the tournament.

3.5. Fitness Function

The fitness function determines how good a solution is with respect to others. In our model, the selection task is treated as a multi-objective optimization problem, where the bias and variance are the objectives to be minimized. We used Webb's definition [41] of bias and variance, but other definitions can also be applied. Both objectives would correspond to $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ from equation (1), respectively. In our case, bias and variance are estimated in the following way:

$$\begin{aligned} f_1(\mathbf{x}) &= P_{(X,Y),D}\left(f_D(x) \neq f(x) \wedge f_D(x) = C_{f,D}(x)\right) \\ f_2(\mathbf{x}) &= P_{(X,Y),D}\left(f_D(x) \neq f(x) \wedge f_D(x) \neq C_{f,D}(x)\right) \end{aligned} \tag{4}$$

where $f_D(x)$ is the predicted output with the model trained with data set D, $f(x)$ is the desired output and $C_{f,D}$ is the central tendency.

One should recall that we just have a limited number of samples and, therefore, the estimation is done by performing $n \times k$-fold cross validation, which is expected to be approximations of those components. We adopted this sampling technique because it has the advantage that every sample is used for training and testing, and each of them is evaluated n times.

$f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ are the functions to be approximated by the surrogates. In consequence, given a new individual, the surrogates give an approximation of the bias and variance component of the models. The best individuals, according to the values given by the surrogates, are selected to perform the $n \times k$-fold cross validation to compute such components.

Hence, the goal of our proposal is to explore the hyper-parameters space looking for models that offer the best possible trade-off between the bias and variance. At the end, a set of non-dominated solutions is obtained, from which a single one is chosen. The remainder of this section explains this in more detail.

3.6. Constructing a Final Classification Model for a Classification Task

As a result of the multi-objective optimization step, a set of non-dominated solutions is obtained, which is expected to be an approximation to the true

Pareto optimal set. In a mathematical sense and in the absence of knowledge about any user's preference, each of these solutions are equally acceptable solutions to the problem at hand. In our case, each solution represents a potential model to be used in the classification task. Therefore, for practical reasons, it is desirable to choose a single solution from the obtained non-dominated set.

We face the issue of selecting a single solution by a simple strategy. One could think to use all solutions that belong to the non-dominated set and to construct and ensemble from these. This could seem reasonable, since each of these different solutions satisfies a different trade-off between the objectives and, therefore, the diversity among them could help to improve the accuracy performance. Notwithstanding, both highly biased models and models with a high variance would also be part of that ensemble, which could affect the generalization performance.

The main goal of model selection is to choose, among a set of candidate models, the one that has the best generalization performance. In this case, the set of non-dominated solutions is the set of candidate models. Hence, we address the selection of a single model in a straightforward fashion, by evaluating the performance of each model with a validation set. Then, we choose as the outcome of our method the one with the lowest error on the validation set. By proceeding in this manner, we expect to avoid overfitting to some extent. In the next section we evaluate our proposed method with a suite of benchmark data sets in order to asses its effectiveness.

4. Experiments and Results

This section describes the experimental study performed to evaluate the feasibility of our proposed method for addressing the model selection for a SVM. It also shows comparisons with state of the art methods for model selection. A statistical analysis is also presented.

4.1. Experimental Settings

For our experiments, we used the IDA[2] benchmark data sets introduced by [30]. This benchmark has been extensively used in several related studies (e.g. [3, 7, 33]). This benchmark contains 13 binary classification data sets.

[2]Available in `http://www.raetschlab.org/Members/raetsch/benchmark`

Table 2: Details of the data sets used in our experiments.

ID	Data set	Feat.	Training Samples	Testing Samples	Replications
1	Banana	2	400	4900	100
2	Breast Cancer	9	200	77	100
3	Diabetes	8	468	300	100
4	Flare Solar	9	666	400	100
5	German	20	700	300	100
6	Heart	13	170	100	100
7	Image	20	1300	1010	20
8	Ringnorm	20	400	7000	100
9	Splice	60	1000	2175	20
10	Thyroid	5	140	75	100
11	Titanic	3	150	2051	100
12	Twonorm	20	400	7000	100
13	Waveform	21	400	4600	100

Table 2 shows some characteristics of these data sets. These data sets were previously pre-processed by [30], in which each data set was divided into 100 partitions for training and test (20 in the cases of image and splice data sets). The typical experimental setup for these data sets was introduced by [30], which consist of performing hyper-parameter selection for the first five partition of each data set. After that, the median values of the hyper-parameters are taken and they are used to evaluate the performance of the model for a data set. This is usually known as the *median protocol*. Nonetheless, the median protocol could introduce an optimistic bias in performance estimation [6]. An alternative protocol consist of performing the model selection independently for each partition of each data set; this is known as the internal protocol.

The performance of the proposed model selection method is assessed by means of the error rate attained on each data set. We compare our experimental results obtained by our proposed method with some evolutionary methods proposed to this aim.

Regarding the parameter configuration used in our experiments, we fixed the crossover probability to 1, the mutation probability was set to 0.10. The distribution index for the crossover (SBX) and for polynomial-based mutation were both set to 20. The number of individuals was fixed to 20 and the number of generations was fixed to 100. The stopping criterion was to reach the maximum number of generations. The size of the validation set was

defined to be 20% of the training set size. The $n \times k$-fold cross validation for estimating bias and variance components were fixed to n equals to 10 and k to 3. Next, the results obtained from the performed experiments are presented.

4.2. Experimental Results

In this section, we present the experimental results obtained by our proposed method. We adopted the internal protocol for our experiments; therefore, a total of 1140 executions of our proposal were performed. Table 3 shows the error rate attained by our proposal (SAMOMS, surrogate-assisted multi-objective model selection) and some reference methods. We compare our results with those obtained by PSMS [16] and SUMO [19], two evolutionary approaches for model selection. As a baseline, Table 3 also shows the results obtained by the SVM, when it is trained with the default parameters.[3] The reported results are the average and standard error over the 100 or 20 replications of each data set.

PSMS [16] is a single-objective model selection method, which uses a particle swarm optimizer (PSO) as the optimization strategy and the fitness function is defined as minimizing the error estimated via k-fold cross validation. On the other hand, SUMO [19] adopts a genetic algorithm to explore the hyper-parameters space. In SUMO, the fitness function can be defined as minimizing some measure obtained via some evaluation strategy; in our experiments, the measure was fixed to be the error rate and the evaluation strategy to be the k-fold cross validation. In both cases, we used the default parameter for k.

From Table 3, one could note that the SVM trained with the default parameters (i.e., model selection is not performed) is not able to reach the best performance when it is compared with others that perform a model selection step. Both SVM and PSMS perform better than the other approaches in 1 out of 13 data sets. SUMO toolbox performs better in 7 out of 13 data sets, whilst SAMOMS performs better in 4 out 13 data sets. In spite of this, SAMOMS reaches the lowest averaged error rate and SUMO is the second best. In order to assess if there exists a statistical significance difference we conducted two types of tests. The first one aims to test the statistical difference per each data set and the second one aims to test the difference among

[3]In the LibSVM package, the default parameters correspond to an SVM trained with a RBF kernel with a gamma value equals to 1/No.-features.

Table 3: Results obtained by our proposed method (SAMOMS) and those obtained by other evolutionary and non-evolutionary approaches. The reported results are the average and standard error over 100 or 20 replications. The best result for each case is shown in **boldface**.

Data Set	SVM	PSMS [16]	SUMO [19]	SAMOMS
Banana	11.17 ± 0.071	11.08 ± 0.083	10.88 ± 0.074	$\mathbf{10.65 \pm 0.054}$
Breast Cancer	34.23 ± 0.466	33.01 ± 0.658	$\mathbf{26.27 \pm 0.448}$	28.22 ± 0.506
Diabetis	32.31 ± 0.223	27.06 ± 0.259	$\mathbf{23.49 \pm 0.177}$	24.46 ± 0.212
Flare Solar	36.29 ± 0.180	34.81 ± 0.173	38.47 ± 0.573	$\mathbf{33.04 \pm 0.236}$
German	28.33 ± 0.258	30.10 ± 0.720	$\mathbf{23.83 \pm 0.213}$	24.55 ± 0.230
Heart	22.95 ± 0.364	20.69 ± 0.634	17.67 ± 0.355	$\mathbf{16.19 \pm 0.373}$
Image	2.99 ± 0.123	2.90 ± 0.112	$\mathbf{2.45 \pm 0.126}$	3.73 ± 0.117
Ringnorm	2.02 ± 0.020	7.98 ± 0.660	$\mathbf{1.72 \pm 0.071}$	1.81 ± 0.035
Splice	10.80 ± 0.162	14.63 ± 0.324	10.94 ± 0.146	$\mathbf{8.31 \pm 0.114}$
Thyroid	4.84 ± 0.221	$\mathbf{4.32 \pm 0.235}$	4.85 ± 0.224	5.09 ± 0.240
Titanic	$\mathbf{22.40 \pm 0.101}$	24.18 ± 0.193	34.99 ± 0.523	23.19 ± 0.217
Twonorm	3.40 ± 0.047	3.09 ± 0.127	$\mathbf{2.55 \pm 0.022}$	2.58 ± 0.034
Waveform	11.43 ± 0.071	12.80 ± 0.325	$\mathbf{9.78 \pm 0.060}$	10.56 ± 0.108
Average	17.17 ± 3.538	17.43 ± 3.204	15.99 ± 3.466	$\mathbf{14.80 \pm 2.980}$

the methods over multiple data sets. In the first case we used an ANOVA test with a 95% of confidence. This test is suitable to compare more than two methods. Moreover, insomuch as our goal is to compare the performance of SAMOMS with respect to the reference results, we performed the Dunnett's test as the post-hoc test. We summarize the results of these tests as follows:

- SAMOMS significantly outperformed to a SVM in 10 out 13 data sets (in the ringnorm and the thyroid data sets, the SVM was not outperformed).
- SAMOMS performs statistically better that PSMS in 11 out of 13 data sets (the statistical tests do not show a statistical difference in the titanic data set) and it was statistically outperformed by PSMS in the thyroid data set.
- With respect to SUMO, SAMOMS significantly outperforms it on 3 data sets (flare solar, splice, and titanic), but it was also significantly outperformed in 3 data sets (diabetis, image, and waveform).

There is no clear advantage between SUMO and SAMOMS; therefore, we conducted a test for the second case (i.e., comparing both methods over multiple data sets). Moreover, we also conducted this kind of test on PSMS so as to compare our proposal with two state of the art model selection methods. To this, we used a Friedman test with a 95% of confidence and a Bonferroni-Dunn test as a post-hoc test. According to these tests, there is a statistical significant difference between PSMS and SAMOMS, but between SUMO and SAMOMS the tests do not show a statistical significance difference.

One should recall that the motivation of approaching the problem using surrogate-assisted optimization is to reduce the number of required fitness functions. Table 4 shows the number of fitness function evaluations performed by each method. One should note that both PSMS and SUMO perform the same number of fitness function evaluations for all cases. SAMOMS, on the other hand, performs a lower number of evaluations. This is due to the fact that the fitness values of some solutions are approximated so as to determine the solutions with the highest potential, which are then evaluated using the corresponding fitness functions. The evaluation of the fitness functions implies to perform a $n \times k$-fold cross validation to estimate the bias and variance components. Therefore, by reducing the number of solutions to be evaluated with such a function, the computational cost should get reduced as well. Table 4 also shows the required time to perform a model selection of each data set and the number of fitness function evaluations performed. The results are the average over all replication of each data set. The time is reported in minutes. We report these values for PSMS, SUMO, and SAMOMS. From this table, one can note that, on average, SAMOMS required a lower time than the other methods. For some data sets, SUMO required less time than SAMOMS (breast cancer, flare solar, heart, thyroid, and titanic). This is due to the fact that SUMO performs an automatic selection of data points, requiring for some data sets a lower number of points, which makes faster the training step. PSMS also required less time than SAMOMS on the thyroid data set. SAMOMS performed faster than the other approaches in the two largest data sets considered in our experiments (image and splice).

Overall, the performance of both SUMO and SAMOMS is quite similar in terms of the generalization performance of the models. Neither SUMO nor SAMOMS are statistically superior to the other one. Notwithstanding, SAMOMS was found to require less time than SUMO for most of the data sets, and it also required a lower number of fitness function evaluations. This is an interesting result, since SAMOMS is able to construct classification

Table 4: Time (in minutes) required and number of fitness functions evaluation performed by each method. The reported values are the average and standard deviation over the 100 or 20 replications performed of each data set.

Data Set	Time			Evaluations		
	PSMS	SUMO	SAMOMS	PSMS	SUMO	SAMOMS
Banana	16.45 ± 2.06	13.24 ± 2.30	6.68 ± 3.73	1000 ± 0.00	1000 ± 0.00	506.21 ± 89.00
Breast Cancer	7.86 ± 0.99	3.10 ± 0.10	3.93 ± 0.48	1000 ± 0.00	1000 ± 0.00	482.59 ± 93.17
Diabetis	14.67 ± 1.69	18.1 ± 0.15	4.50 ± 0.47	1000 ± 0.00	1000 ± 0.00	427.47 ± 85.19
Flare Solar	17.35 ± 2.08	0.94 ± 0.06	5.03 ± 0.69	1000 ± 0.00	1000 ± 0.00	609.41 ± 121.08
German	21.44 ± 0.80	40.67 ± 0.49	5.95 ± 0.86	1000 ± 0.00	1000 ± 0.00	488.33 ± 90.74
Heart	8.28 ± 1.47	2.69 ± 0.02	5.24 ± 0.43	1000 ± 0.00	1000 ± 0.00	592.13 ± 114.61
Image	163.49 ± 3.04	222.8 ± 1.17	68.72 ± 1.27	1000 ± 0.00	1000 ± 0.00	503.15 ± 91.10
Ringnorm	19.38 ± 2.81	12.75 ± 0.53	6.29 ± 0.89	1000 ± 0.00	1000 ± 0.00	514.10 ± 102.87
Splice	78.12 ± 21.32	72.30 ± 3.84	44.02 ± 12.66	1000 ± 0.00	1000 ± 0.00	512.50 ± 99.04
Thyroid	3.54 ± 1.15	1.76 ± 0.03	4.99 ± 1.11	1000 ± 0.00	1000 ± 0.00	514.89 ± 94.80
Titanic	20.68 ± 4.74	0.32 ± 0.01	5.53 ± 3.97	1000 ± 0.00	1000 ± 0.00	507.30 ± 101.18
Twonorm	14.59 ± 2.59	12.80 ± 1.61	6.16 ± 0.59	1000 ± 0.00	1000 ± 0.00	516.39 ± 102.26
Waveform	8.77 ± 2.19	13.50 ± 1.01	6.07 ± 1.24	1000 ± 0.00	1000 ± 0.00	506.96 ± 92.04
Average	30.36 ± 44.09	31.90 ± 60.77	13.31 ± 19.78	1000 ± 0.00	1000 ± 0.00	513.96 ± 45.30

models without significantly overfitting with a lower number of evaluations than SUMO. Evidently, this could translate in important computational time savings when model selection is performed on large-scale data sets.

5. Conclusions and Future Work

In this paper, we introduced SAMOMS, a surrogate-assisted multi-objective model selection method for support vector machines. In our proposed method, the model selection problem is formulated as a multi-objective one, taking into account to components that are closely related to the generalization error: bias and variance.

This paper has made the following contributions: (i) the definition of bias and variance for addressing the model selection can be applicable to any other learning algorithm, which allows to generalize our proposal and, to make it applicable to the full model selection problem; (ii) the use of surrogate-assisted optimization as an alternative allows to reduce the number of evaluations performed, which can be translated into important computational time savings; (iii) a resulting non-dominated front with models having different bias/variance trade-offs could be useful to an expert for performing an analysis of the behavior of such model, given a certain data set.

Experimental results showed that our proposed approach is able to choose models having a highly competitive performance, while requiring a lower number of fitness function evaluations and a lower computational time. This is because, with our proposed approach, a lower number of models needs to be trained.Statistical tests showed that, for most of the data sets, the generated models do not significantly degrade their performance when they are compared to those resulting from other state of the art model selection methods. Based on this, we can conclude that SAMOMS can be seen as a viable tool for dealing with the problem of model selection in classification tasks.

As part of our future work, we would like to extend the proposed method to other algorithms. We are also interested in studying the feasibility of our proposed approach for the full model selection problem. Additionally, we are interested in developing new strategies for choosing a subset of solutions for ensemble construction.

Acknowledgments

The first author is grateful for the support from CONACyT scholarship no. 329013.

References

References

[1] Ayat, N., Cheriet, M., Suen, C., 2005. Automatic model selection for the optimization of SVM kernels. Pattern Recognition 38, 1733 – 1745. doi:http://dx.doi.org/10.1016/j.patcog.2005.03.011.

[2] Aydin, I., Karakose, M., Akin, E., 2011. A multi-objective artificial immune algorithm for parameter optimization in support vector machine. Applied Soft Computing 11, 120 – 129. doi:10.1016/j.asoc.2009.11.003.

[3] Bao, Y., Hu, Z., Xiong, T., 2013. A PSO and pattern search based memetic algorithm for SVMs parameters optimization. Neurocomputing 117, 98 – 106. doi:10.1016/j.neucom.2013.01.027.

[4] Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[5] Boser, B.E., Guyon, I.M., Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers, in: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, ACM. pp. 144–152. doi:10.1145/130385.130401.

[6] Cawley, G.C., Talbot, N.L., 2010. On over-fitting in model selection and subsequent selection bias in performance evaluation. Journal of Machine Learning Research 11, 2079–2107.

[7] Cawley, G.C., Talbot, N.L.C., 2007. Preventing over-fitting during model selection via bayesian regularisation of the hyper-parameters. Journal of Machine Learning Research 8, 841–861.

[8] Chang, C.C., Lin, C.J., 2011. LIBSVM: A library for support vector machines. ACM Trans. Intell. Syst. Technol. 2, 27:1–27:27. doi:10.1145/1961189.1961199.

[9] Chapelle, O., Vapnik, V., Bousquet, O., Mukherjee, S., 2002. Choosing multiple parameters for support vector machines. Machine Learning 46, 131–159. doi:10.1023/A:1012450327387.

[10] Chatelain, C., Adam, S., Lecourtier, Y., Heutte, L., Paquet, T., 2010. A multi-model selection framework for unknown and/or evolutive misclassification cost problems. Pattern Recogn. 43, 815 – 823. doi:10.1016/j.patcog.2009.07.006.

[11] Coello Coello, C.A., Lamont, G.B., Veldhuizen, D.A.V., 2007. Evolutionary Algorithms for Solving Multi-Objective Problems. Genetic and Evolutionary Computation. 2 ed., Springer US.

[12] Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20, 273–297. doi:10.1007/BF00994018.

[13] Deb, K., 2001. Multi-Objective Optimization Using Evolutionary Algorithms. Wiley.

[14] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6, 182–197. doi:10.1109/4235.996017.

[15] Domingos, P., 2000. A unified bias-variance decomposition for zero-one and squared loss, in: Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, AAAI Press. pp. 564–569.

[16] Escalante, H.J., Montes, M., Sucar, L.E., 2009. Particle swarm model selection. Journal of Machine Learning Research 10, 405–440.

[17] Friedman, J.H., 1997. On bias, variance, 0/1—loss, and the curse-of-dimensionality. Data Min. Knowl. Discov. 1, 55–77. doi:10.1023/A:1009778005914.

[18] Gomes, T.A.F., Prudencio, R.B.C., Soares, C., Rossi, A.L.D., Carvalho, A., 2010. Combining meta-learning and search techniques to svm parameter selection, in: Proceedings of the 2010 Eleventh Brazilian Symposium on Neural Networks, IEEE Computer Society. pp. 79–84. doi:10.1109/SBRN.2010.22.

[19] Gorissen, D., Dhaene, T., Turck, F.D., 2009. Evolutionary model type selection for global surrogate modeling. Journal of Machine Learning Research 10, 2039–2078.

[20] Hastie, T., Tibshirani, R., Friedman, J., 2009. The elements of statistical learning: data mining, inference and prediction. Second ed., Springer.

[21] James, G.M., 2003. Variance and bias for general loss functions. Mach. Learn. 51, 115–135. doi:10.1023/A:1022899518027.

[22] Ji, Y., Sun, S., 2010. Feature selection for ensembles using non-dominated sorting in genetic algorithms, in: 2010 Sixth International Conference on Natural Computation (ICNC), pp. 888 –891.

[23] Kohavi, R., Wolpert, D., 1996. Bias plus variance decomposition for zero-one loss functions, in: Proceedings of the 13th ICML, Morgan-Kaufmann Publishers. pp. 275–283.

[24] Kong, E., Dietterich, T., 1995. Error-correcting output coding corrects bias and variance, in: Proceedings of the 12th ICML, Morgan-Kaufmann Publishers. pp. 313–321.

[25] Li, W., Liu, L., Gong, W., 2011. Multi-objective uniform design as a svm model selection tool for face recognition. Expert Syst. Appl. 38, 6689 – 6695. doi:10.1016/j.eswa.2010.11.066.

[26] Lorena, A.C., de Carvalho, A.C., 2008. Evolutionary tuning of SVM parameter values in multiclass problems. Neurocomputing 71, 3326 – 3334. doi:10.1016/j.neucom.2008.01.031.

[27] Miettinen, K., 1999. Nonlinear Multiobjective Optimization. volume 12 of *International Series in Operations Research and Management Science*. Kluwer Academic Publishers, Dordrecht.

[28] Miranda, P.B.C., Prudencio, R.B.C., Carvalho, A.C.P.L.F., Soares, C., 2012a. Combining meta-learning with multi-objective particle swarm algorithms for svm parameter selection: An experimental analysis, in: Proceedings of the 2012 Brazilian Symposium on Neural Networks, IEEE Computer Society, Washington, DC, USA. pp. 1–6. doi:10.1109/SBRN.2012.12.

[29] Miranda, P.B.C., Prudencio, R.B.C., Carvalho, A.C.P.L.F., Soares, C., 2012b. Multi-objective optimization and meta-learning for svm parameter selection, in: Neural Networks (IJCNN), The 2012 International Joint Conference on, pp. 1–8. doi:10.1109/IJCNN.2012.6252378.

[30] Rätsch, G., Onoda, T., Müller, K.R., 2001. Soft margins for adaboost. Machine Learning 42, 287–320. doi:10.1023/A:1007618119488.

[31] Reif, M., Shafait, F., Dengel, A., 2012. Meta-learning for evolutionary parameter optimization of classifiers. Machine Learning 87, 357–380. doi:10.1007/s10994-012-5286-7.

[32] Rosales-Pérez, A., Coello Coello, C.A., Gonzalez, J.A., Reyes-Garcia, C.A., Escalante, H.J., 2013a. A hybrid surrogate-based approach for evolutionary multi-objective optimization, in: 2013 IEEE Congress on Evolutionary Computation (CEC'2013), pp. 2548–2555. doi:10.1109/CEC.2013.6557876.

[33] Rosales-Pérez, A., Escalante, H.J., Gonzalez, J.A., Reyes-Garcia, C.A., Coello Coello, C.A., 2013b. Bias and variance multi-objective optimization for support vector machines model selection, in: Sanches, J.a.M., Micó, L., Cardoso, J.S. (Eds.), Pattern Recognition and Image Analysis, Springer Berlin Heidelberg. pp. 108–116. doi:10.1007/978-3-642-38628-2_12.

[34] Saffari, A., Guyon, I., 2006. Quick start guide for CLOP. Technical Report. Technical Report, May 2006. http://ymer.org/research/files/clop/QuickStartV1. 0. pdf.

[35] Schaffer, J.D., 1985. Multiple objective optimization with vector evaluated genetic algorithms, in: Proceedings of the 1st International Conference on Genetic Algorithms, L. Erlbaum Associates Inc., Hillsdale, NJ, USA. pp. 93–100.

[36] Soares, C., Brazdil, P.B., Kuba, P., 2004. A meta-learning method to select the kernel width in support vector regression. Mach. Learn. 54, 195–209. doi:10.1023/B:MACH.0000015879.28004.9b.

[37] Suttorp, T., Igel, C., 2006. Multi-objective optimization of support vector machines, in: Jin, Y. (Ed.), Multi-Objective Machine Learning.

Springer Berlin / Heidelberg. volume 16 of *Studies in Computational Intelligence*, pp. 199–220. doi:10.1007/3-540-33019-4_9.

[38] Valentini, G., 2005. An experimental bias-variance analysis of svm ensembles based on resampling techniques. Trans. Sys. Man Cyber. Part B 35, 1252–1271. doi:10.1109/TSMCB.2005.850183.

[39] Valentini, G., Dietterich, T.G., 2004. Bias-variance analysis of support vector machines for the development of svm-based ensemble methods. J. Mach. Learn. Res. 5, 725–775.

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

[41] Webb, G.I., 2000. Multiboosting: A technique for combining boosting and wagging. Mach. Learn. 40, 159–196. doi:10.1023/A:1007659514849.

[42] Yu, X., Gen, M., 2010. Introduction to Evolutionary Algorithms. Springer.

[43] Zhang, Q., Li, H., 2007. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions onEvolutionary Computation 11, 712–731. doi:10.1109/TEVC.2007.892759.

[44] Zitzler, E., Laumanns, M., Thiele, L., 2001. SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization, in: Giannakoglou, K.C., Tsahalis, D.T., Périaux, J., Papailiou, K.D., Fogarty, T. (Eds.), Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems, International Center for Numerical Methods in Engineering. pp. 95–100.