# Combining a Multi-Objective Optimization Approach with Meta-Learning for SVM Parameter Selection

**Conference Paper** · October 2012

DOI: 10.1109/ICSMC.2012.6378235



**4 authors**, including:

Péricles B. C. Miranda
Universidade Federal Rural de Pernambuco
**65** PUBLICATIONS **386** CITATIONS

SEE PROFILE

Ricardo B. C. Prudêncio
Federal University of Pernambuco
**88** PUBLICATIONS **1,458** CITATIONS

SEE PROFILE

Carlos Soares
University of Porto
**365** PUBLICATIONS **4,282** CITATIONS

SEE PROFILE

2012 IEEE International Conference on Systems, Man, and Cybernetics
October 14-17, 2012, COEX, Seoul, Korea

# Combining a Multi-Objective Optimization Approach with Meta-Learning for SVM Parameter Selection

Péricles B. C. de Miranda
and Ricardo B. C. Prudêncio
Center of Informatics
Federal University of Pernambuco
Recife, Brazil
pbcm, rbcp@cin.ufpe.br

Andre Carlos P. L. F. de Carvalho
University of São Paulo
São Carlos, Brazil
andre@icmc.usp.br

Carlos Soares
INESC TEC
University of Porto
Porto, Portugal
csoares@fep.up.pt

***Abstract*—Support Vector Machine (SVM) is a supervised technique, which achieves good performance on different learning problems. However, adjustments on its model are essentials to the SVM work well. Optimization techniques have been used to automatize this process finding suitable configurations of parameters which attends some learning problems. This work utilizes Particle Swarm Optimization (PSO) applied to the SVM parameter selection problem. As the learning systems are essentially a multi-objective problem, a multi-objective PSO (MOPSO) was used to maximize the success rate and minimize the number of support vectors of the model. Nevertheless, we propose the combination of Meta-Learning (ML) with a modified MOPSO which uses the crowding distance mechanism (MOPSO-CDR). In this combination, solutions provided by ML are possibly located in good regions in the search space. Hence, using a reduced number of successful candidates, the search process would converge faster and be less expensive. In our work, we implemented a prototype in which MOPSO-CDR was used to select the values of two SVM parameters for classification problems. In the performed experiments, the proposed solution (MOPSO-CDR using ML) was compared to the MOPSO-CDR with random initialization, obtaining pareto fronts with higher quality on a set of 40 classification problems.**

***Index Terms*—Meta-Learning, Multi-Objective Optimization, SVM Parameter Selection, Particle Swarm Optimization**

## I. Introduction

The SVM performance strongly depends on the adequate choice of its parameters, and an exhaustive trial-and-error procedure for selecting good values of parameters is not practical for computational reasons[14]. Hence, the selection of SVM parameters is commonly treated by different authors as an optimization problem in which a search technique is used to find adequate configurations of parameters for the problem at hand. In literature, different techniques were applied to this problem including Evolutionary Algorithms (EA) [3], Particle Swarm Optimization (PSO) [3] and Tabu Search [8]. Previous work commonly used single objective techniques for SVM parameter selection, however this is not totally adequate since this task is inherently a Multi-Objective Optimization (MOO) problem [9]. In this context, we can mention the use of Multi-Objective EA (MOEA) [9], Multi-Objective PSO (MOPSO) [15] and the use of Gradient-Based techniques [14], which considered multiple objectives. Although the application of MOO search techniques represents an automatic and suitable solution to select SVM parameters, this approach can be very expensive and with a low convergence, since a large number of candidate configurations of parameters is often evaluated during the search [1] and the number of restrictions (objectives) to be analysed.

A possible solution for this problem is the use of Meta-Learning (ML), which treats parameter selection as a supervised learning task [1][18]. Each training example for ML (i.e. each meta-example) stores the characteristics of a past problem and the performance obtained by a set of candidate configurations of parameters on the problem. By receiving a set of such meta-examples, a meta-learner predicts the most suitable configuration of parameters for a new problem based on its characteristics. ML is a less expensive solution compared to the search approach. In fact, once the knowledge is acquired by the meta-learner, configurations of parameters can be suggested for new problems without the need of empirically evaluating several candidate configurations. However, ML is very dependent on the quality of its meta-examples. In general the number of problems available for meta-example generation is commonly limited and noisy, requiring a careful treatment of the data.

In a recent work [19], ML and search techniques were combined for SVM parameter selection. In this work, ML was adopted to suggest a number of solutions (configurations of parameters) which are adopted as the initial population of the search technique. The search technique just refined a promising solution returned by ML, speeding up the optimization process. The authors adopted as search technique the single objective PSO, whose objective was to minimize the error rate obtained by the SVM. Despite the good results obtained by this combination, as said the use of multiple objectives would be more adequate [2].

In the current work, we propose the combination of a swarm-based multi-objective search technique and ML to the problem of SVM parameter selection. We emphasize that the combination of swarm-based algorithms with ML was just studied in the single objective field and the use of multi-objective optimization has not been investigated in this context.

978-1-4673-1714-6/12/$31.00 ©2012 IEEE

In this proposal, configurations of parameters suggested by ML are adopted as initial solutions which will be later refined by the search technique. Hence, we expect that ML guides the search directly to promising regions of the search space, thus speeding up the convergence to good solutions.

In order to evaluate our proposal, we implemented the MOPSO using Crowding Distance and Roulette Wheel (MOPSO-CDR) algorithm [11] used as the search technique to optimize the parameters: $\gamma$ of the RBF kernel and the regularization constant $C$, which may have a strong influence in SVM performance [10]. The optimization considers two conflicting objectives: complexity and success rate on classification [16][2][17]. In our work, a database of 40 meta-examples was produced from the evaluation of a set of 399 configurations of ($\gamma$, $C$) on 40 different classification problems. Each classification problem was described by a number of 8 meta-features proposed in [22].

Our experiments evaluated the MOPSO-CDR in two different versions: (1) MOPSO-CDR with initial population suggested by ML (Hybrid MOPSO-CDR ou HMOPSO-CDR) and (2) MOPSO-CDR with random initial population. The results revealed that the hybrid method was able to generate better solutions during the generations when compared to the randomly initialized MOPSO-CDR, according the quality metrics chosen in the experiment.

This paper is presented as follows: Section II brings a brief presentation of basic concepts of particle swarm optimization and essential adjustments to make this algorithm adequate to support multi-objectives. Section III presents details of the proposed work. Section IV describes the experiments and obtained results. Finally, Section V presents some conclusions and the future work.

## II. Basic Concepts of PSO

PSO is a population based algorithm. Each particle $i$ of the population represents a possible solution and has four main attributes: the position in the search space $\vec{x_i}(t)$, the current velocity $\vec{v_i}(t)$, the best position found by the particle $\vec{p_i}(t)$ and the best solution found by its neighborhood $\vec{n_i}(t)$ until the current iteration. The particles move through the search space updating its velocities based on these best solutions already found during the search process so far ($\vec{p_i}(t)$ and $\vec{n_i}(t)$).

At each iteration, the velocities and the positions of all particles are updated according to the equations (1) and (2), respectively. There are, at least, three different equation to update the velocity. In this paper, we used the equation developed by Shi and Eberhart [7]:

$$\vec{v_i}(t+1) = w\vec{v_i}(t) + c_1 r_1(\vec{p_i} - \vec{x_i}(t)) + c_2 r_2(\vec{n_i}(t) - \vec{x_i}(t)), \quad (1)$$

$$\vec{x_i}(t+1) = \vec{x_i} + \vec{v_i}(t+1), \quad (2)$$

where $w$ is the inertia factor; $i = 1, ..., N$, where $N$ is the number of particles; $c_1$ and $c_2$ are the cognitive and the social acceleration coefficients; $r_1$ and $r_2$ are two random numbers generated by using an uniform probability density function in the interval [0,1].

### A. PSO using Multiple Objectives

The PSO is an optimization and search technique that has been used due to its simplicity and convergence velocity. However, many real optimization problems have series of conflicting objectives to be solved. Hence, the PSO was adapted to support multi-objectives [13]. The multi-objective optimization tries to find solutions which satisfies the restrictions of a given problem. A general multi-objective optimization minimization problem can be defined as [13]:

$$\text{minimize } \vec{f}(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), ..., f_n(\vec{x})], \quad (3)$$

subject to:

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, ..., p, \quad (4)$$

$$h_j(\vec{x}) = 0 \quad j = 1, 2, ..., q, \quad (5)$$

where $\vec{x} = (x_1, x_2, ..., x_m) \in \mathbb{R}^n$ is the vector on the decision search space; $n$ is the number of objectives and $g_i(\vec{x})$ and $h_j(\vec{x})$ are the constraint functions and $p + q$ is the number of constraints of the problem. Given two vectors $\vec{z}, \vec{u} \in \mathbb{R}^n$, $\vec{z}$ dominates $\vec{u}$ (denoted by $\vec{z} \prec \vec{u}$) if $\vec{z}$ is better than $\vec{u}$ in at least one objective and $\vec{z}$ is not worse than $\vec{u}$ in any objective. $\vec{z}$ is not dominated if does not exist another current solution $\vec{z_i}$ in the current population, such that $\vec{z_i} \prec \vec{z}$. The set of non-dominated solutions in the objective space is known as *pareto front*.

The set of non-dominated solutions in the objective space is known as Pareto Front. This work treats the SVM Parameters Selection problem as a multi-objective problem, with two conflicting objectives, whose goal is to generate SVM model with high performance rate and low complexity. The first objective is to maximize the success rate, and the second objective is to minimize the number of support vectors.

In this work we used the MOPSO-CDR algorithm, a version of the MOPSO using the Crowding Distance and Roulette Wheel mechanism (CDR) [4]. This algorithm was originally proposed by Santana [11]. It was inspired on the MOPSO-CDLS algorithm, proposed by Tsou *et. al* [12], and incorporates a roulette wheel selection based on the crowding distance to select the $\vec{n_i}(t)$ in eq. 1. This mechanism increases the chance of selecting solutions with greater crowding distance as $\vec{n_i}(t)$. A greater crowding distance means the a region from pareto front is less populated. The mutation operator is the same used in the MOPSO [13] and it is applied at each iteration as well as the MOPSO. So, in order to develop the MOPSO-CDR we added the CDR mechanism to the MOPSO previously created.

## III. Developed Work

This work proposes the combination of MOO algorithms with meta-learning (hybrid approach), where meta-learning is applied to suggest initial solutions (well-succeed configurations) to optimization algorithms, making the search process start in a possible promised region.

Figure 1 presents the general architecture of the proposed solution. Initially, the Meta-Learner module retrieves a predefined number of past meta-examples stored in a Database (DB), selected on the basis of their similarity to the input problem. The process of suggesting meta-examples is not trivial. As we are dealing with a multi-objective problem, the dominance evaluation, showed in the previous section, defines the set of non-dominated solutions among all configurations. After that, the Meta-learner is able to perform the suggestion of non-dominated meta-examples. Following, the Search module adopts as initial search points the configurations of parameters which were well-succeeded on the retrieved meta-examples. The search module iterates its search process by generating new candidate configurations to be evaluated in the SVM. The output configurations of parameters will be the non-dominated solutions generated by the Search module up to its convergence or another stopping criteria.

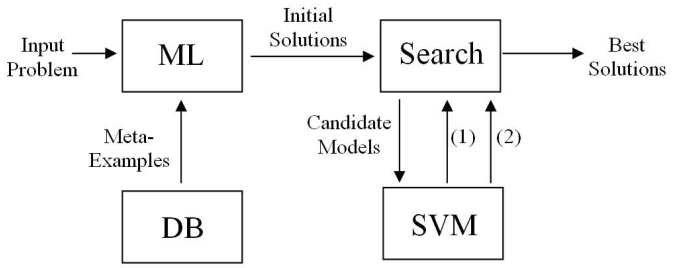


Fig. 1. General Architecture, where (1) is the success rate and (2) the number of support vectors.

### A. Search Module

In our prototype, we implemented a version of the MOPSO called MOPSO-CDR algorithm. The main characteristic of this algorithm is to increase the diversity of the pareto front using the crowding distance and roulette wheel mechanism. Researches in optical networks [5] and computational intelligence [6] have used this algorithm achieving interesting results.

*1) MOPSO-CDR to SVM Parameter Selection:* The MOPSO-CDR was adapted here to perform the search where each particle represents a configuration ($\gamma$, $C$), indicating the position of the particle in the search space. The objective functions evaluate the quality and complexity of each configuration of parameters on a given classification problem. In our work, given a SVM configuration, we define two objective functions: the success rate (SR) and the number of support vectors (NSV) obtained by the SVM in a 10-fold cross validation experiment. So, the objectives of MOPSO-CDR is to find the non-dominated configurations of ($\gamma$, $C$) trying to maximize the SR and minimize the NSV for a given classification problem.

In our work, the MOPSO-CDR was implemented to perform a search in a space represented by a discrete grid of SVM configurations, consisting of 399 different settings of parameters $\gamma$ and $C$. By following the guidelines provided in [20], we considered the following exponentially growing sequences of $\gamma$ and $C$ as potentially good configurations: the parameter $\gamma$ assumed 19 different values (from $2^{-15}$ to $2^{3}$) and the parameter $C$ assumed 21 different values (from $2^{-5}$ to $2^{15}$), thus yielding 19 x 21 = 399 different combinations of parameters in the search space.

### B. Meta-Database

To create meta-examples, we collected 40 datasets corresponding to 40 different classification problems, available in the UCI Repository [21]. Each meta-example is related to a single classification problem and stores: (1) a vector of meta-features describing the problem; and (2) the objective grid which stores the success rate and number of support vectors obtained by the SVM in the search space of configurations ($\gamma$, $C$). The objective grid consists of 399 different settings of parameters $\gamma$ and $C$.

*1) Meta-Features:* In this work, we used 8 meta-features to describe the datasets of classification problems. These meta-features were selected from the set of features defined in [22]. We adopted meta-features divided in three categories: 1) Simple, composed by *Number of examples*, *attributes* and *classes*, 2) Statistical, composed by *Mean correlation of attributes*, *Skewness* and *Kurtosis*, and 3) Information Theory, composed by *Entropy of class*.

The group of statistical values is composed by the *mean correlation of attributes*; *Skewness*, which measure the asymmetry of the distribution regarding the central axis [22]; *Kurtosis*, which measure the dispersion (characterized by the flatness of the distribution curve) [22] and the *geometric mean of the attributes* which evaluates the mean of the data standard deviation. Finally, the information theory group measure the randomness of the instances; being composed by the *Entropy* which defines the degree of uncertainty of classification [22].

*2) Objective Grid:* The objective grid stores the SR and NSV obtained by the SVM on a problem considering different SVM configurations. For each one of the 399 configurations, a 10-fold cross validation experiment was performed to collect SVM SR and NSV. The obtained 399 objective values were stored in the objective grid. We highlight here that the objective grid is equivalent to the search space explored by the search technique. By generating a objective grid for a problem, we can evaluate which configurations of parameters were the best ones in the problem (i.e., the best points in a search space) and we can use this information to guide the search process for new similar problems.

### C. Meta-Learner

Given a new input problem described by the vector $\vec{i} = (i_1, ..., i_p)$, the Meta-Learner selects the $k$ most similar problems according to the distance between meta-attributes. The distance function implemented was the Euclidean Distance. After that, we apply the dominance evaluation in the 399 configurations and generate a pareto front for each one of the $k$ most similar problems. In order to suggest an initial population, we select one random solution of each produced pareto front. Random non-dominated solutions of different problems were sampled in order to enhance diversity of the initial population.

## IV. Experiments

In this section, we present the experiments which evaluated the proposed solution on the set of 40 classification problems considered in our work. The proposed solution was evaluated by following a leave-one-out methodology described bellow.

At each step of leave-one-out, one meta-example was left out to evaluate the implemented prototype and the remaining 39 meta-examples were considered in the DB to be selected by the ML module. A number of $k$ meta-examples were suggested by the ML module as the initial MOPSO-CDR population (in our experiments, we adopted $k$ = 7). The MOPSO-CDR then optimized the SVM configurations for the problem left out up to the number of 10 generations. In each generation, a pareto front (repository of non-dominated solutions) is formed and to evaluate its quality we applied three metrics. This procedure was repeated 30 times to guarantee reliability.

This experiment was divided in two parts: 1) the number of wins of the algorithms per generation regarding all problems, where the algorithm which achieved better quality (according to an specific metric) is the winner and 2) the mean of the metrics values of all problems for each generation. As a basis of comparison, we used for each value of $k$ a randomly initialized population for MOPSO-CDR. Despite its simplicity, the random initialization has the advantage of performing a uniform initial exploration of the search space. Finally, we highlight that each evaluated version of MOPSO-CDR was executed 30 times and the average results were recorded.

### A. Metrics

In our experiments, we evaluated the results (i.e., the pareto fronts) obtained by all hybrid and traditional algorithms for each problem according to different quality metrics usually adopted in the literature of MOO. The adopted metrics were: Hypervolume, Maximum Spread and Coverage. Each metric considers a different aspect of the pareto front.

*1) Hypervolume $HV$:* this metric was proposed by Zitzler and Thiele [23] and is defined by the hypervolume in the space of objectives covered by the obtained pareto front ($\mathcal{P}^*$). For MOP with $w$-objectives, *HV* is defined by:

$$HV = \left\{ \bigcup_i a_i \,|\, s_i \in \mathcal{P}^* \right\}, \tag{6}$$

where $s_i$ $(i = 1, 2, ..., n)$ is a non-dominated solution of the pareto front ($P^*$), $n$ is the number of solutions in the pareto front and $a_i$ is the hypervolume of the hypercube delimited by the position of solution $s_i$ in the space of objectives and the origin. In practice, this metric gives the size of the dominated space, which is also called the *area under the curve*. A large value of $HV$ is desired.

*2) Maximum Spread $MS$:* it was proposed by Zitzler *et. al* [23] and evaluates the maximum extension covered by the non-dominated solutions in the pareto front. *MS* is computed by using Eq. (7).

$$MS = \sqrt{\sum_{m=1}^{P} (max_{i=1}^{n} f_m^i - min_{i=1}^{n} f_m^i)^2}, \tag{7}$$

where $n$ is number of solutions in the pareto front and $k$ is the number of objectives. This measure determines which solution covers a bigger extension of the search space. Hence, large values of this metric are preferred.

*3) Spacing $SP$:* this metric estimates the diversity of the achieved pareto front. $SP$ is derived by computing the relative distance between adjacent solutions of the Pareto Front as follows:

$$SP = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\overline{d} - d_i)^2}, \tag{8}$$

where $n$ is the number of non-dominated solutions, $d_i$ is the distance between adjacent solutions to the solution $v_i$ and $\overline{d}$ is the average distance between the adjacent solutions. $S = 0$ means that all solutions of the Pareto Front are equally spaced. Hence, values of $SP$ near zero are preferred.

### B. Algorithms Settings

In this section, we present the values of the parameters adopted for the HMOPSO-CDR and the MOPSO-CDR algorithms. For that, we adopted the same values suggested by Coello *et. al* [13] and by Santana *et. al* [11]:

- number of particles: 7
- mutation rate: 0.5
- inertia factor $\omega$: linearly decreases from 0.9 to 0.4
- constants $c_1$ and $c_2$: 1.49
- Pareto Front's size: 10 solutions
- number of iterations: 10

In our work, we also performed experiments using a *purely random* search method, as a basis of comparison.

### C. Results

In order to analyse our results adequately we performed statistical analysis. As the data does not follow a normal distribution, we applied the Wilcoxon test to verify our hypothesis: the HMOPSO-CDR is superior than MOPSO-CDR in each metric. All the following analysis used this methodology.

Figures 2, 3 and 4 show the number of wins of the HMOPSO-CDR, MOPSO-CDR and Random regarding $HV$, $SP$ and $MS$ respectively. This analysis intend to count the number of classification problems that each technique won per generation.

As it can be seen in Figure 2, the pareto front of HMOPSO-CDR, in the majority problems, overcame the pareto front formed by MOPSO-CDR, for each generation, according to $HV$ metric. The same success is showed in Figure 3 for the $SP$ metric. In most of the problems, the HMOPSO-CDR generated more well-distributed pareto fronts regarding the pareto fronts of MOPSO-CDR. An explanation for this is that depending on the problem, good solutions can be positioned closely decreasing the Spacing metric. Regarding the $MS$ metric, Figure 4 shows that HMOPSO-CDR presented a superior number of victories in comparison to MOPSO-CDR, generating pareto fronts with higher quality. As it can be seen, the influence of ML-based suggestions was crucial

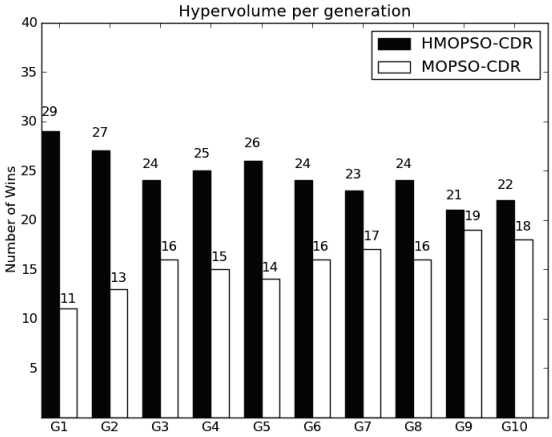


Fig. 2. Number of wins regarding $HS$ using $k = 7$ for 10 iterations.

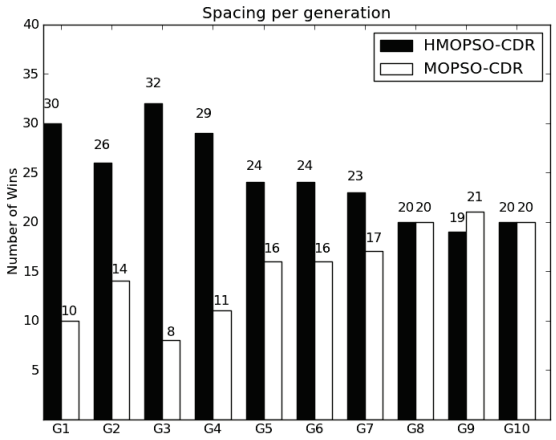


Fig. 3. Number of wins regarding $SP$ using $k = 7$ for 10 iterations.

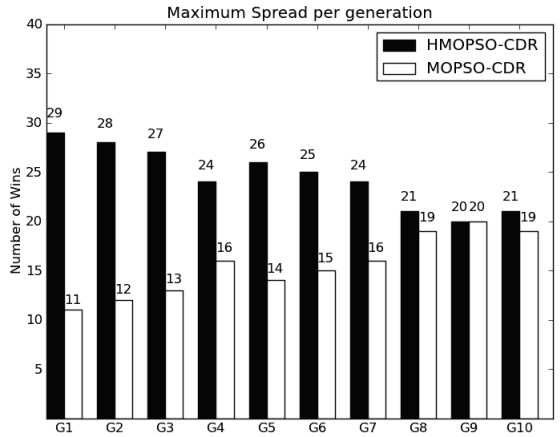


Fig. 4. Number of wins regarding $MS$ using $k = 7$ for 10 iterations.

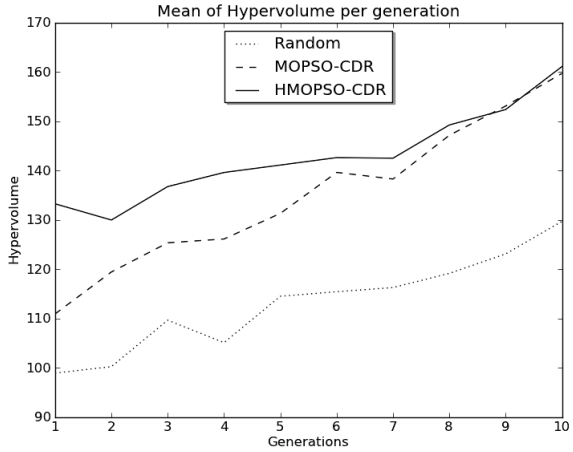


Fig. 5. Average of $HV$ values per generation using $k = 7$ for 10 iterations.

for the initial generations, making our proposal dominate the MOPSO-CDR in all metrics.

Figures 5, 6 and 7 show the average of the metrics' values per generation. These figures compare the Random approach, MOPSO-CDR and HMOPSO-CDR regarding the quality of the pareto per generation. As we can see, the Random approach was outperformed by MOPSO-CDR and HMOPSO-CDR in all cases. Moreover, our proposal achieved better results for all metrics overcoming the MOPSO-CDR with 95% of confidence. In the initial generations we can observe that the influence of the ML-based suggestions had a positive effect in most of the selected problems; and this influence permitted the swarm algorithm lead the solutions to better regions, refining even more their values. The HMOPSO-CDR augmented the area under curve or $HV$ metric of its pareto front, overcoming the MOPSO-CDR in all generations, shown in Figure 5. Figure 6 shows the mean performance of the pareto fronts according to $SP$ metric, and as we can see, our proposal refined the search forming a well distributed pareto front. Figure 7 shows the performance regarding $MS$ metric, and as it can be seen, the mechanism of crowding distance fomented the diversity expanding the distance between the extreme particles, making our proposal out-perform the MOPSO-CDR.

## V. Conclusion

In this current work, we combined Meta-Learning and the MOPSO-CDR, a multiple objective particle swarm optimization technique, to select the parameter $\gamma$ of the RBF kernel and the regularization parameter $C$. In the performed experiments, we observed that the proposed approach was able to generate better pareto fronts compared to a randomly initialized MOPSO-CDR, according all presented metrics with 95% of certainty.

In future work, we intend to collect more classification problems in order to increase the number of meta-examples in the meta-database since we believe that the performance of the proposed approach can be improved as more meta-examples are considered. Also, other search techniques can be used and different swarm sizes can be tested. Besides, we intend to use other quality metrics to perform better analysis of pareto fronts. Finally, we intend to evaluate the proposed solution in other case studies, such as in the SVM parameter selection for regression problems.

## Acknowledgment

The authors would like to thank CAPES, FACEPE, FAPESP and CNPq (Brazilian Agencies) for their financial support.

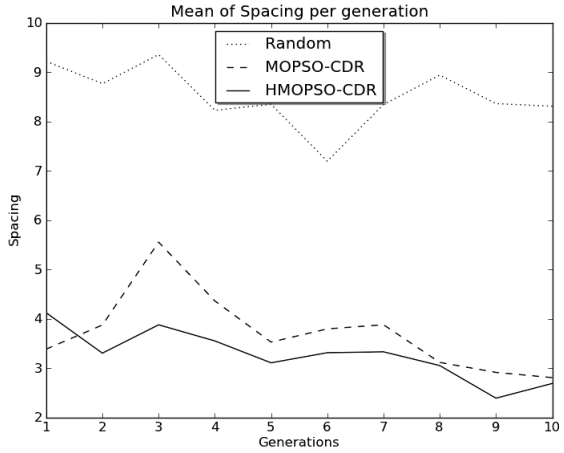


Fig. 6. Average of $SP$ values per generation using $k = 7$ for 10 iterations.

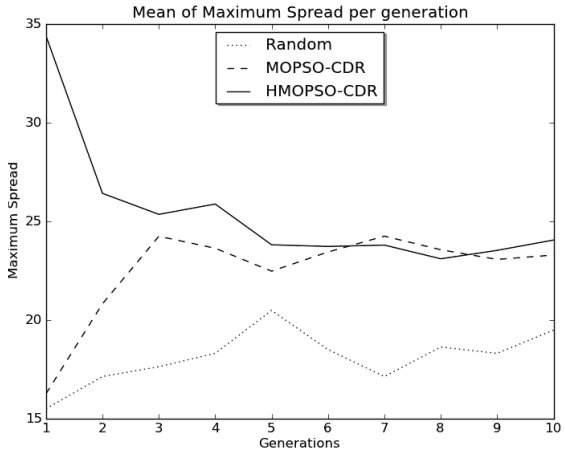


Fig. 7. Average of $MS$ values per generation using $k = 7$ for 10 iterations.

## REFERENCES


[1] C. Soares, P. Brazdil, and P. Kuba. A meta-learning approach to select the kernel width in support vector regression. (4):195–209, 2000.

[2] G. Narzisi. An Experimental Multi-Objective Study of the SVM Model Selection problem.

[3] Yuan Ren and Guangchen Bai. Determination of Optimal SVM Parameters by Using GA/PSO, in Journal of Computers, vol. 5, number 8, 2010, pags. 1160-1168.

[4] Carlo R. Raquel and Jr. Naval and C. Prospero An effective use of crowding distance in multiobjective particle swarm optimization, in Proceedings of the 2005 conference on Genetic and evolutionary computation, pags. 257–264, GECCO '05

[5] Carmelo J. A. Bastos-Filho, Elliackin M. N. Figueiredo, Joaquim F. Martins-Filho, Daniel A. R. Chaves, Marcelo E. V. Segatto, S. Cani and Maria J. Pontes. Design of distributed optical-fiber raman amplifiers using multi-objective particle swarm optimization, in Journal of Microwaves, Optoelectronics and Electromagnetic Applications, 10(2), 323-336, 2011.

[6] Carmelo J.A. Bastos-Filho, and Péricles B.C. Miranda. Multi-Objective Particle Swarm Optimization using speciation, in IEEE Symposium on Swarm Intelligence (SIS), 2011.

[7] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," in *EP '98: Proceedings of the 7th International Conference on Evolutionary Programming VII*. London, UK: Springer-Verlag, 1998, pp. 591–600.

[8] G. Cawley. Model selection for support vector machines via adaptive step-size tabu search, in Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms, 2001, pp. 434437.

[9] C. Igel and T. Suttorp Multi-objective optimization of support vector machines, in Yaochu Jin (Ed.), Multi-objective Machine Learning Studies in Computational Intelligence, Vol. 16, pp. 199-220, Springer-Verlag, 2006.

[10] S.S. Keerthi. Efficient tuning of svm hyperparameters using radius/margin bound and iterative algorithms. IEEE Transactions on Neural Networks, 2002.

[11] R. A. Santana, M. R. Pontes, and C. J. A. Bastos-Filho, "A multiple objective particle swarm optimization approach using crowding distance and roulette wheel," in *Proceedings of the 2009 Ninth International Conference on Intelligent Systems Design and Applications*, ser. ISDA '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 237–242.

[12] C. Tsou, S. Chang, and P. Lai, *Using Crowding Distance to Improve Multi-Objective PSO with Local Search - Swarm Intelligence, Focus on Ant and Particle Swarm Optimization*. IN-Tech Education and Publishing, 2007.

[13] C. Coello, G. Pulido, and M. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 256–279, 2004.

[14] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 131–159, 2002.

[15] B. de Souza, A. de Carvalho, and R. Ishii. Multiclass svm model selection using particle swarm optimization. *Sixth International Conference on Hybrid Intelligent Systems*, 2006.

[16] C. Igel. Multiobjective Model Selection for Support Vector Machines, in Proc. of the 3rd Int. Conf. on Evolutionary Muilt Criterion Optimzation, 2005, pp. 534-546.

[17] Y. Jin and B. Sendhoff, Pareto-Based Multi-Objective Machine Learning : An Overview and Case Studies, IEEE Transactions on Systems, Man and Cybernetics: Part C, vol. 38, no. 3, pp. 397-415, 2008.

[18] S. Ali and K. A. Smith-Miles. A meta-learning approach to automatic kernel selection for support vector machines. *Neurocomputing*, 173–186, 2006.

[19] T. Gomes, R. B. C. Prudencio, C. Soares, A. Rossi, A. Carvalho. Combining meta-learning and search techniques to svm parameter selection, in: Brazilian Symposium on Neural Networks, 2010, pp. 7984.

[20] C.-W. Hsu, C.-C. Chang, and C.-J. Lin. Scikit-Learn: A Toolkit of Machine Learning in Python, Available: http://scikit-learn.org.

[21] Frank, A. Asuncion, A. UCI Machine Learning Repository, Avaiable: http://archive.ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science.

[22] Kate A. Smith-Miles. Cross-Disciplinary Perspectives on Meta-Learning for Algorithm Selection, 2005.

[23] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation Journal*, vol. 8(2), pp. 125–148, 2000.