ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

Fine-Tuning of Support Vector Machine Parameters using Racing Algorithms

Péricles B.C. Miranda[1], Ricardo M. Silva[1] and Ricardo B. Prudêncio[1]

1- Centro de Informática - Universidade Federal de Pernambuco - Brazil

Abstract. This paper investigates the iterative racing approach, I/F-Race, for selecting parameters of SVMs. As a racing algorithm, I/F-Race eliminates candidate models as soon as there is sufficient statistical evidence of their inferiority relative to other models with respect to the objective. The results revealed that the I/F-Race algorithm was able to achieve better parameter values in comparison to default parameters used in literature and parameters suggested by particle swarm optimization techniques.

1 Introduction

SVMs have achieved a considerable attention due to its theoretical foundations and good empirical performance when compared to other learning algorithms [1]. However, the SVM performance strongly depends on the adequate choice of its parameters [1]. In order to avoid an exhaustive exploration of parameters, different authors have deployed search and optimization techniques [10, 11]. Although these approaches automatize the parameter selection process, they also have parameters to be adjusted to perform well. Another limitation is that the search process usually starts with random configurations from the search space. This can result on slow convergence and sensibility to local minima depending on the adopted search technique.

In this work, we investigated the application of a racing algorithm called I/F-Race [4], for the SVM parameter selection problem. This algorithm has been widely used for tuning meta heuristics [3, 7, 8] and its positive point is the simplicity to be designed. However, it has not been applied to select parameters of learning algorithms. Thus, we implemented the most recent version of I/F-Race algorithm [6] to optimize the parameters: γ of the RBF kernel and the regularization constant C, using the success rate (SR) on classification as objective function.

In our experiments, we compared the I/F-Race with Particle Swarm Optimization (PSO) variants and the default heuristic adopted by the LibSVM tool [9]. We chose the PSO algorithm due to good results observed in previous work for SVM parameter problem [11]. The results revealed that the I/F-Race algorithm converged faster, reaching better solutions than LibSVM default heuristic and the PSO variants.

This paper is organized as follows: Section 2 presents a formal definition of the SVM parameter selection. Section 3 brings the basic concepts and the I/F-Race algorithm. Section 4 describes the experiments and obtained results. Finally, Section 6 presents some conclusions and the future work.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

2 SVM Parameter Selection Problem

Here we briefly introduce a formal definition of the algorithm configuration problem, which can also be applied to SVM parameter selection [3]. Let us assume that we have a parametrized algorithm X_d with N^{param} parameters, where $d = 1, ..., N^{param}$, and each of them may take different configurations. A configuration of the algorithm $\theta = (x_1, ..., x_{N^{param}})$ is a unique assignment of values to parameters, and Θ denotes the possibly infinite set of all configurations of the algorithm. When considering a problem to be solved by this algorithm, the set of possible instances is defined as I. It is also given a fitness function $\zeta(\theta, i)$ that assigns a value to each configuration when applied to a single problem instance $i \in I$. The quality measure assigns a value to one run of a particular instance. The criterion that we want to optimize when configuring an algorithm for a problem is c_θ of the fitness of a configuration θ with respect to I. The goal of automatic configuration is finding the best configuration θ^*.

A usual definition of c_θ is the expected fitness of θ. The value of c_θ is achieved by obtaining realizations $c_{\theta,i}$. Considering the problem at hand, the $c_{\theta,i}$ is the SR of configuration $\theta = (\gamma, C)$ on a classification problem i. Hence, the goal is to find the configuration θ which maximizes $c_{\theta,i}$ on instance i.

3 Racing Algorithms

Model selection is an important task in the application of many machine learning methods. Racing algorithms were first applied to the model selection problem in memory-based supervised learning [2]. Racing algorithms can select in a fully automatic way a configuration for an algorithm from a given set of candidate configurations Θ. A racing algorithm works by sequentially processing a given set of instances I. Let i denote the l^{th} instance and let Θ_k be the set of candidate configurations at iteration k. Initially $\Theta_1 = Sample(X)$ (where X is the parameter space), and then, at iteration k of the race, all sampled candidate configurations in Θ_k are run once on instance i. When all results are available, the candidates in Θ_k that are shown to be statistically inferior are eliminated, resulting in a possibly smaller set Θ_{k+1}. This procedure is iterated until either only one candidate remains or a maximum limit on the overall computation time of the racing procedure expires.

The proposal of [2] inspired a diverse number of new algorithms which improved even more the search process. We highligth the I/F-Race was proposed by [4] and it is an iterative application of the F-Race algorithm [5]. In the I/F-Race, at each iteration a number of surviving candidate configurations of the previous iteration bias the sampling of new candidate configurations. It is hoped in this way to focus the sampling of candidate configurations around the most promising ones. Initially, a sample from the complete set of configurations is selected. After that, the best configurations are selected by a racing procedure (See F-Race in [5]). At each step of the F-Race, the candidate configurations are evaluated on a single instance. After each step, those candidate configurations

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

that perform statistically worse than at least another one are discarded, and the race continues with the remaining surviving configurations. Once F-Race terminates, the candidates within the Θ^{elite} are then weighted according to their ranks. We use N^{elite} which denotes the number of candidates that survived the race. The weight of an elite configuration with rank $r_z(z = 1, ..., N^{elite})$ is given by:

$$w_z = \frac{N^{elite} - r_z + 1}{N^{elite}(N^{elite} + 1)/2}. \quad (1)$$

In next iteration, the $|\Theta_{k+1}| - N^{elite}$ new candidate configurations are iteratively sampled around one of the elite configuration. To do so, for sampling each new candidate configuration, first one elite solution $E_z(z \in 1, ..., N^{elite})$ is chosen with a probability proportional to its weight w_z and next a value is sampled for each parameter. This procedure continues until reaching a stopping criterion.

In this work we implemented the most recent version of I/F-Race developed by López-Íbañez et al. in 2011 and recently revised in January 2013 [6]. In the following, we present the experiments conducted in different configurations to compare the effect of the racing procedures described before.

4 Experiments

In this section, we present the experiments which evaluated the I/F-Race for the SVM parameter selection problem. As a basis of comparison, we implemented four variants of the PSO algorithm, considering the inertia and constricted weights, and the two most known topologies in literature, Star and Ring [11]. We also considered, as a baseline, the default heuristic adopted by the LibSVM tool (γ= inverse of the number of attributes and $C = 1$) [9]. In order to guarantee a fair simulation, the I/F-Race and the PSOs were executed 30 times and the average results were recorded. All algorithms were evaluated on the set of 35 different numerical classification datasets available in the *UCI* repository. The list of classification problems used in this work is presented in Table 1. These problems correspond to databases associated with different application domains.

Table 1: Classification Problems

Bal. Scale	Blood	Letter	Column-3c	Yeast
Hrt.-Statlog	Colic	Brst.-w	Column-2c	Sonar
Prim. Tumor	Ecoli	Glass	Haberman	Vehicle
Pr. Diabetes	Hepatitis	Cancer	Parkinson	Vote
Hypothyroid	Br. Tissue	Iris	Kr-vs-kp	Heart
Ionosphere	Libras	Lymph	L. Cancer	Colon
Mamography	Optdigits	H. Valley	Pen Digits	Wine

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

The algorithms performed a search in a space represented by a discrete grid of SVM configurations. By following the guidelines provided in [9], γ assumed 38 different values (from 2^{-15} to 2^{3}) and the parameter C assumed 42 different values (from 2^{-5} to 2^{15}). The difference between the consecutive values in this sequence is $0,25$. Thus yielding $38 \times 42 = 1596$ different combinations of parameters. The objective function considered was the SR on classification, the most direct way to evaluate the performance of the SVM model [1]. The values of the objective function were obtained through the SVM execution in the *10-fold* cross-validation experiment. Considering that, the search aims to find configurations (γ, C) which maximizes the objective function.

The I/F-Race was executed using 10 iterations and the F-Race 5 iterations, using 30 configuration candidates from Θ. The PSOs were executed using 10 iterations, 45 individuals, inertia factor equals to 0.72, learning coefficients $c_1 = c_2 = 1.49$ and maximum velocity equals to 4, as defined in [11]. Although the number of individuals is different, both algorithms performed a total of 450 fitness evaluations. Besides, all the algorithms were set with an uniform random sampling.

In this experiment, we evaluated the results using two quality measures: *Measure 1*, the mean of SR values of all problems for each iteration and *Measure 2*, the number of wins of the algorithms per iteration regarding all problems, where the algorithm which achieved better quality (according to the SR) is the winner.

5 Results

In order to analyze 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 I/F-Race algorithm is a competitive approach for the problem at hand. All the following analysis used this methodology.

Figure 1 shows the average of SR values of each algorithm for all problems per iteration (*Measure 1*). As it can be seen, the I/F-Race (iRace) generates better results than all PSO variants and the LibSVM baseline. In the first iteration, the SR values of all algorithms are statistically similar because all algorithms started using configurations from a random sample. However, from the second iteration to the end we can see how fast the I/F-Race convergence is. This can be explained due to the inherent characteristic of racing algorithms which discard configurations that are statistically worse than the best configurations. According to the Wilcoxon test, the I/F-Race distribution is better than all PSO variants distribution with 95% of confidence from the third to the last iteration.

In order to perform a deeper analysis of the results, we also measured the performance of the algorithms in each problem individually (*Measure 2*). This measure counts the number of classification problems that each algorithm won for each iteration. To define a win we compare the SR achieved by the algorithm a_1 with the SR achieved by the algorithm a_2. If the SR value of a_2 is statistically better than the SR value of a_1, a win is assigned for a_2. As the classification

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

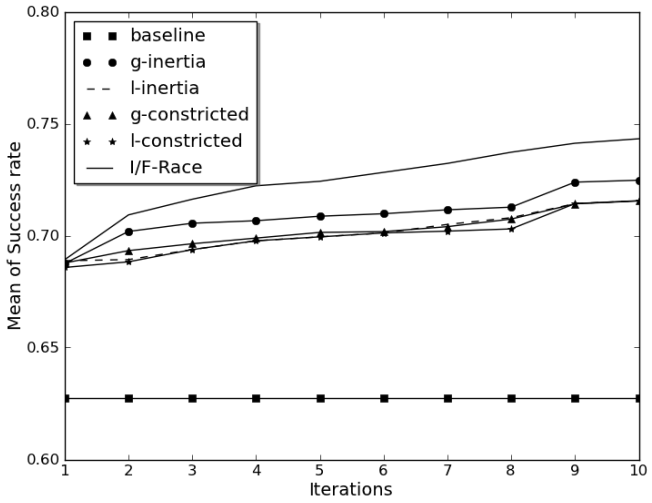

Fig. 1: Mean of SR x Iteration considering 35 classification problems.

problems used in this work are associated to different domain applications, this measure makes the analysis independent of the problem nature.

We performed the experiment considering the *Measure 2* by comparing the I/F-Race with the *g-inertia*, the PSO variant that achieved the best results in the previous experiment. Figure 2 presents the number of classification problems the I/F-Race and *g-inertia* won in each iteration. As it can be seen, the *g-inertia* and the I/F-Race won, in average, $6,2$ and $19,9$ classification problems per iteration. In the initial iterations, the number of ties is large due to the random initialization. However, since the second iteration to the end, the I/F-Race increases the number of wins, reaching 25 classification problems in the end of the simulation whereas the *g-inertia* reaches 7 wins.

6 Conclusion

In this current work, we applied the I/F-Race algorithm to select the parameter γ of the RBF kernel and the regularization parameter C of SVMs. The results presented that the I/F-Race converged faster, generating better configurations, in most classification problems, when compared to the PSO variants and the default heuristic of the LIBSVM. In future work, we intend to optimize the SVM considering more parameters and objective functions. Finally, we intend to evaluate the proposed solution in other case studies, such as in the SVM parameter selection for regression problems.

Acknowledgments: The authors would like to thank CNPq, CAPES, FAPEMIG and FADE for their financial support.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

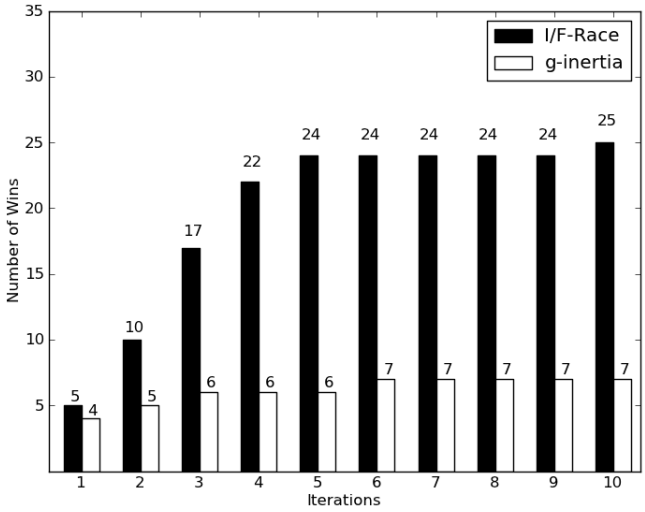

Fig. 2: Number of wins of I/F-Race and *g-inertia* considering 35 classification problems.

References

[1] N. Cristianini and J. Shawe-Taylor. An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods. Cambridge University Press, 2000.

[2] O. Maron and A. W. Moore. Hoeffding races: Accelerating model selection search for classification and function approximation. In Adv. in Neural Inform. Proc. Systems, vol. 6, p 59-66, USA, 1994.

[3] M. Birattari. Tuning Metaheuristics: A Machine Learning Perspective. In Studies in Comput. Intelligence. Springer-Verlag, vol. 197.

[4] P. Balaprakash, M. Birattari and T. Stutzle. Improvement Strategies for the F-Race Algorithm: Sampling Design and Iterative Refinement. In Hybrid Metaheuristics, vol. 4771 of LNCS, p. 108-122, 2007.

[5] M. Birattari, T. Stutzle, L. Paquete and K. Varrentrapp. A Racing Algorithm for Configuring Metaheuristics. In GECCO 2002, p. 11-18.

[6] M. López-Ibáñez, J. Dubois-Lacoste, T. Stützle and M. Birattari. The irace package, Iterated Race for Automatic Algorithm Configuration. In Technical Report TR/IRIDIA/2011-004.

[7] J. Dubois-Lacoste, M. López-Ibãnez, T. Stutzle. A Hybrid TP+PLS Algorithm for Bi-objective Flow-Shop Scheduling Problems. In Computers Operations Research, vol. 38(8), p. 1219-1236, 2011a.

[8] M. López-Ibãnez and T. Stutzle. Automatic Configuration of Multi-Objective ACO Algorithms. In Int. Conf. on Swarm Intelligence, vol. 6234 of LNCS, p. 95-106.

[9] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines. In ACM Transac. on Intel. Systems and Technology (TIST), vol. 2(3), Article No. 27, 2011.

[10] S. Lessmann, R. Stahlbock and S. Crone. Genetic algorithms for support vector machine model selection. In Int. Joint Conf. on Neural Networks, p. 3063–3069, 2006.

[11] M. Hric, M. Chmulík and R. Jarina. Model Parameters Selection for SVM Classification using Particle Swarm Optimization. In 21st Int. Conf. Radioelektronika, p. 1–4, 2011.