Hyper-parameter Tuning of a Decision Tree Induction Algorithm

Rafael G. Mantovani *, Tomáš Horváth* †, Ricardo Cerri ‡, Joaquin Vanschoren §, André C. P. L. F. de Carvalho*
* Institute of Mathematical and Computer Science, University of São Paulo, São Carlos, SP, Brazil
† Institute of Computer Science, Pavol Jozef Šafárik University in Košice, Slovakia
‡ Department of Computer Science, Federal University of São Carlos, São Carlos, SP, Brazil
§ Eindhoven University of Technology, Netherlands
E-mail: rgmantov@icmc.usp.br, tomas.horvath@upjs.sk, cerri@dc.ufscar.br, j.vanschoren@tue.nl, andre@icmc.usp.br

***Abstract*—Supervised classification is the most studied task in Machine Learning. Among the many algorithms used in such task, Decision Tree algorithms are a popular choice, since they are robust and efficient to construct. Moreover, they have the advantage of producing comprehensible models and satisfactory accuracy levels in several application domains. Like most of the Machine Leaning methods, these algorithms have some hyper-parameters whose values directly affect the performance of the induced models. Due to the high number of possibilities for these hyper-parameter values, several studies use optimization techniques to find a good set of solutions in order to produce classifiers with good predictive performance. This study investigates how sensitive decision trees are to a hyper-parameter optimization process. Four different tuning techniques were explored to adjust J48 Decision Tree algorithm hyper-parameters. In total, experiments using 102 heterogeneous datasets analyzed the tuning effect on the induced models. The experimental results show that even presenting a low average improvement over all datasets, in most of the cases the improvement is statistically significant.**

I. Introduction

Supervised classification is one of the main Machine Learning (ML) tasks, and as a consequence, there is a large variety of classification algorithms available. Among them, Decision Tree (DT) induction algorithms have been popularly used [1]. As classifiers, DTs are represented by rules structured as a tree, being widely used especially due to its comprehensible nature which resembles the human reasoning [2]. Some authors stated that DTs also figure among the most used data mining algorithms by researchers and practitioners, which reinforces its importance in the ML area [3], [4].

DT induction algorithms have several advantages over many other ML algorithms, such as robustness to noise (missing values, imbalanced classes), low computational cost, and the ability to deal with redundant attributes [2]. There are many well-known DT induction algorithms in literature, such as Quinlan's C4.5 algorithm [5] and Breiman et al.'s Classification and Regression Tree (CART) [6].

The values chosen for the hyper-parameters (HPs) of ML algorithm directly affect the predictive performance of the models induced by them. Thus, a good choice of these values has been the subject of study in ML for years. These studies have been run to understand the HP effect of different algorithm, using techniques from the simplest ones, such as Grid Search (GS) or Random Search (RS) [7], to the more complex, such as meta-heuristics (MTH) [8] and meta-learning (MtL) [9]. Although many techniques have been proposed for Support Vector Machines (SVMs) [10], [11] and Neural Networks (NNs) [12], few studies have been conducted for HP optimization of DT induction algorithms [13]–[15].

This study investigate how sensitive DT induction algorithms are to a HP tuning process, specially the J48 algorithm, a WEKA [16] implementation for the Quinlan's C4.5 DT induction algorithm [5]. Experiments were carried out with a large number of heterogeneous datasets, and four different tuning techniques: RS, Genetic Algorithm (GA) [17], Particle Swarm Optimization (PSO) [18], and an Estimation of Distribution Algorithms (EDA) [19]. The former three techniques are commonly used MTHs for HP tuning. The results obtained in terms of the predictive accuracy when using these four techniques are compared with the results obtained by the J48 induced by its *default* HPs values (DF).

This paper is structured as follows: section II introduces the HP tuning problems and some related work; section III describes the experimental methodology and the evaluation of the tuning techniques; the results are discussed in section IV; finally, the conclusions and future research directions are presented.

II. Hyper-parameter tuning

HP tuning can largely affect the predictive performance of ML algorithms [9]. Setting a suitable configuration for the HPs of a ML algorithm is usually performed by trial and error. Depending on the training time of the ML algorithm used, finding a good set of values manually can be very time-consuming. As a result, recent works in HP for ML algorithms focus on the development of better HP tuning techniques [12], [20].

The HP process is usually treated as an optimization (black-box) problem, whose objective function is associated with the predictive performance of the model induced by the algorithm. More formally:

Let $\mathcal{H} = \mathcal{H}_1 \times \mathcal{H}_2 \times \cdots \times \mathcal{H}_k$ be the HP-space for the algorithm $a \in \mathcal{A}$ where $\mathcal{A}$ is the set of ML algorithms. Each

978-1-5090-3566-3/16 $31.00 © 2016 IEEE
DOI 10.1109/BRACIS.2016.62

$\mathcal{H}_i$ represents a set of admissible values for the ith HP of a ($i \in \{1, \ldots, n\}$) and can be usually defined by some constraints. Let $\mathcal{D}$ be a set of datasets where $\mathbf{D} \in \mathcal{D}$ is a dataset from $\mathcal{D}$. The function $f : \mathcal{A} \times \mathcal{D} \times \mathcal{H} \rightarrow \mathbb{R}$ measures the predictive performance of the algorithm $a \in \mathcal{A}$ on the dataset $\mathbf{D} \in \mathcal{D}$ given a HP configuration $\mathbf{h} = (h_1, h_2, \ldots, h_k)$. Without loss of generality, higher values of f mean higher predictive performance.

The task of HP tuning is, given $a \in \mathcal{A}$, $\mathcal{H}$ and $\mathbf{D} \in \mathcal{D}$, to find $\mathbf{h}^\star = (h_1^\star, h_2^\star, \ldots, h_k^\star)$ such that

$$\mathbf{h}^\star = \underset{\mathbf{h} \in \mathcal{H}}{arg\,max}\ f(a, \mathbf{D}, \mathbf{h}) \tag{1}$$

The optimization can be carried out based on any performance measure f, which can even be defined by multi-objective criteria. There are some aspects that can make the HP tuning more difficult:

- HP configurations that lead to a model with high predictive performance for a given dataset may not lead to high predictive performance for other datasets;
- HP values often depend on each other (as in the case of SVMs [21]). Hence, optimizing HPs independently is not a reasonable strategy;
- the evaluation of a specific HP configuration, let alone many, can be very time consuming.

A. Recent Approaches

Many techniques have been proposed for HP tuning of classification algorithms [12], [20]. Some studies use Grid Search (GS) [7], a simple deterministic approach which reduces each HP-space dimension $\mathcal{H}_i$ to a finite set of values $\mathcal{H}_i^r = \mathcal{P} = \mathcal{H}_1^r \times \mathcal{H}_2^r \times \cdots \times \mathcal{H}_k^r$ that are strictly evaluated. GS obtained good results in low dimensional problems. For optimization of many HPs in large datasets, GS becomes computationally expensive. For these scenarios, some studies have explored Random Search (RS) techniques [22].

RS starts with a simple HP configuration in $\mathcal{P}$, which is extended by a randomly generated HP configuration at each iteration. Usually, the process stops after a given number of iterations. RS has obtained efficient results in the optimization of Deep Learning (DL) algorithms [20], [23].

Bio-inspired approaches, such as GA or PSO have also been largely used for HP optimization [8], [24], [25]. In these techniques, an initial population $\mathcal{P}$ is continuously updated according to various stochastic strategies imitating evolutionary processes and behaviors of swarms, respectively.

Generally, population based techniques follow a generic iterative process described in Algorithm 1, based on updating a population of initial solutions according to a given strategy **until** some stopping criteria are satisfied.

Sequential Model-based Optimization (SMBO) [26] has also emerged as a successful HP tuning technique in ML. In SMBO, $\mathcal{P}$ is extended by a new HP configuration $\mathbf{h}'$ at each iteration, such that the expected value of $f(a, \mathbf{D}, \mathbf{h}')$ is maximal according to an induced meta-model $\hat{f}$ approximating f on the current population. In the experiments reported in [12], [27], [28], SMBO performed better than GS and RS and matched or outperformed state-of-the-art techniques in several HP optimization tasks.

Algorithm 1 Generic population-based HP tuning process

procedure TUNEHP($a \in \mathcal{A}, \mathbf{D} \in \mathcal{D}, \mathcal{H}, f, strategy$)
  $\mathcal{P} \leftarrow \{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n\}$ ▷ initial population
  $\mathcal{F} \leftarrow \{f(a, \mathbf{D}, \mathbf{h}_i) \mid \mathbf{h}_i \in \mathcal{P}\}$ ▷ population fitness
  repeat
    $\mathcal{P} \leftarrow$ UPDATE($\mathcal{P}, \mathcal{F}, strategy$)
    $\mathcal{F} \leftarrow \{f(a, \mathbf{D}, \mathbf{h}_i) \mid \mathbf{h}_i \in \mathcal{F}\}$
  until stopping criteria not satisfied
return $\mathbf{h}^\star \leftarrow \underset{\mathbf{h} \in \mathcal{P}}{arg\,max}\ f(a, \mathbf{D}, \mathbf{h}) \in \mathcal{F}$

Several automated tools for HP optimization of ML algorithms are also available in the literature, such as techniques based on local search (ParamILS [29]), estimation of distributions (REVAC [30]) and Bayesian optimization (Auto-Weka [31] and Auto-skLearn [32]).

B. Related works

Few studies have investigated the HP tuning of DT induction algorithms. In [13], the authors investigated the prediction of the training time for several time target classifiers, including DT, using MtL. In the process, five numeric HPs of the CART DT induction algorithm were optimized using GS. Similar works can be found: [14] tuned two HPs of the J48 algorithm in a case study with educational datasets; and [15] implemented an open-source MtL system to predict accuracies of target classifiers, one of them is a DT induction algorithm (a version of the C5.0), which has its confidence factor (C) adjusted using GS.

A special case of tuning is done by the "Combined Algorithm Selection and HP optimization" (CASH) tools. They were introduced by [31] as the Auto-WEKA framework, and further studied as the Auto-sklearn tool [32]. Both apply SMBO to select algorithms and their configurations to new problems. Auto-WEKA encapsulates the J48, while the Auto-sklearn a CART DT.

III. Materials and Methods

In the experiments, four different techniques for HP optimization were investigated, using 102 datasets: a simple Random Search strategy (RS), equivalent to a GS process (as suggested by [23]); and three different meta-heuristics (MTHs) - Genetic Algorithms (GAs), Particle Swarm Optimization (PSO), and Estimation of Distribution Algorithm (EDA). The average per class accuracy was used to assess the predictive performance of the induced DT models and guide the search performed by the optimization techniques.

A. J48's Hyper-parameter space

The experiments optimized the HP of the J48 algorithm, a WEKA implementation for Quinlan's C4.5 algorithm. This algorithm was chosen due to its wide acceptance and use in ML [2]. The J48 HP-space is shown in Table I. The

TABLE I
J48 HYPER-PARAMETER SPACE EXPLORED IN EXPERIMENTS.

Symbol	Hyper-parameter	Range	Type	Default	Requires
C	pruning confidence	$\{0.001, 0.5\}$	real	0.25	R = False
M	minimum number of instances	$\{1, 50\}$	integer	2	-
N	number of folds for reduced error pruning	$\{2, 10\}$	integer	3	R = True
O	do not collapse the tree	{False,True}	boolean	False	-
R	use reduced error pruning	{False,True}	boolean	False	-
B	use binary splits only	{False,True}	boolean	False	-
S	do not perform subtree raising	{False,True}	boolean	False	-
A	Laplace smoothing for predicted probabilities	{False,True}	boolean	False	-
J	do not use MDL correction for info gain on numeric attributes	{False,True}	boolean	False	-

experiments focused only on pruned trees, since they looked for the most interpretable models without loss of predictive performance. For each HP, the table shows its allowed range of values, default HP values obtained from WEKA, and its constraints for setting new values. The range for the pruning confidence (C) HP and the M values was the same used in [15] and [13], respectively. During the optimization process, each HP setting is coded as a numeric array with nine elements.

B. Experimental methodology

A few experimental methodologies to repeatedly select and assess classification models can be found in the literature [33]. When a combined optimization and evaluation is required, a nested cross-validation (N-CV) methodology is usually recommended to assess the performance of models. The N-CV is used in a theoretical scenario, and may be not practical in real tasks, especially HP-tuning, due to the computational costs.

Thus, for the experiments in this study, a modified 10-fold N-CV method was applied: an outer loop iterates over 10 folds, and a 8-1 holdout split is used in the inner loop to evaluate the *fitness* of each candidate HP configuration and guide the search of the optimization techniques.

At each CV iteration, test accuracies are assessed using the model induced with the training partitions considering the HP values found by the optimization technique. It is important to mention that the test accuracy values were not used in the model selection process, only to assess the predictive performance of the selected models.

C. Datasets

The experiments were carried out using 102 datasets from the UCI ML repository[1] [34]. Table II summarizes the main aspects of these datasets: number of examples (N), number of attributes (D) and number of classes (C).

D. Tuning techniques

The GA, PSO and EDA meta-heuristics used for HP tuning are implemented using the `GA`, `pso`, and `copulaedas` R packages, respectively. The experiments used the same GA and PSO parameter values suggested in [35]: an uniform random mutation rate of 0.05; a tournament selection with size k=3; and a local arithmetic crossover methodology. For the EDA, the Gaussian Copula EDA (GCEDA) with default parameter values provided by its R package was used. The RS technique was implemented by the authors.

Besides, the `RWeka` implementation of J48 DTs as a baseline algorithm was used in the experiments. The baseline is the HP default values provided by the R package (DFs). To perform well, EDAs need at least 100 individuals in the population [19]. Thus, the size of the initial population for all optimization techniques was set to 100. The maximum number of iterations was empirically defined as $it = 50$. This leads to at most $50 \times 100 = 5000$ evaluations of HP values (individuals) during the search. Since the techniques are stochastic, each one was executed 30 times for each dataset.

IV. EXPERIMENTS

Figure 1 summarizes the experimental results: the first chart shows the average accuracy improvement by the tuning techniques in each dataset ('*improvement*'); second presents the average accuracy obtained by the best tuned solution compared with the DF values per dataset; and the last one depicts the Wilcoxon statistical significance comparison between each technique and the DF values.

A. Average Improvement

The top chart in Fig. 1 shows the average improvement obtained by GA, PSO, EDA and RS, when compared with the DF average accuracies (scaled between 0 and 1). There is a small difference in the gains of each tuning technique regarding the DF. The predictive performance obtained by the tuning techniques are very similar, and their curves almost overlap.

The Friedman test [36] with significance level at $\alpha = 0.05$ was used to assess the statistical significance of the experimental results. The null hypothesis states that the classifiers induced with the solutions found by the tuning techniques are equivalent in terms of the averaged accuracy per class. If the null hypothesis was rejected, the Nemenyi post-hoc test was applied, stating that the performance of two different techniques is significantly different if the corresponding average ranks differ by at least a Critical Difference (CD) value.

Figure 2 presents the CD diagram for the statistical tests. Techniques are connected when there is *no* statistically significant difference between them (at $\alpha = 0.05$ and CD

[1] http://archive.ics.uci.edu/ml/

TABLE II
(MULTI-CLASS) CLASSIFICATION UCI DATASETS USED IN THE EXPERIMENTS. IT IS PRESENTED, FOR EACH DATASET, THE NUMBER OF EXAMPLES (N), NUMBER OF ATTRIBUTES (D) AND THE NUMBER OF CLASSES (C).

No.	Name	N	D	C	No.	Name	N	D	C
•**1**	abalone-11class	3842	8	11	•**52**	habermans-survival	289	3	2
•**2**	abalone-28class	4177	8	28	•**53**	hayes-roth	93	4	3
•**3**	abalone-3class	4177	8	3	•**54**	heart-disease-processed-cleveland	303	13	5
•**4**	abalone-7class	3295	8	7	•**55**	heart-disease-processed-hungarian	293	13	2
5	acute-inflammations-nephritis	99	6	2	•**56**	heart-disease-processed-switzerland	123	12	5
6	analcatdata_authorship	841	70	4	•**57**	heart-disease-processed-va	199	13	5
•**7**	analcatdata_boxing1	120	3	2	•**58**	heart-disease-reprocessed-hungarian	293	13	5
•**8**	analcatdata_boxing2	132	3	2	•**59**	hepatitis	155	19	2
9	analcatdata_creditscore	100	6	2	•**60**	horse-colic-surgical	300	27	2
•**10**	analcatdata_dmft	556	4	6	•**61**	indian-liver-patient	570	10	2
•**11**	analcatdata_germangss	400	5	4	62	ionosphere	350	33	2
12	analcatdata_lawsuit	263	4	2	63	iris	147	4	3
13	appendicitis	106	7	2	•**64**	kr-vs-kp	3196	36	2
14	artificial-characters	4891	7	10	65	leaf	340	15	30
•**15**	autoUniv-au1-1000	997	20	2	•**66**	led7digit	146	7	10
•**16**	autoUniv-au4-2500	2500	100	3	67	leukemia-haslinger	100	50	2
•**17**	autoUniv-au6-1000	1000	40	8	68	lsvt-voice-rehabilitation	126	307	2
•**18**	autoUniv-au6-250-drift-au6-cd1-500	750	40	8	•**69**	lymphography	148	18	4
•**19**	autoUniv-au6-cd1-400	400	40	8	•**70**	mammographic-mass	689	5	2
•**20**	autoUniv-au7-300-drift-au7-cpd1-800	1100	12	5	•**71**	meta-data	528	21	24
•**21**	autoUniv-au7-700	700	12	3	•**72**	mfeat-fourier	1994	76	10
•**22**	autoUniv-au7-cpd1-500	500	12	5	73	molecular-promotor-gene	106	57	2
•**23**	backache	180	31	2	•**74**	monks1	432	6	2
•**24**	balance-scale	625	4	3	•**75**	monks2	432	6	2
•**25**	banana	5292	2	2	76	monks3	438	6	2
•**26**	bank-marketing	4521	16	2	77	movement-libras	330	90	15
•**27**	banknote-authentication	1348	4	2	78	mushroom	8124	21	2
•**28**	blood-transfusion-service	533	4	2	•**79**	optdigits	5620	62	10
29	breast-cancer-wisconsin	463	9	2	•**80**	ozone-eighthr	2526	72	2
30	breast-tissue-4class	105	9	4	•**81**	ozone-onehr	2528	72	2
•**31**	breast-tissue-6class	105	9	6	•**82**	page-blocks	5406	10	5
32	bupa	341	6	2	83	parkinsons	195	22	2
•**33**	car-evaluation	1728	6	4	•**84**	phoneme	5395	5	2
•**34**	cardiotocography-3class	2116	35	3	85	robot-nav-sensor-readings-2	5177	2	4
•**35**	climate-simulation-craches	540	20	2	86	robot-nav-sensor-readings-4	5406	4	4
•**36**	cmc	1425	9	3	•**87**	saheart	462	9	2
•**37**	cloud	108	6	4	•**88**	seeds	210	7	3
38	collins	500	21	15	•**89**	statlog-german-credit-numeric	1000	24	2
39	connectionist-mines-vs-rocks	208	60	2	•**90**	statlog-german-credit	1000	20	2
•**40**	connectionist-vowel-reduced	528	10	11	•**91**	statlog-heart	270	13	2
•**41**	connectionist-vowel	990	13	11	•**92**	statlog-image-segmentation	2086	18	7
•**42**	dermatology	366	34	6	•**93**	statlog-landsat-satellite	2859	36	6
43	ecoli	336	7	8	94	statlog-vehicle-silhouettes	846	18	4
•**44**	energy-efficiency-y1	768	8	37	95	steel-plates-faults	1941	33	2
•**45**	energy-efficiency-y2	768	8	38	96	teaching-assistant-evaluation	110	5	3
•**46**	fertility-diagnosis	100	9	2	•**97**	texture	5473	40	11
•**47**	first-order-theorem	5636	51	6	98	thoracic-surgery	470	16	2
•**48**	flags-colour	194	28	8	•**99**	thyroid-allbp	2751	26	5
49	flags-religion	194	28	8	•**100**	thyroid-allhyper	2751	26	5
•**50**	flare	527	12	6	101	thyroid-newthyroid	215	5	3
51	glass	213	9	6	102	vertebra-column-3c	310	6	3

$= 0.604$). According to the test, EDA had the best average ranking. With the exception of the PSO technique, it was statistically superior to all techniques. When using the DF values, the predictive performance was statistically inferior to the predictive performance of the algorithms optimized by all tuning techniques.

B. Best Solutions

Based on the Friedman average ranking, EDA was selected to represent the 'Best' tuning technique. Figure 1 (middle chart) compares its average performance (yellow line) with the DF values (green line). The datasets were sorted according to their accuracy values reached during the tuning process (highest to lowest). We can note that the best induced solutions are at least equivalent to the DF ones. There are several datasets where the improvements were small if compared to the use of DFs. This may be due to the nature of the classification problem, where the use of DF HP values had already led to almost a maximum performance, or the DFs were chosen as the best overall values on the UCI datasets.

The Wilcoxon paired-test was also applied to assess the statistical significance of the 'Best' and DF results. It was applied to the solutions obtained from each of the 30 executions for each dataset (at $\alpha = 0.05$). The bottom chart at Fig. 1 identifies the datasets where statistically significant improvements were detected when using each one of the techniques. White squares show datasets where DF was the best choice. Black and gray cells show datasets where a technique was better than DF with and without statistical significance, respectively.

For most of the datasets (70 of 102), the test showed statisti-

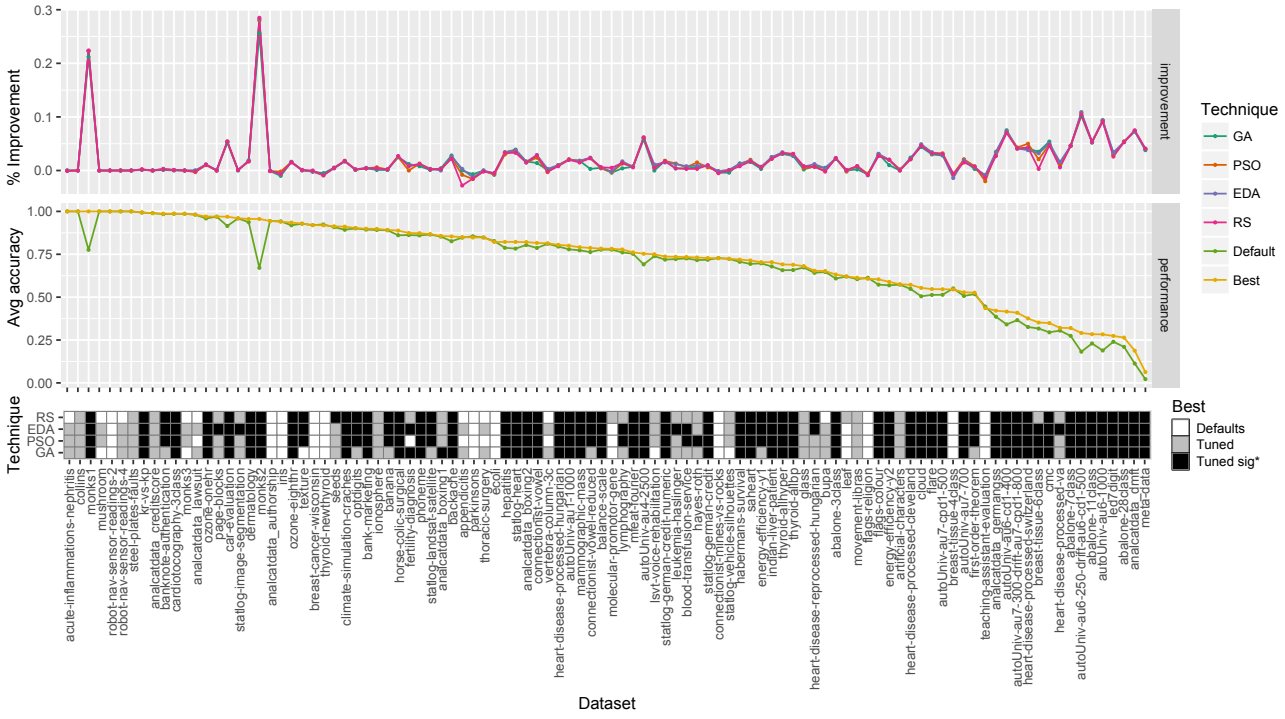

Fig. 1. Average performance improvement obtained by the tuning techniques (top), maximum performance obtained by the DT models generated with best solutions compared with DF values (middle), and Wilcoxon paired-test results between each technique and DF over 30 executions.

cally significant differences, even when the improvement was very small (the mean average improvement overall datasets was around 0.022). These datasets were also highlighted with a bullet (•) in Table II.

Trying to identify the situations where the J48 algorithm should be tuned, some measures to characterize the datasets were extracted from each dataset [37]. These measures were used by a DT induction algorithm, with default hyper-parameter values, to extract a tree able to explain when to tune and when not to tune the learning algorithm.

Two measures were selected by the model: the *'nb'* - the performance of a Naive Bayes (NB) classifier; and *'f2'* - a data complexity measure that describes the overlap of the per-class bounding boxes. They show that default values are good for a NB classifier performs good in datasets with a small number of classes ($nb \geq 90\% \wedge C \leq 4$), or in datasets with a low overlapping ($f2 \leq 0.191$) between the classes.

Using a Random Forest (RF) instead a DT to generate an interpretable model, it was noted that both measures, *'nb'* and *'f2'*, are among the five most important ones. Thus, the use of DF values seems to be adequate in simple classification problems. Usually the DF values are good, but still there many cases where it is better to tune the HPs. However, further analysis should be done to clarify specific cases where the tuning is required.

Regarding the improvements obtained by the HP tuning, DT feature selection embedded process seems to contribute more to the final model than HP tuning. This can be due to the small number of candidate solutions evaluated by the tuning techniques during the optimization: 5000 evaluations may be not sufficient to perform a good search. Thus, this value should to be increased in future experiments.

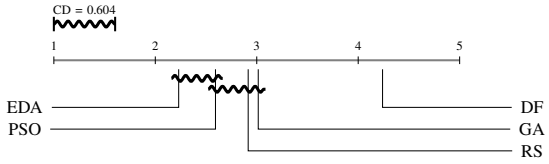

Fig. 2. Comparison of the predictive accuracy values of the tuning techniques according to the Nemenyi test. Groups of technique that are not significantly different (at $\alpha = 0.05$) are connected.

V. Conclusions

This work investigated the sensitivity of the J48 DTs induction algorithm to a HP tuning process. Experiments were carried out with 102 datasets using meta-heuristics and a RS technique. Tuned models were compared with DTs generated with DF HP values (provided by RWeka). Results showed similar predictive performances between the tuning techniques, but most of them performed better than the DF values with statistical significance.

In general, the DF values are good, but analyzing the results per dataset, even if most of them presenting a small improvement, in 70 of the 102 datasets the differences in the predictive performance were statistically significant. According to the

results, the DF values can be adequate to simple classification tasks, but even for some of these cases, HP optimization is better.

Future works include to expand the experiments adding more datasets, and increase the budget size of the tuning techniques. The authors also plan to include SMBO and IRACE techniques in the experiments. Finally the authors want to use OpenML [38] to make available all results and implementations, allowing reproducibility and further studies.

ACKNOWLEDGMENT

The authors would like to thank CAPES and CNPq (Brazilian Agencies) for their financial support, specially to the grants #2012/23114-9 #2013/07375-0 and #2015/03986-0 from São Paulo Research Foundation (FAPESP), and the Slovakian project VEGA 1/0475/14.

REFERENCES

[1] T. M. Mitchell, *Machine Learning*. New York: McGraw Hill, 1997.

[2] R. Barros, M. Basgalupp, A. de Carvalho, and A. Freitas, "A survey of evolutionary algorithms for decision-tree induction," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 42, no. 3, pp. 291–312, May 2012.

[3] O. Maimon and L. Rokach, *Data Mining and Knowledge Discovery Handbook*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[4] D. Jankowski and K. Jackowski, "Evolutionary algorithm for decision tree induction," in *Computer Information Systems and Industrial Management*, ser. Lecture Notes in Computer Science, K. Saeed and V. Snel, Eds. Springer Berlin Heidelberg, 2014, vol. 8838, pp. 23–32.

[5] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[6] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. Chapman & Hall (Wadsworth, Inc.), 1984.

[7] I. Braga, L. P. do Carmo, C. C. Benatti, and M. C. Monard, "A note on parameter selection for support vector machines," in *Advances in Soft Computing and Its Applications*, ser. LNCC, F. Castro, A. Gelbukh, and M. González, Eds. Springer Berlin Heidelberg, 2013, vol. 8266, pp. 233–244.

[8] F. Friedrichs and C. Igel, "Evolutionary tuning of multiple svm parameters," *Neurocomput.*, vol. 64, pp. 107–117, 2005.

[9] M. Feurer, T. Springenberg, and F. Hutter, "Initializing bayesian hyperparameter optimization via meta-learning," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, Jan. 2015.

[10] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1-3, pp. 131–159, Mar. 2002.

[11] S. Ali and K. A. Smith-Miles, "A meta-learning approach to automatic kernel selection for support vector machines," *Neurocomp.*, vol. 70, no. 13, pp. 173–186, 2006.

[12] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kélg, "Algorithms for hyper-parameter optimization," in *Proc. Neural Information Processing Systems 24 (NIPS2011)*, 2011, pp. 2546–2554.

[13] M. Reif, F. Shafait, and A. Dengel, "Prediction of classifier training time including parameter optimization," in *KI 2011: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, J. Bach and S. Edelkamp, Eds. Springer Berlin Heidelberg, 2011, vol. 7006, pp. 260 – 271.

[14] M. M. Molina, J. M. Luna, C. Romero, and S. Ventura, "Meta-learning approach for automatic parameter tuning: A case study with educational datasets," in *Proceedings of the 5th International Conference on Educational Data Mining, EDM 2012*, 2012, pp. 180–183.

[15] M. Reif, F. Shafait, M. Goldstein, T. Breuel, and A. Dengel, "Automatic classifier selection for non-experts," *Pattern Analysis and Applications*, vol. 17, no. 1, pp. 83–96, 2014.

[16] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. San Francisco: Morgan Kaufmann, 2005.

[17] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.

[18] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, Perth, Australia, 1995, pp. 1942 – 1948.

[19] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 111 – 128, 2011.

[20] R. Bardenet, M. Brendel, B. Kégl, and M. Sebag, "Collaborative hyperparameter tuning," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, S. Dasgupta and D. Mcallester, Eds., vol. 28, no. 2. JMLR Workshop and Conference Proceedings, 2013, pp. 199–207.

[21] A. Ben-Hur and J. Weston, "A user's guide to support vector machines," in *Data Mining Techniques for the Life Sciences*, ser. Methods in Molecular Biology. Humana Press, 2010, vol. 609, pp. 223–239.

[22] S. Andradottir, "A review of random search methods," in *Handbook of Simulation Optimization*, ser. International Series in Operations Research & Management Science, M. C. Fu, Ed. Springer New York, 2015, vol. 216, pp. 277 – 292.

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

[24] A. Kalos, "Automated neural network structure determination via discrete particle swarm optimization (for non-linear time series models)," in *Proceedings of the 5th WSEAS International Conference on Simulation, Modelling and Optimization*, ser. SMO'05. World Scientific and Engineering Academy and Society (WSEAS), 2005, pp. 325–331.

[25] T. A. F. Gomes, R. B. C. Prudêncio, C. Soares, A. L. D. Rossi, and nd André C. P. L. F. de Carvalho, "Combining meta-learning and search techniques to select parameters for support vector machines," *Neurocomputing*, vol. 75, no. 1, pp. 3–13, 2012.

[26] E. Brochu, V. M. Cora, and N. de Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *CoRR*, vol. abs/1012.2599, 2010.

[27] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 2951–2959.

[28] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proc. 30th Intern. Conf. on Machine Learning*, 2013, pp. 1–9.

[29] F. Hutter, H. Hoos, K. Leyton-Brown, and T. Stützle, "Paramils: an automatic algorithm con- figuration framewor," *Journal of Artificial Intelligence Research*, no. 36, pp. 267–306, 2009.

[30] V. Nannen and A. E. Eiben, "Relevance estimation and value calibration of evolutionary algorithm parameters," in *Proc. of the 20th Intern. Joint Conf. on Art. Intelligence*, ser. IJCAI'07, 2007, pp. 975–980.

[31] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms," in *Proc. of KDD-2013*, 2013, pp. 847–855.

[32] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2944–2952.

[33] D. Krstajic, L. J. Buturovic, D. E. Leahy, and S. Thomas, "Cross-validation pitfalls when selecting and assessing regression and classification models." *Journal of cheminformatics*, vol. 6, no. 1, 2014.

[34] K. Bache and M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[35] A. L. D. Rossi and A. C. P. L. F. Carvalho, "Bio-inspired optimization techniques for svm parameter tuning," in *Proceed. of 10th Brazilian Symp. on Neural Net.* IEEE Computer Society, 2008, pp. 435–440.

[36] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[37] M. Reif, F. Shafait, and A. Dengel, "Meta-learning for evolutionary parameter optimization of classifiers," *Machine Learning*, vol. 87, pp. 357–380, 2012.

[38] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo, "Openml: Networked science in machine learning," *SIGKDD Explor. Newsl.*, vol. 15, no. 2, pp. 49–60, 2014.