An Empirical Analysis of Meta-learning for the Automatic Choice of Architecture and Components in Ensemble Systems

Diego S. C. Nascimento and Anne M. P. Canuto and André L. V. Coelho

Abstract—**Studies with ensemble systems have gained attention recently and, most of them, propose new methods for the design (generation) of different components in these systems. In parallel, new contributions of meta-learning have been presented as an efficient alternative to automatic recommendation of algorithms. In this paper, we apply meta-learning in the process of recommendation of important parameters of ensemble systems, which are: architecture and individual classifiers. The main goal is to provide an efficient way to design ensemble systems. In order to validate the proposed approach, an empirical investigation is conducted, recommending three possible types of ensemble architectures (Bagging, Boosting and Multi-Boosting) and five possible types of learning algorithms to compose the ensemble systems (individual classifiers or components). An initial analysis of the results confirms that meta-learning can be a promising tool to be used in the automatic choice of important parameters in ensemble systems.**

I. Introduction

INSTEAD of focusing on the use of individual estimators (classifiers or regressors) when applying to a particular problem, the concept of ensemble systems is to provide independent modules that seek for a decision potentially more effective than the one produced by any of the members, when acting separately. In a typical architecture of ensemble, a new input pattern is presented for all K components. The individual classifiers provide their outputs and send them to a combination method, which is responsible for providing the final output of the system. The outputs are generally combined using simple voting or weighted vote. Therefore, ensemble systems can be seen as a two-step decision making process, in which the first step is related to the decision of the individual classifiers, while the second step refers to the decision of the combination method.

In order to obtain acceptable results in terms of accuracy/generalization, it is important that all components of an ensemble system present a certain level of diversity among themselves. The need for diversity is due to the fact that if the whole set of components provides the same output, this will lead to an increase in the computational cost, without increasing the performance of the ensemble system. One way to achieve diversity is by training each component with different datasets. In this case, the methods for generating ensembles most used in the literature are: Bagging [1], Boosting [2] and Multiboosting [3]. These methods, in turn, make use of techniques for resampling data.

In parallel, new researches have been proposed in the field of meta-learning. The major contribution of this field is to exploit knowledge about the learning process, allowing us to understand and to improve the performance of machine learning algorithms [4]. For instance, in [5], the authors presented a meta-learning approach to predict the accuracy of two algorithms, multi-layer perceptron with backpropagation and with Levenberg-Marquardt. On the other hand, in [6], the authors use a hybrid approach with meta-learning and search algorithms in order to automatically adjust the parameters of a support vector machine (SVM). In [7], the authors used meta-learning to build a model to predict a ranking of performance among the main learning algorithms used in gene expression tasks.

In the context of ensemble systems, very little effort has been done to use meta-learning as a recommendation tool in the automatic design of these systems [8], [9]. In [8], for instance, it is presented an approach to create customized model ensembles on demand, inspired by Lazy Learning. In their approach, called lazy Meta-Learning, an ensemble system is created and their meta-information are used for dynamic bias compensation and relevance weighting. It is important to emphasize that the approach proposed in [8] does not apply meta-learning on the recommendation level of configuration parameters for an ensemble system, unlike the methodology we apply in this paper. In addition, in [9], the authors combined the idea of meta-learning with ensemble systems, with the goal to help the design of efficient and robust ensemble systems. However, they recommended different parameters of an ensemble systems (size and structure) of the one we recommend in this paper (ensemble architecture and individual classifiers) and they used different meta-learning methodology (meta-features and meta-learner).

The main aim of this paper is to investigate the use of meta-learning to select two important parameters in the design of ensemble systems, which are: the best ensemble architecture (bagging, boosting or multi-boosting) as well as the components to compose an ensemble system to be applied in different classification problems. The main contribution of this investigation is to present an efficient tools to design ensemble systems using meta-learning, which has not been proposed in the literature.

D.S.C. Nascimento is with Information System Group. Federal Institute of Rio Grande do Norte (IFRN), Ipanguaçu – RN, 59.508-000, Brazil. e-mail addresses: diego.nascimento@ifrn.edu.br, diegoscnascimento@gmail.com.

A.M.P. Canuto is with Informatics and Applied Mathematics Department, Federal University of Rio Grande do Norte (UFRN), Natal – RN, 59.072-970, Brazil. e-mail address: anne@dimap.ufrn.br.

A.L.V. Coelho is with the University of Fortaleza (UNIFOR), Graduate Program in Applied Informatics, Fortaleza, CE 60.811-905, Brazil. E-mail addresses: acoelho@unifor.br, coelho.alv@gmail.com.

978-1-4799-5618-0/14 $31.00 © 2014 IEEE
DOI 10.1109/BRACIS.2014.12

II. Ensemble Systems

One important issue in the design of ensemble system is the choice of the individual classifiers. The appropriate choice of the set of individual classifiers is fundamental to the overall performance of an ensemble. Depending on its particular structure, an ensemble can be realised using two main approaches: heterogeneous and homogeneous. The first approach combines different types of learning algorithms as individual classifiers. In contrast, the second approach combines learning algorithms of the same type.

As already mentioned, diversity plays an important role in the design of ensembles that are accurate and generalize well [10]. The ideal situation, in terms of combining classifiers, would be a set of classifiers that presents uncorrelated errors (diversity). Diversity in ensemble systems can be reached by using different parameter settings, different classifier training datasets and different classifier types. The most common way to promote diversity is through the use of learning strategies, also known as ensemble architectures or simply architectures, that provide different datasets for the individual classifiers of an ensemble system. The most common architectures are:

- Bagging [1]: It is based on the idea of data resampling. Diversity is promoted in Bagging by using bootstrapped replicas of the training dataset and each replica is generated by randomly drawing, with replacement, a subset of the training data;
- Boosting: This algorithm was proposed originaly by Schapire [11] and cited by Breiman [12] as Arcing (adaptive resampling and combining). It is very similar to Bagging since it also applies a resampling procedure. However, Boosting does not use training datasets obtained by uniform random resampling, but using a probability distribution assigned to each pattern of the training set and it is adaptively adjusted; and
- MultiBoosting: In [13], the authors proposed a combination of Bagging and Boosting, called Multiple Boosting, or simply MultiBoosting. The whole training process is similar to Boosting and the main difference is related to the definition of the weights, which is randomly chosen for each pattern of the training set for each sub-ensemble.

III. Meta-learning for Algorithm Recommendation

Recently, meta-learning techniques have been emerged as an efficient alternative for recommendation [14], [15]. It can be considered as an automatic process of knowledge acquisition that relates the performance of the learning algorithms with the characteristics of the machine learning problems. The idea of meta-learning can be applied to individual learning algorithms or ensemble systems. In the second case, the performance of the homogeneous ensemble systems is related to a set of characteristics (meta-features) of the corresponding machine learning problems. Hence, it acquired knowledge over the parameters of each configuration of an ensemble system and this acquired knowledge is used in the design of of these systems, when a new task is presented.

In a general perspective, the design of an algorithm (ensemble system) recommendation system is composed of four phases [16], which are: characterization of the dataset (meta-features); definition of evaluation metrics; definition of the recommendation output; and the development of the recommendation model.

In the first phase, the main aim is to find or develop features that describe appropriately the problems that have been solved by the used algorithms, aiming to provide morphological information that can be applied to the meta-learner. One of the first studies to define meta-features was the statlog project [17]. Some of the evaluation features defined by this project are: number of instances, number of attributes, the first canonical correlation, kurtosis average attributes, attribute entropy, signal/noise ratio, among others.

In the second phase, the process of selecting the best algorithm for the problems in the dataset is performed. In this case, it is necessary the application of evaluation measures in order to ensure that the best model has been selected for a specific problem, taking into account the more satisfactory performance for the analysed problem. Several evaluation metrics can be employed to evaluate the used algorithms such as, predictive accuracy, error rate, precision, F-measure, area under the ROC curve, computational cost for training/testing phases, memory necessary, complexity of the induced model and interpretation of the resulting model, among others. In this paper, we use the error rate as the main evaluation metric.

The third phase is related to the final result that will be presented by the recommendation system. In this case, the authors in [16] suggest three techniques: definition of the best algorithm, definition of a group of best algorithms or a ranking of the best algorithms. In this paper, we select to recommend the best algorithm, since it is the simplest strategy to be implemented and because it is the most important output that we desire from a recommendation system.

Finally, in the fourth phase, the goal is to learn an implicit mapping between meta-features and classes in the meta-label.

IV. Automatic Choice of Architecture and Components in Ensembles

For the proposed recommendation procedure of architecture and individual classifiers of ensemble systems, four phases are needed. The first one is related to the feature extraction of the training set. In other words, for a training set, the characterization of the dataset phase described in the previous section is performed and the meta-features are extracted (one dataset for each parameter to be suggested).

In the second phase, however, each training dataset is tested using a 10-fold cross validation process with Bagging, Boosting and Multiboosting algorithms. The ensemble systems are composed (individual classifiers) of RBF Neural Networks (NN), Decision Tree (DT), Support Vector Machines (SVM), Naïve Bayes (NB) and k-Nearest Neighbour (k-NN), in homogeneous structures.

In the third phase, the performance of the ensemble systems created in the previous phase is evaluated and the meta-labels are filled, filling all attributes of the meta-datasets. These meta-datasets are used to train the meta-learners that are responsible for recommending the architecture (architecture meta-dataset) and the individual classifiers (classifier meta-dataset) for the ensemble systems.

In the first meta-dataset, as illustrated in Figure 1, architecture meta-dataset, the label attribute indicates one of the learning strategies used in this investigation (1 – Bagging, 2 – Boosting or 3 – Multiboosting). It indicates the architecture that delivered the best performance. In order to calculate the performance of each ensemble, the average error rate of the obtained ensembles when using all individual classifiers is defined (1 – NN, 2 – DT, 3 – SVM, 4 – NB and 5 – k-NN) and we select as the best case as the one that delivered the lowest error rate.

In the second meta-dataset, the label attribute is related to the best individual classifier to the specific ensemble architecture. For this meta-dataset, we define two different approaches for recommending the best individual classifier. In the first approach, we create three different datasets, one for Bagging, one for Boosting and one for Multiboosting. This approach is called approach I and illustrated in Figure 1(a).

In the second approach, called approach II and illustrated in Figure 1(b), only one meta-dataset is considered when recommending the best individual classifier. For this second approach, all meta-datasets for Bagging, Boosting and Multiboosting were pooled into a single meta-dataset (general components in Figure 1). In this case, one meta-feature is included, which is related to the best architecture. Therefore, a recommendation is done taking into consideration the output of the first step (best architecture).

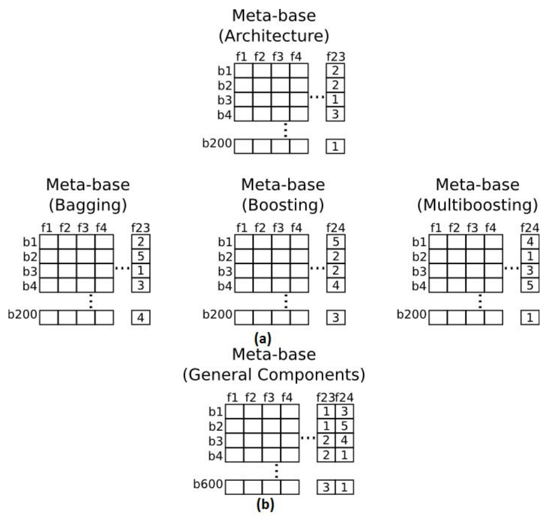

Fig. 1. Design of meta-datasets.

In the fourth phase, the meta-learners are built using the meta-datasets created in the previous phase. This phase was constructed in two different ways (based on the two approaches described above) with the same goal. In the first one (approach I), when a new instance is presented, the corresponding features are extracted and presented to two meta-learners. The first meta-learner recommends the most suitable architecture (Bagging, Boosting or Multiboosting) for this instance. Based on the output of the first meta-dataset, the second meta-dataset recommends the type of individual classifier (NN, DT, SVM, NB or k-NN) for the selected architecture (Bagging, Boosting or Multiboosting).

In the second one (approach II), when a new instance is presented, the corresponding features are extracted and presented to the first meta-dataset (architecture), that recommends the most suitable architecture for this problem. Then, based on the result of this first meta-dataset, the second meta-dataset is activated. In this second approach,the meta-learner received the extracted features, along with the recommended architecture (additional meta-feature), and it recommends the type of individual classifier to compound the ensemble.

V. Empirical Assessment

A prototype to conduct this empirical investigation was implemented using Java language. It is important to emphasize that we used some of the resources provided by the machine learning Weka framework [18]. This framework has been widely used as basis in the development and validation of different approaches in machine learning, mainly those approaches that use ensemble systems [19], [20]. In addition, for simplicity reasons, we used only homogeneous ensembles in this investigation. Therefore, we selected three architectures for homogeneous ensembles (Bagging, Boosting and Multiboosting). For each case, we used learning algotihms with different classification criteria (NN, DT, SVM, NB, and k-NN). In this investigation, we use the default parameter values of Weka package in all these learning algorithms.

In this investigation, we selected 20 classification problems extracted from UCI repository [21]. Table I describes the classification problems in terms of number of features, instances and classes.For each classification problem, we randomly created 10 new datasets, with resampling. These new datasets are used, along with the original 20 problems, in this empirical investigation. In this case, our meta-datasets contain 200 intances.

Recently, the Metal project (www.metal-kdd.org) has developed tools to help users to select the most suitable combination of pre-processing, classification and regression. In this investigation, we apply these tools to extract meta-features. The extracted meta-features are described in Table II.

A 10-fold stratified cross-validation method on the training dataset has been applied over each classification problem. In addition, the results are analysed in terms of error rates presented by each analysed model.

VI. Empirical Results

Initially, we analyse the results obtained by all the homogeneous ensembles. Tables III, IV and V present the

TABLE I
DESCRIPTION OF THE DATASETS USED

ID	Dataset	Features	Instances	Classes
1	anneal	38	898	6
2	breast-cancer	30	569	2
3	bupa	6	345	2
4	car	6	1728	4
5	colic	28	368	2
6	credit-a	15	690	2
7	diabetes	8	768	2
8	gaussian3	600	60	3
9	glass	9	214	7
10	haberman	3	306	2
11	heart-c	75	303	2
12	hepatitis	20	155	2
13	ionosphere	34	351	2
14	iris	4	150	3
15	segment	18	2310	7
16	sick	29	3772	2
17	sonar	60	208	2
18	vehicle	18	846	4
19	vote	16	435	2
20	waveform-5000	40	5000	3

TABLE II
META-FEATURES DESCRIPTIONS

Meta-features	Average	St. Dev.	Min.	Max.
Num of instances	863.68	1153.57	54.00	4500.00
Num attributes(NA)	49.40	127.66	4.00	601.00
Num of classes	3.20	1.69	2.00	7.00
Num of symbolic att	6.50	8.86	0.00	32.00
Num of numeric att	41.90	129.24	0.00	600.00
Missing values (MV)	388.48	1226.31	0.00	5495.00
MV relative (MVR)	0.02	0.05	0.00	0.23
Lines with MV total	200.88	738.81	0.00	3395.00
Lines with MVR	0.15	0.31	0.00	1.00
Mean absol. skewness	1.29	0.77	0.05	3.38
Mean kurtosis	8.61	8.46	2.70	38.36
NA with outliers	6.29	14.69	0.00	67.00
Entropy of class	1.23	0.56	0.33	2.81
Entropy of att	1.32	0.86	0.34	3.53
Mutual information	0.11	0.095	0.01	0.31
Joint entropy	2.11	0.93	0.66	4.33

TABLE III
AVERAGE ERROR DELIVERED BY BAGGING

ID	NN	DT	SVM	NB	kNN
1	0.043$\pm$0.01	0.011$\pm$0.01	0.031$\pm$0.02	0.135$\pm$0.05	**0.007$\pm$0.01**
2	0.266$\pm$0.09	0.273$\pm$0.08	0.301$\pm$0.08	**0.259$\pm$0.09**	0.286$\pm$0.10
3	0.325$\pm$0.05	**0.281$\pm$0.05**	0.417$\pm$0.03	0.441$\pm$0.09	0.362$\pm$0.11
4	0.082$\pm$0.02	0.067$\pm$0.03	0.064$\pm$0.02	0.140$\pm$0.05	**0.056$\pm$0.02**
5	0.193$\pm$0.06	**0.156$\pm$0.07**	0.179$\pm$0.07	0.215$\pm$0.08	0.198$\pm$0.05
6	0.199$\pm$0.03	**0.141$\pm$0.02**	0.148$\pm$0.02	0.222$\pm$0.04	0.196$\pm$0.04
7	0.249$\pm$0.07	0.237$\pm$0.05	**0.231$\pm$0.05**	0.241$\pm$0.06	0.289$\pm$0.06
8	0.417$\pm$0.18	0.217$\pm$0.25	**0.000$\pm$0.00**	0.333$\pm$0.24	0.017$\pm$0.05
9	0.309$\pm$0.10	**0.271$\pm$0.10**	0.391$\pm$0.11	0.485$\pm$0.10	0.300$\pm$0.10
10	0.258$\pm$0.06	0.281$\pm$0.08	0.291$\pm$0.05	**0.245$\pm$0.06**	0.333$\pm$0.06
11	**0.165$\pm$0.07**	0.198$\pm$0.08	0.171$\pm$0.07	0.171$\pm$0.07	0.241$\pm$0.08
12	**0.129$\pm$0.04**	0.187$\pm$0.06	0.142$\pm$0.05	0.149$\pm$0.08	0.180$\pm$0.06
13	**0.086$\pm$0.05**	0.091$\pm$0.05	0.108$\pm$0.05	0.177$\pm$0.05	0.137$\pm$0.04
14	**0.040$\pm$0.05**	0.060$\pm$0.08	0.047$\pm$0.04	0.040$\pm$0.05	0.047$\pm$0.04
15	0.118$\pm$0.02	**0.025$\pm$0.01**	0.073$\pm$0.01	0.199$\pm$0.02	0.032$\pm$0.01
16	0.036$\pm$0.01	**0.012$\pm$0.01**	0.060$\pm$0.01	0.073$\pm$0.02	0.036$\pm$0.01
17	0.192$\pm$0.07	0.201$\pm$0.06	0.216$\pm$0.09	0.298$\pm$0.05	**0.135$\pm$0.05**
18	0.323$\pm$0.05	**0.245$\pm$0.03**	0.259$\pm$0.04	0.556$\pm$0.06	0.298$\pm$0.04
19	0.046$\pm$0.03	**0.032$\pm$0.03**	0.032$\pm$0.02	0.097$\pm$0.03	0.081$\pm$0.04
20	0.146$\pm$0.01	0.186$\pm$0.01	**0.134$\pm$0.01**	0.200$\pm$0.01	0.258$\pm$0.02

TABLE IV
AVERAGE ERROR DELIVERED BY BOOSTING

ID	NN	DT	SVM	NB	kNN
1	0.024$\pm$0.02	**0.003$\pm$0.01**	0.004$\pm$0.01	0.058$\pm$0.03	0.007$\pm$0.01
2	**0.301$\pm$0.08**	0.346$\pm$0.10	0.322$\pm$0.10	0.305$\pm$0.10	0.311$\pm$0.11
3	0.343$\pm$0.05	**0.276$\pm$0.06**	0.339$\pm$0.06	0.362$\pm$0.07	0.359$\pm$0.11
4	0.076$\pm$0.02	**0.038$\pm$0.02**	0.061$\pm$0.02	0.098$\pm$0.02	0.062$\pm$0.02
5	0.201$\pm$0.07	**0.168$\pm$0.08**	0.206$\pm$0.04	0.222$\pm$0.07	0.198$\pm$0.05
6	0.171$\pm$0.04	0.158$\pm$0.03	**0.149$\pm$0.03**	0.199$\pm$0.04	0.197$\pm$0.04
7	0.261$\pm$0.08	0.281$\pm$0.04	**0.231$\pm$0.05**	0.238$\pm$0.04	0.291$\pm$0.07
8	0.367$\pm$0.19	0.433$\pm$0.22	**0.000$\pm$0.00**	0.283$\pm$0.24	0.017$\pm$0.05
9	0.326$\pm$0.12	**0.267$\pm$0.11**	0.405$\pm$0.12	0.542$\pm$0.10	0.295$\pm$0.09
10	0.297$\pm$0.05	0.287$\pm$0.06	0.262$\pm$0.05	**0.261$\pm$0.04**	0.336$\pm$0.06
11	0.211$\pm$0.09	0.211$\pm$0.07	**0.178$\pm$0.06**	0.185$\pm$0.08	0.247$\pm$0.08
12	0.174$\pm$0.07	0.193$\pm$0.06	0.193$\pm$0.08	**0.162$\pm$0.05**	0.180$\pm$0.06
13	**0.080$\pm$0.07**	0.086$\pm$0.03	0.105$\pm$0.05	0.091$\pm$0.05	0.134$\pm$0.04
14	0.040$\pm$0.05	0.060$\pm$0.06	**0.027$\pm$0.05**	0.060$\pm$0.05	0.047$\pm$0.04
15	0.092$\pm$0.02	**0.017$\pm$0.01**	0.073$\pm$0.01	0.201$\pm$0.02	0.030$\pm$0.01
16	0.040$\pm$0.01	**0.010$\pm$0.01**	0.053$\pm$0.01	0.058$\pm$0.01	0.037$\pm$0.01
17	0.172$\pm$0.04	0.192$\pm$0.07	0.178$\pm$0.07	0.197$\pm$0.05	**0.144$\pm$0.07**
18	0.324$\pm$0.03	**0.236$\pm$0.04**	0.260$\pm$0.04	0.556$\pm$0.05	0.299$\pm$0.03
19	0.048$\pm$0.03	**0.037$\pm$0.03**	0.039$\pm$0.03	0.039$\pm$0.02	0.074$\pm$0.04
20	0.152$\pm$0.01	0.179$\pm$0.02	**0.133$\pm$0.01**	0.200$\pm$0.01	0.262$\pm$0.02

error rate of all three architectures, respectively. These tables illustrate the error rates for all five learning algorithms used in this analysis. As we have 21 instances (cases) for each classification problem, the values presented in this section represent the average values of all 21 cases of each classification problem.

In a general perspective, it is possible to observe that the learning algorithm that provided the best performance was DT (decision tree), achieving the lowest error rate in 27 cases (out of 60), followed by SVM (13 out of 60 cases), NN (9 out of 60 cases), k-NN (6 out of 60 cases) and NB (5 out of 60 cases).

In analysing the error rates delivered by Bagging (Table III), we noticed that in around 40% of the analysed classification problems, the lowest error rates were obtained by ensembles composed of stable classifiers, such as SVM, NB and k-NN. In other words, the best results were obtained with ensembles that are composed of learning algorithms that are not suitable to be used in this architecture. In this case, when designing an ensemble using the Bagging architecture in a real classification problem, we would not usually choose the learning algorithms that provided the best results.

When analysing the performance of the ensembles generated by Boosting (Table IV), we can assess the use of this algorithm in problems with noisy and missing values, since Boosting usually delivers a poor performance in these cases, when compared with other architectures. In the cases with noisy data (mainly guassian3 and waveform-500 datasets), we can observe that ensembles generated by Boosting and composed of NN and NB present lower error rate than Bagging for Gaussian3. In addition, for the waveform-500 dataset, ensembles generated by Boosting and composed of NN provided lower error rate than Bagging. These results confirm that the Boosting method delivered a reasonable performance, even under circumstances in which it is known to be not very effective.

For the Multiboosting method (Table V), we can observe that although this method contains the main advantages of Bagging and Boosting, the performance delivered by the use of this method was superior than Bagging and Boosting in only some cases.

In order to define the best overall architecture, Table VI

TABLE V
AVERAGE ERROR DELIVERED BY MULTIBOOSTING

ID	NN	DT	SVM	NB	kNN
1	0.037±0.02	**0.004±0.01**	0.007±0.01	0.134±0.05	0.007±0.01
2	0.283±0.09	0.294±0.08	0.308±0.08	0.291±0.09	**0.279±0.13**
3	0.348±0.06	**0.276±0.06**	0.389±0.05	0.391±0.09	0.359±0.11
4	0.076±0.02	**0.049±0.02**	0.060±0.03	0.126±0.03	0.062±0.02
5	0.182±0.07	**0.155±0.06**	0.190±0.07	0.193±0.07	0.198±0.05
6	0.178±0.04	**0.140±0.04**	0.154±0.04	0.213±0.04	0.197±0.04
7	0.260±0.08	0.245±0.04	**0.229±0.05**	0.240±0.05	0.291±0.07
8	0.367±0.19	0.467±0.17	**0.000±0.00**	0.283±0.24	0.017±0.05
9	0.322±0.11	**0.220±0.09**	0.381±0.12	0.518±0.12	0.295±0.09
10	0.272±0.07	0.301±0.06	0.255±0.03	**0.252±0.06**	0.336±0.06
11	**0.168±0.07**	0.218±0.07	0.171±0.08	0.175±0.07	0.247±0.08
12	**0.142±0.06**	0.174±0.06	0.161±0.06	0.162±0.07	0.180±0.06
13	**0.068±0.04**	0.074±0.04	0.111±0.05	0.088±0.05	0.134±0.04
14	0.040±0.05	0.047±0.06	**0.027±0.03**	0.047±0.03	0.047±0.04
15	0.103±0.01	**0.019±0.01**	0.072±0.01	0.201±0.02	0.030±0.01
16	0.039±0.01	**0.009±0.01**	0.061±0.01	0.068±0.02	0.037±0.01
17	0.183±0.06	0.187±0.09	0.188±0.08	0.216±0.09	**0.144±0.07**
18	0.330±0.06	**0.236±0.04**	0.264±0.05	0.557±0.05	0.299±0.03
19	0.041±0.03	**0.032±0.03**	0.035±0.02	0.076±0.04	0.074±0.04
20	0.142±0.01	0.172±0.02	**0.134±0.01**	0.199±0.01	0.262±0.02

presents the name of the best case (BA for Bagging, BO for Boosting and MB for Multiboosting) for all 20 classification problems and five homogeneous structures. In this table, we present the architecture that provided the best performance (the lowest error rate in most of the 21 instances for each classification problem), either Bagging (BA), Boosting (BO) or Multiboosting (MB). The last line and column present the best architecture for the specific dataset (line) and homogeneous structure (column). In case of a draw, both names are presented. For example, in the last line (homogeneous ensemble structure), we have two draws, ensembles composed of decision trees (DTs), Bagging and Multiboosting, and SVMs, Boosting and Multiboosting. According to this table, we can observe that all three architectures have a similar performance, mainly when we consider the homogeneous architectures (columns), in which Bagging provided the best performance for three cases (NN, DT and KNN), while Boosting and Multiboosting provided the best performance in 2 cases (SVM and NB for Boosting and DT and SVM for Multiboosting).

In order to analyse the difference in performance, from a statistical point of view, the Friedman Test was applied, comparing the results of all three architectures, using all 220 instances of the classification problems and five homogeneous ensemble structures. The results of the Friedmann Test are presented in Table VII and the cases in which the performance are statistically different are in bold. As it can be seen in Table VII, the performance of all three architectures are very similar, since the performance are statistically different in only 20 cases (out of 100 cases).

A. Meta-learning Results

Based on the results obtained in the previous section, we observe that there is no pattern of behaviour for these parameters (architecture and components). In addition, in cases where we expected one method to have a good performance, it did not happen. The opposite also happened in some cases. Therefore, the idea of automatic design of ensembles

TABLE VI
THE BEST ARCHITECTURE ACHIEVE FOR EACH PROBLEM IN EACH HOMOGENEOUS STRUCTURE.

ID	NN	DT	SVM	NB	kNN	Best
1	BO	BO	BO	BO	BA	BO
2	BA	BA	BA	BA	MB	BA
3	BA	MB	BO	BO	BO	BO
4	BO	BO	MB	BO	BA	BO
5	MB	BA	BA	MB	BO	BA/MB
6	BO	BA	BA	BO	BA	BA
7	BA	BA	MB	BO	BA	BA
8	BO	BA	BA	BO	BA	BA
9	BA	MB	MB	BA	BO	BA/MB
10	BA	BA	MB	BA	BA	BA
11	BA	BA	MB	BA	BA	BA
12	BA	MB	BA	BA	BA	BA
13	MB	MB	BO	MB	BO	MB
14	BA	MB	BO	BA	BA	BA
15	BO	BO	MB	BA	BO	BO
16	BA	MB	BO	BO	BA	BO/BA
17	BO	MB	BO	BO	BA	BO
18	BA	BO	BA	BO	BO	BO
19	MB	BA	BA	BO	BO	BA/BO
20	MB	MB	BO	MB	BA	MB
Best	BA	BA/MB	BO/MB	BO	BA	

TABLE VII
RESULTS OF THE FRIEDMAN TEST

ID	NN	DT	SVM	NB	kNN
1	0.1313	**0.0150**	**0.0002**	**0.0048**	1.0000
2	0.1962	**0.0050**	0.6331	0.0821	0.6412
3	0.4692	1.0000	**0.0018**	**0.0060**	0.8669
4	0.5811	**0.0002**	0.6951	**0.0018**	0.0695
5	0.4994	0.1629	0.1072	0.2828	1.0000
6	0.1313	0.2143	0.6483	0.0688	1.0000
7	0.2564	0.1186	0.9623	0.7788	1.0000
8	0.6065	**0.0089**	1.0000	0.8669	1.0000
9	0.8789	**0.0342**	0.1664	**0.0388**	0.7165
10	0.1306	0.9155	0.1846	0.3941	0.8669
11	**0.0080**	0.9692	0.4493	0.9608	0.3679
12	0.1132	0.5045	0.0663	0.7558	1.0000
13	0.3172	0.1316	0.9556	**0.0004**	0.3679
14	1.0000	0.2636	0.1054	0.5488	1.0000
15	**0.0008**	0.0572	0.8825	0.5134	**0.0067**
16	**0.0319**	0.4227	0.2437	0.1394	0.6065
17	0.7026	1.0000	0.1778	**0.0014**	0.3679
18	0.5984	0.3128	0.8920	0.6703	1.0000
19	0.8521	0.3679	0.4378	**0.0004**	**0.0498**
20	**0.0020**	0.1096	0.9747	0.2359	0.0622

systems has emerged, apart from the approaches that apply bio-inspired methods, such as genetic algorithms [19], [20].

The main aim of this investigation is to learn about the performance of the different ensemble systems in different classification problems and to build meta-learners that are used to recommend the best architecture and components for an unseen classification problem. Table VIII presents the results (error rate and standard deviation) of the meta-learners (MLP, DT, SVM and k-NN) for the best architecture, while Table IX presents the recommendation of the learning algorithm to be used in the ensemble systems, when applying Bagging, Boosting and Multiboosting meta-datasets (approach I). Finally, Table X, illustrates the recommendation of learning algorithms according to a recommended architecture (approach II).

In a general perspective, we can observe from Ta-

TABLE VIII
ERROR RATES OF THE META-LEARNERS FOR ARCHITECTURE

Meta-learners			
MLP	DT	SVM	k-NN
0.271 ± 0.038	0.281 ± 0.044	0.357 ± 0.026	0.249 ± 0.040

TABLE IX
ERROR RATES FOR THE META-LEARNERS FOR COMPONENTS RECOMMENDATION: APPROACH I

	Meta-learners			
	MLP	DT	SVM	k-NN
BA	0.145 ± 0.026	0.153 ± 0.031	0.277 ± 0.008	0.237 ± 0.035
BO	0.145 ± 0.027	0.139 ± 0.032	0.274 ± 0.007	0.258 ± 0.030
MB	0.133 ± 0.027	0.136 ± 0.029	0.279 ± 0.008	0.223 ± 0.033

bles VIII, IX and X that the results obtained by the meta-learners are very promising. We obtained error rates lower than 30% in all cases (architecture and individual classifiers). For the recommendation of the individual classifiers, the error rate was lower than 15% in most of the cases. Of the meta-learners, we can observe that the decision tree has provided the lowest error rates in the recommendation process in the majority of cases. When comparing both approaches for the recommendation of the individual classifiers (approaches I and II, in Tables IX and X), the performance of both approaches are very similar, but the lowest error rate was obtained by the DT meta-learner using approach II (Table X), reaching an error rate of 0.132, which is very promising for the meta-learning field.

The results obtained in this paper corroborates to strengthen even further the idea that meta-learning can be a powerful tool to be exploited in ensemble systems.

VII. CONCLUSIONS

In this paper, we presented a meta-learning approach for the recommendation of architecture and individual classifiers in ensemble. For the architecture recommendation, we selected the most used methods, Bagging, Boosting and Multiboosting. In addition, we selected well-known learning algorithms with different classification criteria to compose the ensemble – NN, DT, SVM, NB and k-NN.

Through this analysis, we can conclude that meta-learning is a feasible technique to be used in the choice process of important parameters of an ensemble, since we were able to reach meta-learners with accuracy level superior to 85%, which can be considered as having good performance. This approach can be seen as a promising alternative in the automatic design of ensemble, when compared with some existing approaches, mainly the ones using bio-inspired methods.

As future work, we aim to investigate the use of different types of meta-features that may be more representative for the recommendation process with ensemble systems. In addition, we aim to apply the meta-learning approach to recommend other parameters in ensemble systems, such as the selection of heterogeneous components and the combination method.

TABLE X
ERROR RATES FOR THE META-LEARNERS FOR COMPONENTS RECOMMENDATION: APPROACH II

Meta-learners			
MLP	DT	SVM	k-NN
0.139 ± 0.017	0.132 ± 0.015	0.272 ± 0.004	0.241 ± 0.019

ACKNOWLEDGMENTS

This work has been financially supported by CNPq/Brazil, under process numbers 305103/2012-1 and 304603/2012-0.

REFERENCES

[1] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.
[2] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," 1996.
[3] G. I. Webb, "Multibooosting: A technique for combining boosting and wagging," *Machine Learning*, vol. 40, pp. 159–39, 2000.
[4] C. Giraud-Carrier, R. Vilalta, and P. Brazdil, "Introduction to the special issue on meta-learning," *Machine Learning*, vol. 54, no. 3, pp. 187–193, 2004.
[5] S. Guerra, R. Prudêncio, and T. Ludermir, "Meta-aprendizado de algoritmos de treinamento para redes multi-layer perceptron," *Anais do Congresso da Sociedade Brasileira de Computação (CSBC)*, pp. 1022–1031, 2007.
[6] T. A. F. Gomes, R. B. C. Prudencio, C. Soares, A. L. D. Rossi, and A. Carvalho, "Combining meta-learning and search techniques to svm parameter selection," in *Proceedings of the 2010 Eleventh Brazilian Symposium on Neural Networks*, 2010, pp. 79–84.
[7] B. F. de Souza, C. Soares, and A. C. de Carvalho, "Meta-learning approach to gene expression data classification," *International Journal of Intelligent Computing and Cybernetics*, vol. 2, pp. 285 – 303, 2009.
[8] P. P. Bonissone, "Lazy meta-learning: creating customized model ensembles on demand," in *Proceedings of the 2012 World Congress conference on Advances in Computational Intelligence*. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 1–23.
[9] R. R. Parente, A. M. P. Canuto, and J. C. Xavier, "Characterization measures of ensemble systems using a meta-learning approach," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*, 2013, pp. 1–8.
[10] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles," *Machine Learning*, vol. 51, pp. 181–207, 2003.
[11] R. E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, pp. 197–227, 1990.
[12] L. Breiman, "Arcing classifiers," *The Annals of Statistics*, vol. 26, no. 3, pp. 801–824, 1998.
[13] G. I. Webb, "Idealized models of decision committee performance and their application to reduce committee error," School of Computing and Mathematics, Tech. Rep. TR C98/11, 1998.
[14] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artificial Intelligence Review*, vol. 18, pp. 77–95, 2002.
[15] C. M. M. de Oliveira Pinto Soares, "Learning rankings of learning algorithms," Ph.D. dissertation, Universidade do Porto, 2004.
[16] A. Kalousis, "Algorithm selection via meta-learning," Ph.D. dissertation, University of Geneva, 2002.
[17] D. Michie, D. J. Spiegelhalter, and C. Taylor, *Machine learning, neural and statistical classification*. Upper Saddle River, NJ, USA: Ellis Horwood, 1994.
[18] I. H. Witten and E. Frank, *Data Mining: Pratical Machine Learning Tools and Techiniques*, 2nd ed. Elsevier, 2005.
[19] A. Canuto and D. Nascimento, "A genetic-based approach to features selection for ensembles using a hybrid and adaptive fitness function," in *The International Joint Conference on Neural Networks (IJCNN)*, 2012, pp. 1–8.
[20] D. Nascimento, A. Canuto, L. Silva, and A. Coelho, "Combining different ways to generate diversity in bagging models: An evolutionary approach," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*, 2011, pp. 2235–2242.
[21] A. Asunción and D. J. Newman, *UCI Machine Learning Repository*, University of California at Irvine, http://ics.uci.edu/~mlearn/MLRepository.html, 2007.