A Two-Step Cascade Classification Method

Eunelson J. Silva*, Alceu S. Britto Jr*, Luiz. S. Oliveira†, Fabricio Enembreck*,
Robert Sabourin‡ and Alessandro L. Koerich‡

*Postgraduate Program in Informatics
Pontifical Catholic University of Parana (PUCPR)
Curitiba, Parana, Brazil
Email: {eunelson, alceu, fabricio}@ppgia.pucpr.br

†Postgraduate Program in Informatics
Federal University of Parana (UFPR)
Curitiba, Parana, Brazil
Email: lesoliveira@inf.ufpr.br

‡École de Technologie Supérieure (ÉTS)
University of Quebec
Montreal, Quebec, Canada
Email: {robert.sabourin, alessandro.lameiras-koerich}@etsmtl.ca

***Abstract*—This paper proposes a classification approach in which monolithic and multiple classifier systems are combined in a cascading fashion. The rationale behind that is to deal with the existing trade-off between the need for increasing the accuracy, while reducing the complexity of the classification method. In other words, the idea is to offer an interesting strategy to conciliate the different levels of efforts necessary to deal with easy and hard patterns usually observed in a classification problem. The experimental results have shown that for some problems more than 90% of the instances can be processed in the first step of the cascade, saving efforts by avoiding the use of the second step in which a more complex classification method is used. It means that for some problems the reduction of the classification cost achieved more than 70% when compared to the use of an MCS. In addition to this interesting classification cost reduction, the cascade approach has shown to be able of improving the classification accuracy up to 15.19 percentage points.**

I. Introduction

Multiple Classifier Systems (MCSs) have been advocated as an alternative to the difficult task of building a monolithic classifier capable of absorbing the whole variability of a classification problem. The rationale behind that lies in the observation that different classifiers may to ensure different errors, i.e., errors presenting low correlation. Thus, it is expected that an MCS composed of diverse classifiers most of time should surpass the performance of a single classifier-based system, since the former may provide a broad coverage of the problem space variability.

With this is mind the searching for attaining high classification accuracy may frequently lead us to the construction of classification systems with an increasing complexity. However, such a trend in increasing complexity has been a source of frequent criticism against MCS, mainly when the gain in terms of accuracy is not substantial enough to justify such an increasing in complexity. An interesting discussion in this direction is presented in [1]. The authors have compared the performance of MCSs based on dynamic selection (DS) of classifiers against the best single classifier in the generated pool and the combination of all available classifiers. They have shown that for some classification problems DS-based methods may present a significant improvement in performance, but this is not the general case. In fact, their analysis have shown that the performance of DS-based methods is problem dependent, probably related to problem complexity.

Another aspect to be considered is that we may observe different levels of difficulty to discriminate among the several classes of a problem, i.e., a classification problem is usually composed of easy and hard patterns. We say that an instance is easy when it is correctly assigned with a high confidence to a single class. On the other hand, a hard instance is that for which the classification algorithm is not able to assign it to a single class since two or more classes present similar levels of confidence. Therefore, why do we need a complex system to handle all patterns? In fact, we can frequently observe that easy patterns can be handled using simple rules. This instigate us to investigate how to deal with such a trade-off between classification accuracy and time consuming (or system complexity).

An interesting alternative is the use of an approach based on cascade classification as already suggested in [2]. In such a fusion scheme, the instances rejected by the first classifier are processed by the subsequent ones in the cascade, which have an increased complexity relative to the previous ones. It is important to remember that the first study about rejection and classification error was done in [3]. In addition, some important contribution to the understanding of rejection mechanism is presented in [4], while an interesting multistage pattern recognition scheme is presented in [5]. Another important work

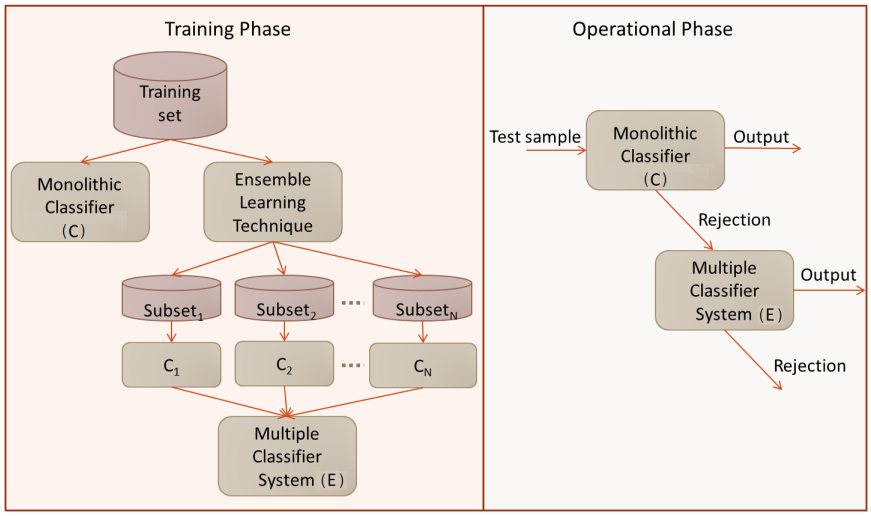

Fig. 1. Overview of the proposed cascade classification method. Training and Operational Phases

in the literature is that proposed by Viola & Jones [6] in which special attention is done to the more complex regions of images for object detection by successively combine in a cascade fashion more complex classifiers.

Inspired on their work, in this paper, a two-step cascade method is proposed. The novelty of the propose approach is the use of a monolithic classifier at the first step, while the second step is based on an MCS. Such a cascade approach is motivated by the observation that in many classification problems, the majority of instances are well-behaved, and for this reason a single classifier is able to accomplish the task satisfactorily. However, for hard instances a more sophisticated classification scheme becomes necessary.

In such a direction, in the proposed cascading classifier the instances rejected by the first step represented by a monolithic classifier are handled by the second step, where a more sophisticated classification scheme, based on an MCS, is used. In our study, as an MCS, we have investigated the combination of all classifiers in the pool generated through the use of different learning techniques and also the use of dynamic selection methods.

The research hypothesis is that by combining a monolithic classifier with an MCS composed of diverse experts in a cascading approach, we will be able to deal with problems composed of different levels of difficulty, while reducing the efforts necessary to accomplish the classification task. In other words, it means a better compromise between accuracy and complexity than using always an MCS, while keeping at least the same precision on the classification task, mainly for problems where an MCS is not necessary. It is worth noting that here the difficulty level of a classification problem is estimated based on data complexity measures, which take into account overlap between single feature values, the separability of classes and the geometry and density of classes.

With this in mind, we have the following research questions: a) Could the proposed cascade method really contribute to reduce classification cost of using always an MCS?; b) How significant could be the cost reduction provided by the proposed cascade?; and if so c) Is this reduction achieved for problems presenting different levels of difficulty?

In this paper we have addressed all theses questions by carrying out a comprehensive evaluation of the proposed approach on 12 datasets related to classification problems representing different difficulty levels. The experimental results have shown that for some easy problems the majority of the instances can be processed in the first step of the cascade saving efforts by avoiding the use of a more complex classification method. For some problems, the classification cost reduction achieved more than 70.0% when compared to the use of an MCS, while keeping similar accuracy or even showing some improvement relative to an MCS.

This paper is organized in 4 sections. Section 2 describes the proposed method. Section 3 presents the experiments undertaken to evaluate the proposed method. Finally, in Section 4 we present our conclusion and suggestions for further work.

II. Proposed Method

The proposed method combines a monolithic classifier (C) in the first step and a multiple classifier system (E) in the second one to make up a two-step cascade classification method. The second step is designed to process just the instances rejected at the first step. By configuring properly the rejection threshold of the classifier C, we can leave for E just the hard cases for which a more sophisticated classification scheme is needed. The pool of classifiers in E is generated using the same training samples used for training the monolithic classifier of the first step.

Figure 1 presents a general overview of the proposed method, describing its two phases. In the Training Phase the monolithic classifier C and the pool of classifiers necessary for E are trained. As base classifiers, we have considered the K-nearest Neighbors (KNN) and the Support Vector Machines (SVMs). The motivation is that the former is usually the choice when pools of weak classifiers are generated, while the latter is usually the choice when a robust monolithic classifier is needed. The pool in E is generated considering three different learning techniques capable of training pools of diverse classifiers: Bagging [7], Boosting [8], and Random

Subspace Selection (RSS) [9]. Thus, we may have diverse classifiers by training them on different subsets of the training set or on different representations (input features). In addition, we have also considered a different approach for the Boosting technique (BoostW), where the rejections of the monolithic classifier in the first step are boosted before being used for training the classifiers of the pool. We did that by considering a different weight for the rejected instances, before starting the original Boosting technique. The rationale behind that is to increase the probability of the instances rejected by the first step being selected to train the classifiers for the pool. The pool size was defined as 10, except for the RSS technique because of the small number of features available in some of the classification problems used. The details about the pool generation are presented in Section III-C.

Besides the combination of all classifiers in the pool using the majority voting rule, we have also investigated the use of dynamic selection of classifiers. For this purpose, we use well-known strategies to evaluate the competence of the classifiers in a local region around the test sample defined in a validation dataset, such as Overall Local Accuracy (DS-OLA) [10], Local Class Accuracy (DS-LCA) [10], Knora-Eliminate (DS-KE) and Knora-Union (DS-KU) [11].

Still in the training phase, for each classification problem, after training the classifiers, a rejection level (λ) is defined on a validation set by considering that the error rate must be $\leq 1\%$. This is done for the monolithic classifier (C) and also for each MCS approach (E) considered for the second step of the proposed cascade. It is worth to notice that the rejection scheme for the second step is done based on the whole MCS, i.e., based on the result obtained with the fusion of the classifiers constituting E.

During the Operational Phase, an unknown sample is submitted to the classifier C. It is expected that easy instances will be classified in the first step of the method, while difficult ones will be rejected. When rejected, an instance is submitted to the second step E, which may also reject it.

III. Experimental Results

This section presents the experiments undertaken to validate the proposed cascade method. Figure 2 presents a general overview about the composition of our experimental protocol. We considered 2 base classifiers (KNN and SVM), 4 pool generation methods (Bagging, Boosting, Random Subspace, BoostingW), 5 methods for classifier combination (All, DS-OLA, DS-LCA, DS-KE, DS-KU) and 12 classification problems. The main objective is to investigate whether the cascade approach can reduce or not the efforts necessary for the classification task, while keeping a similar recognition performance.

We start by presenting the datasets used in our experiments describing them in terms of main features and also some idea about their level of difficulty. Afterwards, the experiments are divided into three sets. In the first set of experiments we have trained and compared the performance of the monolithic classifiers for the first step of our cascade approach, with and without considering rejection. The idea is to show the recognition rate of the monolithic classifiers and the impact of considering a rejection mechanism.

The second set of experiments evaluates two different MCSs for the second step, as follows: a) the use of ensembles represented by the combination of all classifiers available in the pool; and b) four DS-based methods, i.e. the use of dynamic selection of classifiers. In the last set of experiments we evaluate the cascade approach considering different setups.

The performance of the evaluated setups is computed in terms of recognition (accuracy), error and reliability rates, as denoted by the Equations 1, 2 and 3 respectively. In these equations N is the total number of samples, while Nc and Ne are the total of well-recognized and misrecognized samples, respectively.

$$Recog = \frac{Nc}{N} \times 100 \tag{1}$$

$$Error = \frac{Ne}{N} \times 100 \tag{2}$$

$$Reliability = \frac{Nc}{Nc + Ne} \times 100 \tag{3}$$

The reduction in terms of cost of classification is estimated using a measure called total-number of value features (TVF), which was proposed in [12] and is given by:

$$TVF = \sum_{i=0}^{L} f_i * m_i \tag{4}$$

where, L is the number of classifiers, while f_i and m_i are the number of features and the number of classified samples related to the classifier i. This measure is useful to show the burden on the classifier for a given classification task.

A. Datasets

Table I presents the main features of the datasets used to evaluate the proposed method. Most of datasets are available at the UCI Machine Learning Repository [13], except the FS dataset which is a robust set of macroscopic images of forest species. FS is available at "http://web.inf.ufpr.br/vri/image-and-videos-databases/forest-species-database", and a detailed description can be found in [14].

TABLE I. Description of the used Datasets

Dataset	# classes	# training samples	# testing samples	# features
Liver Disorder (LD)	2	172	173	6
Haberman (HB)	2	153	153	3
Blood (BD)	2	374	374	4
Pima Diabetes (PD)	2	384	384	8
Vehicle (VE)	4	423	423	18
Sonar (SO)	2	104	104	60
Ionosphere (IO)	2	175	176	34
Forest Species (FS)	41	11768	35304	1352
Wine (WI)	3	89	89	13
Wisconsin Breast Cancer (WC)	2	284	285	30
Image Segmentation (IS)	7	210	2100	19
Iris (IR)	3	75	75	4

To have an idea about how difficult are the classification problems used in our experiments, we have computed three

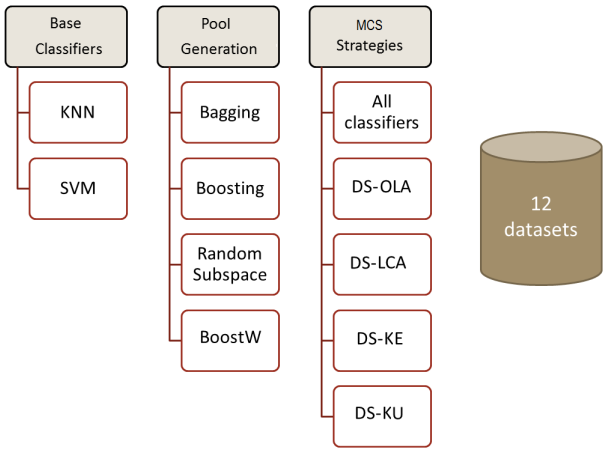

Fig. 2. Experimental Protocol Composition: 2 base classifiers (KNN, SVM); 4 pool generation methods (Bagging, Boosting, Random Subspace, BoostW); 5 MCS strategies (All classifiers; DS-OLA, DS-LCA, DS-KE; DS-KU); and 12 datasets.

complexity measures for each classification problem. According to the definitions in [15] and [16], we have selected one measure of overlap between single feature values ($F1$ - maximum Fisher's discriminant ratio), one measure of separability of classes ($N2$ - ratio of average intra/inter class nearest neighbor distance) and one measure related to the geometry and density of classes ($N4$ - Nonlinearity of the one-nearest neighbor classifier). All these measures were computed using the DCOL library described in [17].

The Fisher's Discriminant Ratio ($F1$) is a well-known measure of class overlapping which is calculated over each single feature dimension as denoted in Equation 5, where M is the number of classes and μ is the overall mean, while n_i, μ_i and s_j^i are the number of samples, the mean and the jth sample of the class i, respectively. In this generalization of $F1$, δ is the Euclidian distance. A high $F1$ value indicates the presence of discriminating features and hence a classification problem easier.

$$F1 = \frac{\sum_{i=1}^{M} n_i \cdot \delta(\mu, \mu_i)}{\sum_{i=1}^{M} \sum_{i=1}^{n_i} \delta(s_j^i, \mu_i)} \tag{5}$$

The $N2$ measure is based on a non-parametric separability of classes. It compares the intraclass dispersion with the interclass separability, as denoted in Equation 6. For this purpose, let $\eta_1^{intra}(s_i)$ and $\eta_1^{inter}(s_i)$ denote the intra and inter class nearest neighbors of the sample s_i, while δ represents the Euclidian distance. As can be observed, $N2$ calculates the ratio between the intra and the inter class dispersions. A small $N2$ value suggests high separability, and consequently, a classification problem easier.

$$N2 = \frac{\sum_{i}^{N} \delta(\eta_1^{intra}(s_i), s_i)}{\sum_{i}^{N} \delta(\eta_1^{inter}(s_i), s_i)} \tag{6}$$

The last measure, named $N4$, creates a test set from a training set using a linear interpolation between randomly drawn pairs of points from the same class. Afterwards, the error rate of the KNN on this test set is measured. In other words, $N4$ uses the error rate of KNN with the training set to describe the nonlinearity of the KNN classifier. A small $N4$ value suggests high separability, and consequently, a classification problem easier (see Equation 7).

$$N4 = error\_rate(KNN(training\_set)) \tag{7}$$

TABLE II. DATASETS RANKED BY DIFFICULTY. THE VALUES OF F1, N2 AND N4 COMPLEXITY MEASURES AND THE MEAN RANK (MR) FOR EACH DATASET

	F1	N2	N4	MR
LD	0.055	0.91	0.342	1.7
HB	0.189	0.76	0.364	2.7
BD	0.298	0.63	0.396	3.3
PD	0.577	0.84	0.274	4.3
VE	0.451	0.62	0.299	5.3
SO	0.466	0.74	0.094	7.0
IO	0.614	0.63	0.159	7.0
FS	1.854	0.79	0.103	7.3
WI	3.831	0.54	0.131	10.0
WC	3.405	0.56	0.013	11.0
IS	8.809	0.05	0.111	11.6
IR	7.097	0.13	0.081	12.0

Table II shows the values of the three complexity measures for each dataset. The datasets in that table are ranked according to the values of the mean rank (MR) estimated considering the three complexity measures. As one may see, the Liver Disorder (LD) dataset shows the highest complexity, while the IR dataset is the most easy problem according to the estimated measures. We expect that for the less difficult problems the first step of the proposed cascade can classify most of instances, while for the hard problems the second step will be more demanded. In fact, the final objective is not to show some gain in terms of recognition performance, but reduce the effort to classify the instances without losing accuracy when compared to an MCS.

B. First Step - Monolithic Classifier

We have considered the KNN and the SVM as the two base classifiers. Table III shows the best parameters of each classifier for each dataset. In case of KNN classifier the value of K is presented, while for SVM classifier the values of parameters $Gamma$ and $Cost$. In addition, this table contains the value of the rejection threshold (λ) estimated for each

problem on a validation set representing 25% of the training samples.

TABLE III. BEST PARAMETERS OF THE TRAINED MONOLITHIC CLASSIFIERS FOR EACH DATASET. BEST K FOR THE KNN CLASSIFIER, BEST $Gamma$ AND $Cost$ FOR THE SVM CLASSIFIER, AND THE REJECTION THRESHOLD (λ) ESTIMATED ON THE VALIDATION SET TO OBTAIN AN ERROR RATE $\leq 1\%$

Dataset	KNN		SVM		
	K	λ	$Gamma$	$Cost$	λ
LD	9	0.89	1	14	0.93
HB	9	1.0	1	16	0.83
BD	10	1.0	10	8	0.85
PD	5	1.0	0.1	16	0.93
VE	1	0.99	1	16	0.80
SO	1	0.99	0.1	16	0.79
IO	5	1.0	1	16	0.95
FS	1	1,00	0,01	16	0,73
WI	14	0.65	0.1	16	0.44
WC	8	0.75	1	10	0.77
IS	1	0.96	1	14	0.66
IR	3	0.66	0.1	16	0.62

Table IV presents the performance of the monolithic classifiers representing the first step of the proposed cascade method on the testing set, with and without rejection.

As one may see for some of the datasets the first step can recognize most of patterns even when the rejection mechanism is considered. This is the case of IR, WI and WC datasets, which are easy problems as shown in Table II. On the other hand, LD, HB, BD and PD have shown to be the worst cases, and their instances are highly rejected by both SVM and KNN monolithic classifiers. It is possible to observe that there is an interesting relation between the problem difficulty and the rejection as most of the difficult problems in Table II (LD, HB, BD, PD, for instance) have a high rejection rate in the first step. On the other hand, easy problems like WI, WC and IR have shown a comparatively low rejection rate.

The KNN-based classifier in the first step can classify, correctly or erroneously, in average around 53% of the problem instances, rejecting about 47%, which will be submitted to the second step. When using the SVM-based classifier, the number of instances classified in the first step is almost the same of the KNN, about 51%, while about 49% need to be processed by the second step. Such observations already confirmed that a significant amount of instances can be solved by the first step of the proposed cascade. However, the question is how many rejected instances can be classified by the second step? This is the subject of the next sections.

C. Second Step - MCS

At this point the first task was to generate pools of diverse classifiers. As described before we have used three different learning techniques capable of training pools of diverse classifiers: Bagging [7], Boosting [8], and Random Subspace Selection (RSS) [9]. In addition, for Boosting we have tried to initialize the instances with different weights. Named BoostW, in this case the samples rejected or mis-classified by the first step received a weight greater than those correctly classified.

For Bagging and Boosting, 66% of the training samples were used to generate the subsets for training each classifier. Pools with 10 classifiers were generated for each problem, except for some problems when RSS was used. Table V shows the cardinality and the number of classifiers generated when the RSS was used. As one may see, for some problems, thanks to the small number of features it was not possible to generate 10 classifiers using RSS.

Different variants of two possible approaches were evaluated for the second step of the cascade method. First, the use of ensembles without any selection mechanism, in which all classifiers available in the generated pool are combined. Second, the use of dynamic classifier selection methods. In both cases the same pools of classifiers were used.

TABLE V. DETAILS OF THE POOLS GENERATED WITH RSS

Dataset	Subset Cardinality	# of Classifiers
LD	3	10
HB	2	3
BD	2	6
PD	4	10
VE	6	10
SO	12	10
IO	8	10
FS	100	10
WI	6	10
WC	5	10
IS	4	10
IR	2	6

As mentioned before, the rejection level of each MCS is also defined considering an error rate $\leq 1\%$ on a validation set. All classifiers in the pool are combined using majority voting rule.

A total of 8 ensembles were evaluated for each problem, since we have considered 2 base classifiers (KNN, SVM) and 4 techniques to generate the pools. Therefore, 88 experiments were carried out. Table VI summarizes all the results of the ensembles evaluated for the second step. It shows the best result observed for each problem. It is worth noting that these results do not consider the cascade scheme. However, they are important to be compared with the cascade performance. As expected, the ensemble configuration is problem dependent. However, despite the variation on the ensemble configuration, we observed high rejection rates for hard problems (LD, HB, BD, PD and VE, for instance), and low rejection rates for easy problems (SO, WI, WC, IS and IR, for instance).

TABLE VI. PERFORMANCE OF THE ENSEMBLES EVALUATED FOR THE SECOND STEP ON THE TEST SETS. ALL CLASSIFIERS IN THE POOL ARE COMBINED USING MAJORITY VOTE. THE BEST RESULT FOR EACH CLASSIFICATION PROBLEM

Dataset	Learning Technique	Base Classifier	Recog. rate	Error rate	Rejection rate	Reliability rate
LD	RSS	SVM	16.18	5.20	78.61	75.68
HB	Boosting	KNN	13.07	3.27	83.66	80.00
BD	RSS	SVM	35.56	4.01	60.43	89.86
PD	Bagging	SVM	27.86	2.08	70.05	93.04
VE	Bagging	SVM	53.90	2.13	43.97	96.20
SO	Boosting	KNN	84.62	15.38	0.00	84.62
IO	RSS	SVM	86.36	2.84	10.80	96.82
FS	Bagging	SVM	42.51	4.19	53.30	91.02
WI	Bagging	SVM	98.88	1.12	0.00	98.88
WC	Bagging	SVM	97.19	1.40	1.40	98.58
IS	Boosting	KNN	91.62	8.38	0.00	91.62
IR	Bagging	KNN	97.33	1.33	1.33	98.65

Another option that was also evaluated for the second step of our cascade approach is the use of DS-based methods. However, firstly, we have delimited the scope of our experiments. The reason is that, if we consider four DS methods (DS-OLA, DS-LCA, DS-KE, and DS-KU), four ensemble

TABLE IV. PERFORMANCE OF THE MONOLITHIC CLASSIFIERS ON THE TEST SETS WITH AND WITHOUT REJECTION

Dataset	KNN						SVM					
	w/o Rejection		with Rejection				w/o Rejection		with Rejection			
	Recog.	Error	Recog.	Error	Rejection	Reliability	Recog.	Error	Recog.	Error	Rejection	Reliability
LD	63.58	36.42	1.16	0	98.84	100.0	69.36	30.64	5.78	1.16	93.06	83.33
HB	75.82	24.18	0	0	100.0	-	75.16	24.84	9.15	1.31	89.54	87.50
BD	79.14	20.86	0	0	100.0	-	78.34	21.66	2.14	0.27	97.59	88.89
PD	73.44	26.56	0	0	100.0	-	77.86	22.14	8.59	0.52	90.89	94.29
VE	70.69	29.31	70.69	29.31	0	70.69	79.67	20.33	53.19	1.89	44.92	96.57
SO	84.62	15.38	84.62	15.38	0	84.62	86.54	13.46	46.15	4.81	49.04	90.57
IO	85.80	14.20	0	0	100.0	-	94.89	5.11	55.11	0.57	44.32	98.98
FS	56.43	43.57	0	0	100.0	-	71.64	28.36	37.69	15.79	46.52	89.33
WI	97.75	2.25	92.13	1.12	6.74	98.79	100.0	0	100.0	0	0	100.0
WC	97.89	2.11	85.96	0.70	13.33	99.19	98.60	1.40	93.33	0.70	5.96	99.25
IS	91.62	8.38	91.62	8.38	0	91.62	92.10	7.90	81.0	1.62	17.38	98.04
IR	96.00	4.00	96.00	4.00	0	96.00	96.00	4.00	93.33	1.33	5.33	98.59

techniques (Bagging, Boosting, BoostingW and RSS), and two base classifiers (KNN and SVM), it sums up another 32 possible configurations, plus the 8 based on ensembles, it will sums up 40 possible configurations for the second step. In addition, in the next experiments, when the cascade is evaluated, we also need to consider 4 variations related to the 2 possibilities of base-classifiers for each step (KNN, SVM), then $40 \times 4 = 160$ experiments for each problem dataset.

TABLE VII. PERFORMANCE OF THE DS METHODS EVALUATED FOR THE SECOND STEP ON THE TEST SETS. POOLS OF KNNS GENERATED USING BAGGING. THE BEST RESULT FOR EACH CLASSIFICATION PROBLEM. "*" STANDS FOR ANY METHOD

Dataset	DS Method	Recog.	Error	Rej.
LD	DS-OLA	13.29	7.51	79.19
HB	DS-KU	10.46	1.96	87.58
BD	DS-KE	10.96	1.07	87.97
PD	DS-KU	17.19	1.04	81.77
VE	*	0.00	0.00	100.00
SO	DS-LCA	14.42	7.69	77.88
IO	*	0.00	0.00	100.00
FS	*	0.00	0.00	100.00
WI	DS-KU	95.51	1.12	3.37
WC	DS-OLA	94.39	1.40	4.21
IS	DS-LCA	13.57	4.00	82.43
IR	DS-KU	97.33	2.67	0.00

It is worth to remember that our objective is to show that by using the proposed cascade scheme we can save efforts necessary to classify the instances of a problem by dealing with easy instances in the first step. It is not the case of comparing the performance of all possible configurations when using a DS method. With this in mind we have done some choices for the set of experiments related to DS-methods, as follows: the KNN was selected as the base classifier. This instance-based method is usually the choice in the literature to construct a pool of weak classifiers. Bagging as the technique to generate the pool of 10 classifiers.

Table VII summarizes all the results of the DS-methods evaluated for the second step. It shows the best result observed for each problem. It is worth noting that these results do not consider the cascade scheme. However, they are important to be compared with the cascade performance. As one may see, the DS-method surpassed the monolithic classifier in 6 over 12 datasets, while, except for IR dataset, they performed worst than combining all classifiers.

D. Cascade Classifier

Table VIII summarizes the best results obtained by the proposed cascade method for each dataset. Beyond the accuracy, we have estimated the impact on the computational cost when considering the proposed cascade method. To this end, we have estimated the computational cost reduction based on the TVF measure as the ratio of the TVF for the entire cascade method and the TVF considering that the problem is always handled by an MCS (see Equation 8). As expected, for easy classification problems we have observed a significant cost reduction when using the proposed method relative to an MCS, while for hard problems the cost reduction is lower or sometimes we can observe a small cost increasing relative to an MCS.

$$CostReduction = \left(1 - \frac{TFV_{Cascade}}{TFV_{MCS}}\right) * 100 \qquad (8)$$

For instance, considering a hard problem as the BD dataset, only 2.41% of the instances were classified in the first step. From this total, 2.14% were correctly classified, 0.27% were misclassified, while 97.59% were rejected by the used monolithic SVM classifier. At the second step from the total amount of rejected instances, 39.18% were classified, being 35.34% correctly and 3.84% were misclassified. The second stage rejected 60.82% of the instances already rejected by the first step. Finally, we can see that the recognition rate for the BD dataset increases from 2.14% to 36.63%. Albeit small, the impact on the computational cost was negative, i.e., the cost augmented in (7.1%).

In the other hand, considering an easy problem like WC dataset, 94.03% of the instances were classified in the first step. From this total, 93.33% were correctly classified, 0.70% were misclassified, while only 5.96% were rejected by the used monolithic SVM classifier. At the second step from the total amount of rejected instances, 76.47% were classified, being 64.71% correctly and 11.76% were misclassified. The second stage rejected 23.53% of the instances already rejected by the first step. Finally, we can see that the recognition rate for the WC dataset increases from 93.33% to 97.19%. The impact on the computational cost was positive, i.e., the cost was reduced in (84.04%). It is important to say that for the datasets where the RSS technique present the best results (BD and IR), the cost of using the second step is lower also because the number of features is smaller than that used to train the monolithic classifier of the first step.

As one may see, as expected, the best monolithic classifiers in the experiments were the SVMs. From 12 datasets, 4 (IR, IS, WC and WI) had more than 80% of instances classified in the first step. Three datasets (IR, WI and WC) presented less

TABLE VIII. THE BEST CASCADE RESULT FOR EACH DATASET. THE RECOGNITION (RECOG.), ERROR AND REJECTION (REJ.) RATES OF THE FIRST AND SECOND STEPS; AND THE FINAL CASCADE RECOGNITION AND RELIABILITY RATES WITH THE CORRESPONDING COST REDUCTION

	1st Step				2nd Step						Cascade		
Data	Base Clas.	Recog.	Error	Rej.	Base Clas.	Learning Tech.	Dynamic Selection	Recog.	Error	Rej.	Recog.	Reliability	Cost Reduction
LD	SVM	5.78	1.16	93.06	KNN	Bagging	DS-OLA	13.66	8.70	77.64	18.50	66.67	-3.06
HB	SVM	9.15	1.31	89.54	SVM	Bagging	-	10.22	2.92	86.86	18.30	82.35	0.46
BD	SVM	2.14	0.27	97.59	SVM	RSS	-	35.34	3.84	60.82	36.63	90.13	-30.92
PD	SVM	8.59	0.52	90.89	SVM	Bagging	-	21.20	1.72	77.08	27.86	93.04	-0.89
VE	SVM	53.19	1.89	44.92	SVM	Boosting	-	14.21	10.53	75.26	59.57	90.00	45.08
SO	SVM	46.15	4.81	49.04	KNN	Boosting	-	82.35	17.65	0.00	86.54	86.54	40.96
IO	SVM	55.11	0.57	44.32	SVM	RSS	-	70.51	5.13	24.36	86.36	96.82	13.18
FS	SVM	37.69	15.79	46.52	SVM	Bagging	-	43.02	5.03	51.95	57.70	80.29	43.48
WI	SVM	100.0	0.0	0.0	-	-	-	-	-	-	100.0	100.0	90.00
WC	SVM	93.33	0.70	5.96	SVM	Bagging	-	64.71	11.76	23.53	97.19	98.58	84.04
IS	SVM	81.00	1.62	17.38	KNN	Boosting	-	65.75	34.25	0.0	92.43	92.43	72.62
IR	SVM	93.33	1.33	5.33	SVM	BoostW*	-	100.0	0.0	0.0	98.67	98.67	84.67

than 10% of instances rejected in the first step. The dataset WI presents no rejections, i.e., all instances were classified in the first step. On the other hand, 4 datasets (BD, HB, LD and PD) had more than 89% of instances rejected in the first step. As one may see, all these observations have high correlation with the results in Table II.

By using the second step the recognition performance was improved up to 40.39 percentage points when compared with the corresponding first step. The second step was able to recognize up to 100% of the instances rejected by the first step. Most of time the ensembles based on Bagging and Boosting methods achieved the best results in the cascade approach and the DS methods showed the best results only for the LD problem.

From Table IX, it is possible to compare the best classification results obtained from the monolithic classifier and the MCS approaches with the best results obtained with the cascade approach. As we can see, the accuracy provided by the cascade approach most of time surpasses the accuracy of all other approaches. It is important to notice that the MCS in most of cases are the combination of all classifiers in the pool, except for the LD dataset for which DS-OLA presented the best result. As mentioned before, this was not the main objective of the proposed method, i.e., to improve accuracy. However, this has shown to be possible.

TABLE IX. RECOGNITION RATES (%) OF THE PROPOSED CASCADE APPROACH COMPARED WITH BOTH THE PERFORMER MONOLITHIC AND MCS APPROACH FOR EACH DATASET. ALL RESULTS CONSIDERING THE REJECTION SCHEME. THE BEST RESULTS ARE IN BOLDFACE

Dataset	Monolithic	MCS	Proposed Cascade
LD	5.78	16.18	**18.50**
HB	9.15	13.07	**18.30**
BD	2.14	35.86	**36.63**
PD	8.59	27.86	**27.86**
VE	53.19	53.90	**59.57**
SO	46.15	84.62	**86.54**
IO	55.11	**86.36**	**86.36**
FS	37.69	42.51	**57.70**
WI	**100.00**	98.88	**100.00**
WC	93.33	**97.19**	**97.19**
IS	81.00	91.62	**92.43**
IR	93.33	97.33	**98.67**

At this point we have the answers for our three research questions. First, the experiments have shown that the cascade method can really contribute to reduce the cost of classification task. Second, for easy problems, the reduction was very significant, on 8 over 12 datasets some reduction were observed, being 4 superior to 70%. Finally, we can say that the observed cost reduction is problem dependent, and it is related to the its level of difficulty.

IV. CONCLUSION

In this work we have presented a cascade classification method, in which a monolithic classifier is combined with an MCS. The rationale here was to better deal with easy and hard instances of classification problems, since it provides a better trade-off between classification accuracy and complexity than the current methods that employ always an MCS. The experimental results have shown that the propose method is very promising. The first step (monolithic classifier) was able to classify up to 100% of instances for some problems, without rejecting any instance for the second stage. It means up to 90% of cost reduction when compared to the use of an MCS. On the other hand, for a most complex problem, 97.59% of samples were rejected by the first step. Working on the rejection of the first step, for some problems, the MCS was able to classify up to 100% of the rejected instances. In addition, most of time the final cascade accuracy surpassed the use of the corresponding monolithic classifier and the MCS used in a separated way.

Further work can be done to better investigate the relation of the problem complexity and the performance of the proposed cascade method.

ACKNOWLEDGMENTS

This research has been supported by the Brazilian National Council for Scientific and Technological Development (CNPq) and by the Fundação Araucária that supports scientific and technological development in the State of Parana (Brazil).

REFERENCES

[1] A. S. Britto Jr., R. Sabourin, L. E. S. Oliveira, Dynamic selection of classifiers - a comprehensive review, Pattern Recognition 47 (11) (2014) 3665 – 3680. doi:http://dx.doi.org/10.1016/j.patcog.2014.05.003.

[2] L. S. Oliveira, A. S. J. Britto, R. Sabourin, Improving cascading classifiers with particle swarm optimization, in: Proceedings of the Eighth International Conference on Document Analysis and Recognition, ICDAR05, IEEE Computer Society, Washington, DC, USA, 2005, pp. 570–574.

[3] C. Chow, On optimum recognition error and reject tradeoff, Information Theory, IEEE Transactions on 16 (1) (1970) 41–46. doi:10.1109/TIT.1970.1054406.

[4] G. Fumera, F. Roli, G. Giacinto, Reject option with multiple thresholds, Pattern Recognition 33 (12) (2000) 2099–2101.

[5] P. Pudil, J. Novovicova, S. Blaha, J. Kittler, Multistage pattern recognition with reject option, in: Pattern Recognition, 1992. Vol.II. Conference B: Pattern Recognition Methodology and Systems, Proceedings., 11th IAPR International Conference on, 1992, pp. 92–95. doi:10.1109/ICPR.1992.201729.

[6] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, Vol. 1, 2001, pp. I–511–I–518 vol.1. doi:10.1109/CVPR.2001.990517.

[7] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.

[8] R. Schapire, Y. Freund, P. Bartlett, W. Lee, Boosting the margin: A new explanation for the effectiveness of voting methods, in: Proc. of 14^{th} International Conference on Machine Learning, Nashville, USA, 1997, pp. 322–330.

[9] T. K. Ho, The random subspace method for constructing decision forests, IEEE Trans. on Pattern Analysis and Machine Intelligence 20 (8) (1998) 832–844.

[10] K. Woods, W. Kegelmeyer Jr, K. Bowyer, Combination of multiple classifiers using local accuracy estimates, Pattern Analysis and Machine Intelligence, IEEE Transactions on 19 (4) (1997) 405–410. doi:10.1109/34.588027.

[11] A. Ko, R. Sabourin, A. Britto Jr., From dynamic classifier selection to dynamic ensemble selection, Pattern Recognition 41 (5) (2008) 1718–1731. doi:10.1016/j.patcog.2007.10.015.

[12] A. Abbasi, H. Chen, Applying authorship analysis to extremist-group web forum messages, IEEE Intelligent Systems 20 (5) (2005) 67–75.

[13] A. Asuncion, D. J. Newman, UCI machine learning repository, URL:http://www.ics.uci.edu/ mlearn/MLRepository.html (2007).

[14] J. Martins, L. S. Oliveira, S. Nisgoski, R. Sabourin, A database for automatic classification of forest species, Machine Vision and Applications (2012) 1–10.

[15] T. Ho, M. Basu, Complexity measures of supervised classification problems, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2002) 289–300.

[16] J. Sanchez, R. Mollineda, J. Sotoca, An analysis of how training data complexity affects the nearest neighbor classifiers, Pattern Anal. Appl. 10 (3) (2007) 189–201. doi:10.1007/s10044-007-0061-2.

[17] A. Orriols-Puig, N. Macià, T. K. Ho, Documentation for the data complexity library in C++, Tech. rep., La Salle - Universitat Ramon Llull (2010).