On evaluating the online local pool generation method for imbalance learning

Mariana A. Souza[†§], George D. C. Cavalcanti[†], Rafael M. O. Cruz[‡] and Robert Sabourin[§]
[†]Centro de Informática - Universidade Federal de Pernambuco, Recife, Pernambuco, Brazil
Email: mariana.araujo.souza@gmail.com, gdcc@cin.ufpe.br
[‡]Stradigi AI, Montreal, Quebec, Canada
Email: rafaelmenelau@gmail.com
[§]École de Technologie Supérieure - Université du Québec, Montreal, Quebec, Canada
Email: robert.sabourin@etsmtl.ca

Abstract*—Imbalanced problems are characterized by a disproportion between the number of samples from the classes in a classification problem. This difference in amount of examples may lead to a bias toward the majority class, hindering the recognition of the underrepresented minority class. Ensemble methods have been widely used for dealing with such problems, and have been shown to perform well on them. In this context, Dynamic Selection (DS) approaches, which perform the classification task on a local level, have been receiving some attention for their promising results. More specifically, the Frienemy Indecision Region Dynamic Ensemble Selection++ (FIRE-DES++) framework, which has yielded state-of-the-art results on imbalanced problems, use a data preprocessing technique for noise removal and a class-balanced neighborhood definition for coping with imbalanced datasets. A different DS-based approach proposed in a previous work, an online local pool generation method, generates on the fly locally accurate classifiers for labelling samples near the class borders. Though the local generation of the classifiers may reduce the impact of class imbalance on the performance of the technique, its suitability for imbalance learning was not yet evaluated. Thus, in this work we evaluate how well the online local pool generation method deals with imbalanced problems. We perform a comparative analysis with a baseline technique using three Dynamic Classifier Selection (DCS) techniques over 64 imbalanced datasets and four performance measures. We also evaluate the use of the preprocessing and balanced neighborhood definition steps from the FIRE-DES++ on the online scheme to assess their impact on the performance of the method. Moreover, we evaluate the online technique and its variants against seven state-of-the-art ensemble methods, including both static and DS approaches. Experimental results show that the approach of locally generating the classifiers is advantageous for imbalance learning, providing an improvement to the DCS techniques and yielding state-of-the-art results. Furthermore, the addition of the noise removal and the balanced neighborhood definition steps to the online scheme improved the overall results of the technique, which indicates the advantage of including such steps in DS-based techniques.

I. Introduction

Imbalanced distributions, in which a given class is underrepresented in comparison with the remaining ones in the problem, may be encountered in many real-world classification problems, such as biomedical diagnosis [1], detection of fraudulent bank account transactions [2], image retrieval [3], text classification [4], and so on. For two-class problems, the class with fewer instances is referenced as the minority or positive class, while the class that contains the most instances is called the majority or negative class. Since one class outnumbers the other, many traditional classification models may end up biased in favor of the majority class, leading to a poor recognition rate of the examples from the minority class [5].

Several techniques have been proposed to deal with imbalance learning, and they can be categorized into four groups [6], [7]: algorithm-level approaches, data-level approaches, cost-sensitive learning frameworks, and ensemble based approaches. Algorithm-level approaches adapt the existing learning algorithms so as to adjust their bias toward the minority class. Data-level approaches, on the other hand, make use of resampling techniques to rebalance the class distribution of the problem. Cost-sensitive frameworks combine both data-level and algorithm-level approaches by adding cost to instances and modifying the learning procedure to incorporate those costs. Finally, the ensemble-based approaches make use of one of the previous approaches, usually data preprocessing, and an ensemble learning algorithm.

Ensemble methods have been widely used for handling imbalanced problems. Classical exemples are the cost-sensitive-based AdaCost [8] and RareBoost [9], and the preprocessing-based SMOTE-Boost [10], RUSBoost [11], OverBagging [12] and UnderBagging [13], among others. These techniques are based on Static Selection (SS), which defines the same ensemble to be used in the labelling of all query instances of a problem. Dynamic Selection (DS) approaches, on the other hand, single out the classifiers better fit for labelling each query sample in particular, usually by evaluating the local competence of each classifier in the neighborhood of that sample. Since they do not take the entire dataset into account when performing the classification task, which may reduce the impact of the disproportion between the classes, DS techniques are seen as a promising alternative for imbalance learning [14].

A few recent works on ensemble-based approaches apply DS for dealing with imbalanced problems. The effect of using data preprocessing techniques combined with DS approaches for imbalance learning has been studied in [15]. In [16], a DS technique featuring cost-sensitive selection criteria was proposed for dealing with imbalanced data applied to the customer classification problem. The K-Nearest Oracles

Borderline-Imbalance (KNORA-BI) [17] technique was also proposed as an adaptation of the K-Nearest Oracles Eliminate (KNORA-E) Dynamic Ensemble Selection (DES) technique for imbalanced distributions. The Frienemy Indecision Region Dynamic Ensemble Selection++ (FIRE-DES++) framework [18], which yielded state-of-the-art results over several public imbalanced datasets, makes use of a data preprocessing technique for noise removal and a class-balanced neighborhood definition in order to better select the classifiers in imbalanced local regions.

In a previous work [19], an online local pool generation method was proposed for dealing with samples in local class overlap regions of the feature space. The method consists of iteratively generating hyperplanes on the fly using the instances in the neighborhood of these samples, and then selecting the most competent ones using a Dynamic Classifier Selection (DCS) technique. This online scheme yielded a statistically equivalent performance to several classification models, including four state-of-the-art DES techniques [19]. Though the online technique was not evaluated in the context of imbalance learning, the class-balanced local generation of the classifiers combined with their local evaluation and selection via DCS technique suggests that it may be somewhat robust to class imbalance.

So, in this work, we perform an evaluation of the online local pool generation technique from [19] with regards to imbalance learning. To that end, we compare its performance with a baseline pool generation technique using three DCS techniques over 64 two-class datasets, in order to evaluate whether generating the classifiers locally provide any advantage in imbalanced scenarios compared to only locally selecting them. The online scheme is also evaluated using the two additional modules included in the FIRE-DES++ framework, namely the DSEL pre-processing and the balanced region of competence definition, in order to assess whether these modules have a positive impact on the performance of the online method over imbalanced problems. Finally, the online local pool generation method and its variants including the additional modules are evaluated against seven state-of-the-art ensemble methods, four static and three DS-based, to assess its suitability for dealing with imbalance learning.

The rest of this work is organized as follows. In Section II, the online pool generation method is briefly introduced. The comparative analysis between the online method and its variants and the FIRE-DES++ is then performed in Section III. Lastly, the conclusions derived from this work are summarized and future works are suggested in Section IV.

II. Online local pool generation technique

The online local pool generation technique proposed in [19] is divided into two phases: an offline phase, in which the borderline training samples are identified, and an online phase, in which a pool of locally accurate classifiers is generated for labelling the query instances located on overlap regions of the feature space. The offline and the online phases of the method are described in Section II-A and Section II-B, respectively.

Fig. 1. Overview of the offline phase of the online local pool generation method. $\mathcal{T}$ is the training set and H is the set of KDN estimates, containing the KDN score (Equation 1) of all instances in $\mathcal{T}$.

A. Offline phase

In the offline phase, shown in Figure 1, the borderline samples in the training set $\mathcal{T}$ are singled out using the K-Disagreeing Neighborhood (KDN) measure, shown in Equation 1, where $\mathcal{T}$ is the training dataset, $\mathbf{x_i}$ and $\mathbf{x_j}$ are training samples, y_i and y_j are their respective labels and k_h is the neighborhood size. The KDN measure calculates the proportion of samples from a different class in the neighborhood of a given sample. This measure indicates the degree of class overlap in the region where the sample is located, and it is used in the online phase for deciding whether to generate the local pool on the fly or not.

$$\mathrm{KDN}(\mathbf{x_i}, \mathcal{T}, k_h) = \frac{|\{\mathbf{x_j} | \mathbf{x_j} \in \mathrm{KNN}(\mathbf{x_i}, \mathcal{T}, k_h) \wedge y_i \neq y_j\}|}{k_h} \tag{1}$$

Algorithm 1 Offline phase.

Input: $\mathcal{T}, k_h$ // Training dataset and KDN parameter
Output: H // Estimated KDN scores
1: **for** every $\mathbf{x_i}$ in $\mathcal{T}$ **do**
2: $H(i) \leftarrow KDN(\mathbf{x_i}, \mathcal{T}, k_h)$ // Calculate KDN score (Eq. 1)
3: **end for**
4: **return** H

Algorithm overview: The pseudocode of the offline phase of the online local pool generation scheme is shown in Algorithm 1. Its inputs are the training set $\mathcal{T}$ and the KDN neighborhood size k_h. From Step 1 to Step 3, the KDN score of each instance $\mathbf{x_i} \in \mathcal{T}$ is calculated and stored in H, which is then returned in Step 4.

B. Online phase

The online phase of the method, described in Figure 2, is divided in three steps: region of competence estimation, local pool generation and generalization. In the first step, the region of competence θ_q of the query sample $\mathbf{x_q}$ is first obtained using regular KNN with a neighborhood size of k_s over the training set $\mathcal{T}$.

The region of competence θ_q is then evaluated based on the KDN scores stored in H, which was obtained in the offline phase. If none of the sample's neighbors are borderline samples, that is, if their KDN score is zero, then the procedure goes directly to the last step, generalization, and the KNN classifier yields the output label ω_l of $\mathbf{x_q}$. If, however, any of the neighbors $\mathbf{x_i} \in \theta_q$ is a borderline sample, the region is identified as an overlap region and the local pool (LP) is

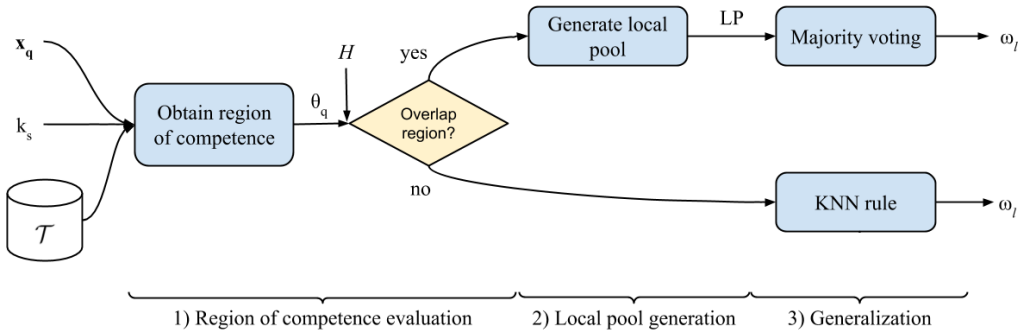

Fig. 2. Overview of the online phase of the online local pool generation method. $\mathcal{T}$ is the training set, $\mathbf{x_q}$ is the query sample, θ_q is its region of competence, k_s is the neighborhood size, H is the set of KDN estimates, LP is the local pool and ω_l is the output label of $\mathbf{x_q}$. In the online phase, the region of competence θ_q is first defined and evaluated based on the KDN scores in H, obtained in the offline phase. If θ_q does not contain borderline samples, that is, it is not an overlap region, the KNN rule is used to label $\mathbf{x_q}$ in the last step. Otherwise, the local pool is generated in the second step, and $\mathbf{x_q}$ is labelled via majority voting of the classifiers in LP in the third step.

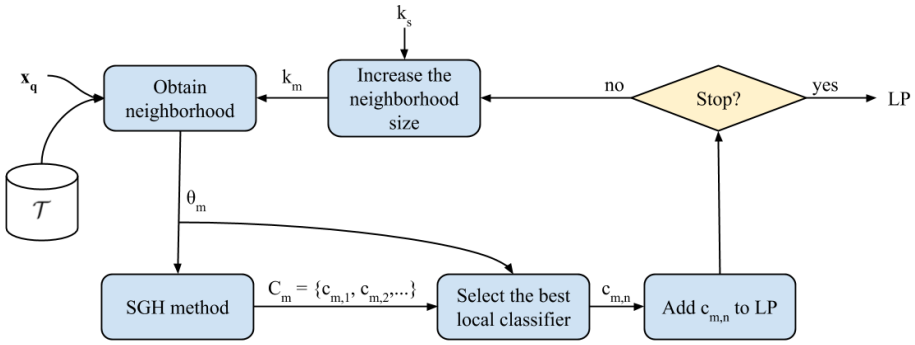

Fig. 3. Local pool generation step. $\mathcal{T}$ is the training step, $\mathbf{x_q}$ is the query sample and k_s is the size of the query sample's region of competence and LP is the final local pool, which is obtained iteratively. In the m-th iteration, the query sample's neighborhood θ_m of size k_m is obtained and used as input to the Self-Generating Hyperplanes (SGH) method [20], which yields the subpool C_m. The classifiers from C_m are then evaluated over θ_m using a DCS technique. The classifiers' notation refers a classifier $c_{m,k}$ as the k-th classifier from the m-th subpool (C_m). The most competent classifier $c_{m,n}$ in subpool C_m is then selected and added to the local pool LP. This process is then repeated until LP contains a pre-defined amount of locally accurate classifiers.

generated in the second step. The generation procedure of the local pool is explained next. Then, the generated locally accurate classifiers in LP are combined using the majority voting rule to obtain the output label ω_l in the generalization step.

Figure 3 shows the generation procedure of the local pool (LP). The LP is constructed iteratively, and in each iteration the most locally accurate classifier produced in that iteration is added to LP. In a given m-th iteration, the query sample's neighboring instances in the training set $\mathcal{T}$ are obtained using a nearest neighbors procedure with neighborhood size k_m, calculated based on the region of competence size k_s. For two-class problems, the K-Nearest Neighbors Equality (KNNE) [21] is used. The KNNE is a variation of the regular KNN that selects the same amount of samples from each of the classes in the problem. Figure 4 illustrates this procedure.

The query sample's neighborhood θ_m is then used as input to the Self-Generating Hyperplanes (SGH) method [20], a pool generation method that yields an Oracle [22] accuracy rate of 100% over the input dataset. The SGH method then returns a local subpool C_m that fully covers the class-balanced

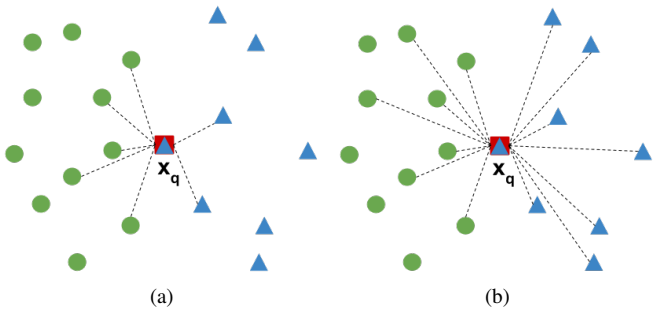

Fig. 4. Neighborhood definition of (a) KNN and (b) KNNE, with neighborhood size $K = 7$. In (a) the K closest examples to the query sample $\mathbf{x_q}$ are selected, while in (b) the K closest examples from each class are singled out.

neighborhood θ_m. That is, the presence of at least one competent classifier $c_{m,k} \in C_m$ for each instance in θ_m is guaranteed. The indexes in the classifiers' notation indicates that the classifier $c_{m,k}$ is the k-th classifier from the m-th subpool (C_m). Then, the most competent classifier $c_{m,n}$ from C_m in the region delimited by the neighborhood θ_q is

Algorithm 2 Online phase.

Input: $\mathbf{x_q}, \mathcal{T}, H$ // Query sample, training set and KDN estimates
Input: k_s, M // Neighborhood size and pool size of local pool (LP)
Output: ω_l // Output label of $\mathbf{x_q}$

1: $\theta_q \leftarrow obtainRoC(\mathbf{x_q}, k_s, \mathcal{T})$ // Obtain the query instance's region of competence
2: **if** $\{\exists \mathbf{x_i} \in \theta_q | H(i) > 0\}$ **then** } 1) Region of competence evaluation
3: $LP \leftarrow \{\}$ // Local pool initially empty
4: **for** every m in $\{1, 2, ..., M\}$ **do**
5: $k_m \leftarrow k_s + 2 \times (m-1)$ // Increase neighborhood size by 2
6: $\theta_m \leftarrow obtainNeighborhood(\mathbf{x_q}, k_m, \mathcal{T})$ // Obtain neighborhood of $\mathbf{x_q}$
7: $C_m \leftarrow generatePool(\theta_m)$ // Generate local subpool C_m
8: $c_{m,n} \leftarrow selectClassifier(\mathbf{x_q}, \theta_m, C_m)$ // Select best classifier in C_m
9: $LP \leftarrow LP \cup \{c_{m,n}\}$ // Add $c_{m,n}$ to LP
10: **end for** } 2) Local pool generation
11: $\omega_l \leftarrow majorityVoting(\mathbf{x_q}, LP)$ // Label $\mathbf{x_i}$ with majority voting on LP
12: **else**
13: $\omega_l \leftarrow KNN(\mathbf{x_q}, k_s, \mathcal{T})$ // Label query sample using KNN rule } 3) Generalization
14: **end if**
15: **return** ω_l

selected by a DCS technique and added to the local pool. The same procedure is performed in iteration $m+1$ with the neighborhood size k_{m+1} increased by 2. This process is then repeated until the local pool contains a pre-defined amount of locally accurate classifiers.

Algorithm overview: The online phase of the technique is described in more detail in Algorithm 2. Its inputs are the query sample $\mathbf{x_q}$, training set $\mathcal{T}$, the set of KDN scores H, the region of competence size k_s and the local pool size M.

Step 1 to Step 2 represent the region of competence evaluation step of the online phase (Figure 2). In Step 1, the query sample's region of competence θ_q is obtained by selecting the k_s closest samples to $\mathbf{x_q}$ in the training set. The region of competence is then evaluated in Step 2. If all instances in θ_q are not borderline samples, that is, their KDN value is zero, the method goes straight to Step 13 (generalization step of Figure 2) and the query sample's output label ω_l is obtained using the KNN rule with parameter k_s and returned in Step 15.

However, if there is one instance $\mathbf{x_i}$ from θ_q whose KDN estimate $H(i)$ is above zero, the region is considered an overlap one and the method proceeds to Step 3. Each classifier in the local pool (LP) is obtained in the loop that iterates M times, so Step 4 to Step 10 describe the local pool generation procedure from Figure 3. In each iteration, the neighborhood size k_m is calculated in Step 5. Then, the query sample's neighborhood θ_m is obtained using a nearest neighbors method in Step 6. For two class problems, the KNNE is used in this step.

The subpool C_m is then generated in Step 7 using the SGH method with θ_m as training set. In Step 8, a DCS technique is then used to select the most competent classifier $c_{m,n}$ in C_m. The classifier $c_{m,n}$ is added to LP in Step 9, and then the loop continues until the local pool is complete. Finally, the query sample's label ω_l is obtained using majority voting over the locally accurate classifiers in LP in Step 11 and it is then returned in Step 15.

III. Experiments

Experiments were conducted over the 64 two-class datasets presented in Table I from the Knowledge Extraction based on Evolutionary Learning (KEEL) repository [23]. Each dataset was evaluated using a stratified 5-fold cross validation, one fold for test and the remaining for training, with the partitions already provided in the KEEL website. Since the number of samples in each dataset is quite small, the training set was also used as the dynamic selection dataset (DSEL) [14] in the experiments for all evaluated DS techniques, as in [18]. The DSEL is a set of labelled samples used for region of competence definition in DS techniques.

The measures used to evaluate the performance of the techniques were the mean accuracy rate and three frequently used measures for imbalance learning: the area under the Receiver Operation Characteristic (ROC) curve (AUC) [24], the F-measure [25] and the Geometric Mean (G-mean) [26]. The F-measure and the G-mean are described in Equation 2 and Equation 3, respectively, in which P is the number of test samples from the minority class (positive class), N is the number of test samples from the majority class (negative class), TP is the number of true positives, TN is the number of true negatives, TN is the number of true negatives and FP is the number of false positives.

$$F\text{-}measure = 2 \times \frac{TP}{P + TP + FP} \tag{2}$$

$$G\text{-}mean = \sqrt{\frac{TP}{P} \times \frac{TN}{N}} \tag{3}$$

The experiments were performed using the libraries scikit-learn [27], imbalanced-learn [28] and DESLib [29], which provide implementations of several classification models in the Python 3.5 language.

The experimental analysis is divided into two parts. In Section III-A, the online scheme and its variants are evaluated and compared to the baseline technique using three DCS techniques. In Section III-B, the online technique and

TABLE I
MAIN CHARACTERISTICS OF THE DATASETS USED IN THE EXPERIMENTS.

Dataset	# Feats.	# Samples	IR
glass1	9	214	1.82
ecoli0vs1	7	220	1.86
wisconsin	9	683	1.86
pima	8	768	1.87
iris0	4	150	2.00
glass0	9	214	2.06
yeast1	8	1484	2.46
haberman	3	306	2.78
vehicle2	18	846	2.88
vehicle1	18	846	2.90
vehicle3	18	846	2.99
glass0123vs456	9	214	3.20
vehicle0	18	846	3.25
ecoli1	7	336	3.36
new-thyroid1	5	215	5.14
new-thyroid2	5	215	5.14
ecoli2	7	336	5.46
segment0	19	2308	6.00
glass6	9	214	6.38
yeast3	8	1484	8.10
ecoli3	7	336	8.60
page-blocks0	10	5472	8.79
ecoli-0-3-4vs5	7	200	9.00
yeast-2vs4	8	514	9.08
ecoli-0-6-7vs3-5	7	202	9.09
ecoli-0-2-3-4vs5	7	222	9.10
yeast-0-3-5-9vs7-8	8	506	9.12
glass-0-1-5vs2	9	172	9.12
yeast-0-2-5-7-9vs3-6-8	8	1004	9.14
yeast-0-2-5-6vs3-7-8-9	8	1004	9.14
ecoli-0-4-6vs5	6	203	9.15
ecoli-0-1vs2-3-5	7	224	9.17
ecoli-0-2-6-7vs3-5	7	224	9.18
glass-0-4vs5	9	92	9.22
ecoli-0-3-4-6vs5	7	205	9.25
ecoli-0-3-4-7vs5-6	7	257	9.28
yeast-05679vs4	8	528	9.35
vowel0	13	988	9.98
ecoli-0-6-7vs5	6	220	10.00
glass-016vs2	9	192	10.29
ecoli-0-1-4-7vs2-3-5-6	7	336	10.59
led7digit-0-2-4-5-6-7-8-9vs1	7	443	10.97
glass-0-6vs5	9	205	11.00
ecoli-0-1vs5	6	240	11.00
glass-0-1-4-6vs2	9	205	11.06
glass2	9	214	11.59
ecoli-0-1-4-7vs5-6	6	332	12.28
ecoli-0-1-4-6vs5	6	280	13.00
cleveland-0vs4	13	177	12.62
shuttle-c0vsc4	9	1829	13.87
yeast-1vs7	7	459	14.30
glass4	9	214	15.47
ecoli4	7	336	15.80
page-blocks-13vs4	10	472	15.86
glass-0-1-6_vs_5	9	184	19.44
shuttle-c2-vs-c4	9	129	20.50
yeast-1458vs7	8	693	22.10
glass5	9	214	22.78
yeast-2vs8	8	482	23.10
yeast4	8	1484	28.10
yeast-1289vs7	8	947	30.57
yeast5	8	1484	32.73
ecoli-0137vs26	7	281	39.14
yeast6	8	1484	41.40

its variants are compared to seven state-of-the-art ensemble methods, four of which being static ensembles and three being dynamic ensemble selection frameworks.

A. Comparison with baseline technique

In this section, the online local pool generation technique is evaluated against the baseline technique, Bagging [30], with a pool size of 100 classifiers, with the Perceptron as the base classifier (learning rate $\alpha = 0.001$ and number of iterations $n_{iter} = 100$), over three DCS techniques. The DCS techniques used in the experiments were the Overall Local Accuracy (OLA) [31], Local Class Accuracy (LCA) [31] and Multiple Classifier Behavior (MCB) [32], all with a region of competence size of $K = 7$. No Dynamic Ensemble Selection (DES) techniques were evaluated since the generation procedure of the online scheme is not fit for selecting more than one classifier at a time [19].

The inclusion of the DSEL preprocessing and the balanced region of competence definition from the FIRE-DES++, namely using the Edited Nearest Neighbors (ENN) [33] with $K = 3$, and the KNNE with $K = 7$, respectively, to the online scheme is also evaluated in order to assess its impact on the method's performance. Thus, the online local pool generation technique is evaluated in four versions: the original one (referenced as OLP), which defines the region of competence using KNN and generates the classifiers using KNNE, the OLP+KNNE, which uses KNNE for region of competence definition *and* for pool generation, the OLP+ENN, which is the original OLP with the DSEL set pre-processed using ENN, and the OLP+KNNE+ENN, which includes the two modifications previously mentioned. The pool size of all versions of the online technique was set to $M = 5$.

Table II shows the mean performance of the online schemes and the baseline technique using the three evaluated DCS techniques. It can be observed that the OLP and its variants obtained a greater overall accuracy rate in comparison with Bagging using OLA and MCB. They also statistically outperformed the baseline technique in this scenario according to a Wilcoxon signed-rank test with a significance level of $\alpha = 0.05$.

In terms of AUC and G-mean, it can be observed that the baseline technique obtained a greater performance overall using OLA and MCB, though its performance was statistically similar to all versions of the online scheme. Using LCA, though, all versions of the OLP statistically outperformed Bagging considering these performance measures.

As for the F-measure, the OLP+KNNE+ENN obtained the highest mean performance using OLA and MCB, while using LCA the OLP+KNNE yielded the greater overall F-measure. The two versions of the OLP coupled with the ENN also obtained a statistically superior performance to Bagging using MCB. As in the previous case, all versions of the online scheme obtained a statistically similar performance to Bagging using OLA, and a significantly superior performance using LCA.

Thus, the results suggest that based on the same local evaluation by the DCS techniques, the approach of locally generating the classifiers on the fly is at least as effective for imbalance learning as dynamically selecting the classifiers from a globally generated pool obtained offline, and at most a more advantageous approach (as in the case of using LCA).

It can also be observed from Table II that the inclusion of a balanced region of competence and a DSEL pre-

TABLE II

MEAN (A) ACCURACY RATE, (B) AUC, (C) F-MEASURE AND (D) G-MEAN OF THE EVALUATED TECHNIQUES. BEST RESULTS ARE IN BOLD. THE ROWS *p-value* SHOW THE RESULTING P-VALUE OF A WILCOXON SIGNED-RANK TEST WITH $\alpha = 0.05$ BETWEEN THE PERFORMANCE OF THE INDICATED TECHNIQUE (BAGGING OR OLP) AND THE REMAINING METHODS. THE P-VALUES BELOW α ARE UNDERLINED.

(a)

Technique		Bagging	OLP	OLP+KNNE	OLP+ENN	OLP+KNNE+ENN
OLA	Avg.	92.81	93.95	**93.98**	93.62	93.66
	p-value (Bag)	n/a	0.000	0.000	0.000	0.000
	p-value (OLP)	-	n/a	0.534	0.030	0.076
LCA	Avg.	92.55	92.63	**92.90**	92.18	92.47
	p-value (Bag)	n/a	0.802	0.273	0.169	0.692
	p-value (OLP)	-	n/a	0.001	0.001	0.553
MCB	Avg.	92.95	94.00	**94.11**	93.84	93.84
	p-value (Bag)	n/a	0.000	0.000	0.000	0.000
	p-value (OLP)	-	n/a	0.012	0.549	0.627

(b)

Technique		Bagging	OLP	OLP+KNNE	OLP+ENN	OLP+KNNE+ENN
OLA	Avg.	**0.819**	0.808	0.809	0.815	0.817
	p-value (Bag)	n/a	0.470	0.583	0.849	0.700
	p-value (OLP)	-	n/a	0.125	0.012	0.006
LCA	Avg.	0.771	0.797	**0.798**	0.795	0.794
	p-value (Bag)	n/a	0.000	0.000	0.003	0.004
	p-value (OLP)	-	n/a	0.193	0.495	0.494
MCB	Avg.	**0.823**	0.808	0.810	0.821	**0.823**
	p-value (Bag)	n/a	0.174	0.405	0.553	0.354
	p-value (OLP)	-	n/a	0.141	0.000	0.000

(c)

Technique		Bagging	OLP	OLP+KNNE	OLP+ENN	OLP+KNNE+ENN
OLA	Avg.	0.672	0.673	0.675	0.682	**0.684**
	p-value (Bag)	n/a	0.241	0.209	0.104	0.057
	p-value (OLP)	-	n/a	0.301	0.091	0.044
LCA	Avg.	0.606	0.634	**0.640**	0.630	0.633
	p-value (Bag)	n/a	0.011	0.003	0.025	0.010
	p-value (OLP)	-	n/a	0.002	0.747	0.377
MCB	Avg.	0.680	0.678	0.681	0.693	**0.694**
	p-value (Bag)	n/a	0.093	0.051	0.005	0.006
	p-value (OLP)	-	n/a	0.144	0.002	0.006

(d)

Technique		Bagging	OLP	OLP+KNNE	OLP+ENN	OLP+KNNE+ENN
OLA	Avg.	**0.773**	0.738	0.739	0.753	0.755
	p-value (Bag)	n/a	0.297	0.416	0.935	0.858
	p-value (OLP)	-	n/a	0.125	0.006	0.002
LCA	Avg.	0.680	**0.732**	0.730	0.730	0.725
	p-value (Bag)	n/a	0.000	0.000	0.001	0.002
	p-value (OLP)	-	n/a	0.213	0.427	0.543
MCB	Avg.	**0.781**	0.738	0.740	0.759	0.762
	p-value (Bag)	n/a	0.161	0.399	0.540	0.353
	p-value (OLP)	-	n/a	0.226	0.000	0.000

processing improved the overall performance of the original online scheme. More specifically, the addition of the KNNE alone improved significantly the OLP using LCA and MCB in terms of accuracy, while using the ENN yielded a significant improvement using MCB in terms of AUC and F-measure and using OLA and MCB in terms of G-mean.

B. Comparison with the state-of-the-art

In this section, we compare the performance of the online local pool scheme and its variants evaluated in the previous section with seven state-of-the-art ensemble techniques. The static ensembles evaluated were AdaBoost [34], RUSBoost [11], Random Forest (RF) [35] and Balanced Random Forest (BalancedRF) [36], all with a pool of 100 decision trees and with the hyperparameters set to the values suggested in [37]. The DES techniques evaluated were the FIRE-KNORA-E++ (FKNE++) [18], the META-DES [38] and the Randomized Reference Classifier (RRC) [39], also with a pool size of 100 classifiers, but with Perceptrons as base-classifiers. For simplicity, the online techniques used in the comparative analysis were coupled with MCB, since they obtained a greater overall performance in comparison with the other two DCS techniques evaluated in the previous section.

Table III shows the mean performance of the evaluated techniques with regards to accuracy rate, AUC, F-measure and G-mean. It can be observed that the four versions of the OLP obtained the greatest mean accuracy rates save for the RF. They also yielded the highest average ranks after the RF according to a Friedman test.

In terms of AUC and F-measure, the BalancedRF obtained the highest average rank, while the OLP+KNNE+ENN and the OLP+ENN yielded the second and third highest mean ranks, respectively. The three alternate versions of the OLP also obtained the highest mean ranks in terms of F-measure, with the META-DES and the OLP right after in the ranking.

Since the p-values indicated in Table III were below the confidence level of $\alpha = 0.05$, a post-hoc Bonferroni test with the same α was performed and the results are shown in the critical difference diagrams in Figure 5. It can be observed that the OLP and its variants were significantly superior to FKNE++, RUSBoost and BalancedRF and statistically equivalent to the remaining state-of-the-art ensemble methods in terms of accuracy.

In terms of AUC and G-mean, the BalancedRF significantly outperformed all evaluated techniques. The two versions of the online scheme that use ENN were significantly superior to RF only, while the other two had a similar performance to all techniques but the BalancedRF. In terms of F-measure, the OLP+ENN and the OLP+KNNE+ENN significantly outperformed RUSBoost, BalancedRF, AdaBoost and FKNE++, while the other two versions (OLP and OLP+KNNE) were significantly superior to RUSBoost only.

Thus, the results from Figure 5 suggest that the online local pool technique yields a statistically similar performance to most of the evaluated state-of-the-art ensemble methods with regards to imbalance learning. The addition of the ENN and the KNNE to the online scheme also provided enough improvement to outperform few state-of-the-art techniques.

IV. CONCLUSIONS AND FUTURE WORK

In this work, we conducted an evaluation of the online local pool generation method from [19] with regards to imbalance learning, with the aim of assessing its suitability for handling imbalanced problems. A total of 64 datasets of varying imbalance degrees and four performance measures were used in the experiments. The comparative analysis featured three

TABLE III

MEAN PERFORMANCE OF THE EVALUATED STATE-OF-THE-ART ENSEMBLE METHODS, THE ONLINE LOCAL POOL SCHEME AND ITS VARIANTS, THE LATTER OF WHICH COUPLED WITH MCB. BEST RESULTS ARE IN BOLD. THE ROW *Avg. rank* INDICATES THE AVERAGE RANK OF EACH TECHNIQUE ACCORDING TO A FRIEDMAN TEST OVER THE EVALUATED TECHNIQUES, AND THE *p-value* ROW INDICATES THE RESULTING P-VALUE OF THE TEST.

Measure		AdaBoost	RUSBoost	RF	BalancedRF	FKNE++	META-DES	RRC	OLP	OLP+KNNE	OLP+ENN	OLP+KNNE+ENN
Accuracy	Avg.	93.37	92.57	**94.30**	86.87	92.79	93.71	93.77	94.00	94.11	93.84	93.84
	Avg. rank	6.492	8.367	**3.968**	9.757	7.640	5.414	5.296	4.804	4.414	4.953	4.890
	p-value											9.635×10^{-39}
AUC	Avg.	0.805	0.815	0.797	**0.880**	0.829	0.825	0.811	0.808	0.810	0.821	0.823
	Avg. rank	7.015	6.843	7.359	**2.492**	6.054	5.656	6.625	6.828	6.585	5.335	5.203
	p-value											6.471×10^{-20}
F-measure	Avg.	0.659	0.647	0.661	0.624	0.682	**0.700**	0.674	0.678	0.681	0.693	0.694
	Avg. rank	6.835	7.632	5.945	7.179	6.6719	5.304	6.281	5.703	5.265	**4.585**	4.593
	p-value											5.856×10^{-10}
G-mean	Avg.	0.740	0.760	0.711	**0.876**	0.791	0.777	0.745	0.738	0.740	0.759	0.762
	Avg. rank	7.101	6.656	7.679	**2.304**	6.00	5.664	6.687	6.937	6.671	5.226	5.070
	p-value											6.909×10^{-24}

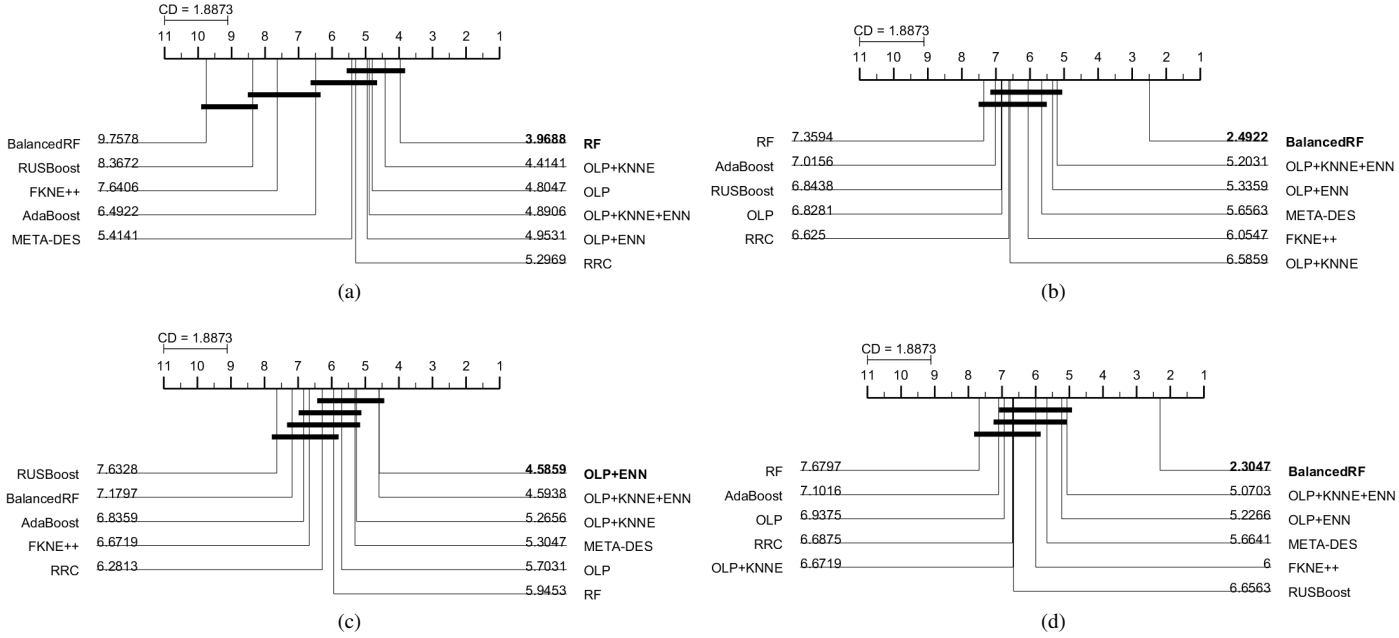

Fig. 5. Critical difference diagram of Bonferroni-Dunn post-hoc test on the results from Table III in terms of (a) accuracy rate, (b) AUC, (c) F-measure and (d) G-mean. The critical value (CD) was computed with a confidence level pf $\alpha = 0.05$.

variations of the online scheme incorporating the data pre-processing and the balanced region of competence definition from the FIRE-DES++ [18] technique, in order to evaluate the impact of these additional steps on the performance of the method. The online scheme was also evaluated against four state-of-the-art static ensembles and three state-of-the-art dynamic selection frameworks.

Experimental results show that the online pool generation method and its alternate approaches performed generally better in terms of accuracy in comparison with the baseline technique, Bagging, using three DCS techniques. In terms of AUC, F-measure and G-mean, the online scheme was mostly statistically equivalent to Bagging when using OLA and MCB, and significantly superior to it when using LCA, which suggests generating the classifiers locally is an advantageous approach for dealing with imbalanced problems. Moreover, the inclusion of a balanced region of competence definition to the online scheme improved its accuracy rate, while the addition of a pre-processing step using the ENN improved the results in terms of AUC, F-measure and G-mean.These results indicate that the noise removal provided by the ENN and the use of balanced local regions is quite advantageous for local learning algorithms when dealing with imbalanced problems.

The comparative analysis with the state-of-the-art showed that the online local pool technique obtained a statistically similar performance to most evaluated state-of-the-art techniques, which suggests it is suitable for dealing with imbalanced problems. Furthermore, the online technique coupled with the additional modules of the FIRE-DES++ yielded a significantly superior performance in comparison with a few of the evaluated techniques.

Future works may include the design of a dynamic local

noise removal or prototype generation procedure for better dealing with local imbalanced regions during the generation and the selection of the local classifiers.

ACKNOWLEDGMENTS

The authors would like to thank the Brazilian agencies CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) and FACEPE (Fundação de Amparo à Ciência e Tecnologia de Pernambuco) and the Canadian agency NSERC (Natural Sciences and Engineering Research Council of Canada).

REFERENCES

[1] M. A. Mazurowski, P. A. Habas, J. M. Zurada, J. Y. Lo, J. A. Baker, and G. D. Tourassi, "Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance," *Neural networks*, vol. 21, no. 2-3, pp. 427–436, 2008.

[2] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web*, vol. 16, no. 4, pp. 449–475, 2013.

[3] L. Piras and G. Giacinto, "Synthetic pattern generation for imbalanced learning in image retrieval," *Pattern Recognition Letters*, vol. 33, no. 16, pp. 2198–2205, 2012.

[4] Z. Zheng, X. Wu, and R. Srihari, "Feature selection for text categorization on imbalanced data," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 80–89, 2004.

[5] R. C. Prati, G. E. Batista, and D. F. Silva, "Class imbalance revisited: a new experimental setup to assess the performance of treatment methods," *Knowledge and Information Systems*, vol. 45, no. 1, pp. 247–270, 2015.

[6] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463–484, 2012.

[7] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning from Imbalanced Data Sets*. Springer International Publishing, 2018.

[8] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan, "Adacost: misclassification cost-sensitive boosting," in *6th International Conference on Machine Learning*, 1999, pp. 97–105.

[9] M. V. Joshi, V. Kumar, and R. C. Agarwal, "Evaluating boosting algorithms to classify rare classes: Comparison and improvements," in *Proceedings IEEE International Conference on Data Mining*. IEEE, 2001, pp. 257–264.

[10] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "Smoteboost: Improving prediction of the minority class in boosting," in *European conference on principles of data mining and knowledge discovery*. Springer, 2003, pp. 107–119.

[11] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "Rusboost: A hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 40, no. 1, pp. 185–197, 2010.

[12] S. Wang and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," in *IEEE Symposium on Computational Intelligence and Data Mining*. IEEE, 2009, pp. 324–331.

[13] R. Barandela, R. M. Valdovinos, and J. S. Sánchez, "New applications of ensembles of classifiers," *Pattern Analysis & Applications*, vol. 6, no. 3, pp. 245–256, 2003.

[14] R. M. Cruz, R. Sabourin, and G. D. Cavalcanti, "Dynamic classifier selection: Recent advances and perspectives," *Information Fusion*, vol. 41, pp. 195–216, 2018.

[15] A. Roy, R. M. Cruz, R. Sabourin, and G. D. Cavalcanti, "A study on combining dynamic selection and data preprocessing for imbalance learning," *Neurocomputing*, vol. 286, pp. 179–192, 2018.

[16] J. Xiao, L. Xie, C. He, and X. Jiang, "Dynamic classifier ensemble model for customer classification with imbalanced class distribution," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3668–3675, 2012.

[17] D. V. R. Oliveira, G. D. C. Cavalcanti, T. N. Porpino, R. M. O. Cruz, and R. Sabourin, "K-nearest oracles borderline dynamic classifier ensemble selection," in *International Joint Conference on Neural Networks (IJCNN)*, July 2018, pp. 1–8.

[18] R. M. Cruz, D. V. Oliveira, G. D. Cavalcanti, and R. Sabourin, "FIRE-DES++: Enhanced online pruning of base classifiers for dynamic ensemble selection," *Pattern Recognition*, vol. 85, pp. 149–160, 2019.

[19] M. A. Souza, G. D. Cavalcanti, R. M. Cruz, and R. Sabourin, "Online local pool generation for dynamic classifier selection," *Pattern Recognition*, vol. 85, pp. 132–148, 2019.

[20] ——, "On the characterization of the oracle for dynamic classifier selection," in *International Joint Conference on Neural Networks (IJCNN), 2017*. IEEE, 2017, pp. 332–339.

[21] B. Sierra, E. Lazkano, I. Irigoien, E. Jauregi, and I. Mendialdua, "K nearest neighbor equality: Giving equal chance to all existing classes," *Information Sciences*, vol. 181, no. 23, pp. 5158–5168, 2011.

[22] L. I. Kuncheva, "A theoretical study on six classifier fusion strategies," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 281–286, 2002.

[23] J. Alcalá, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2-3, pp. 255–287, 2011.

[24] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[25] C. van Rijsbergen, *Information Retrieval*. Butterworth, 1979.

[26] M. Kubat, S. Matwin *et al.*, "Addressing the curse of imbalanced training sets: one-sided selection," in *International Conference on Machine Learning*, vol. 97. Nashville, USA, 1997, pp. 179–186.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[28] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017.

[29] R. M. Cruz, L. G. Hafemann, R. Sabourin, and G. D. Cavalcanti, "Deslib: A dynamic ensemble selection library in python," *arXiv preprint arXiv:1802.04967*, 2018.

[30] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[31] K. Woods, W. P. Kegelmeyer Jr, and K. Bowyer, "Combination of multiple classifiers using local accuracy estimates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 405–410, 1997.

[32] G. Giacinto, F. Roli, and G. Fumera, "Selection of classifiers based on multiple classifier behaviour," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition and Structural and Syntactic Pattern Recognition*. Springer-Verlag, 2000, pp. 87–93.

[33] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Transactions on Systems, Man, and Cybernetics*, no. 3, pp. 408–421, 1972.

[34] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, pp. 119–139, 1997.

[35] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[36] C. Chen, A. Liaw, and L. Breiman, "Using random forest to learn imbalanced data," *University of California, Berkeley*, vol. 110, pp. 1–12, 2004.

[37] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems?" *Journal of Machine Learning Research*, vol. 15, pp. 3133–3181, 2014.

[38] R. M. O. Cruz, R. Sabourin, G. D. C. Cavalcanti, and T. I. Ren, "META-DES: A dynamic ensemble selection framework using meta-learning," *Pattern Recognition*, vol. 48, no. 5, pp. 1925–1935, 2015.

[39] T. Woloszynski and M. Kurzynski, "A probabilistic model of classifier competence for dynamic ensemble selection," *Pattern Recognition*, vol. 44, no. 10, pp. 2656–2668, 2011.