On dynamic ensemble selection and data preprocessing for multi-class imbalance learning

Rafael M. O. Cruz
Ecole de Technologie Supérieure
Montreal, Canada
rafaelmenelau@gmail.com

Robert Sabourin
Ecole de Technologie Supérieure
Montreal, Canada
Robert.Sabourin@etsmtl.ca

George D. C. Cavalcanti
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brazil
gdcc@cin.ufpe.br

***Abstract*—Class-imbalance refers to classification problems in which many more instances are available for certain classes than for others. Such imbalanced datasets require special attention because traditional classifiers generally favor the majority class which has a large number of instances. Ensemble of classifiers have been reported to yield promising results. However, the majority of ensemble methods applied too imbalanced learning are static ones. Moreover, they only deal with binary imbalanced problems. Hence, this paper presents an empirical analysis of dynamic selection techniques and data preprocessing methods for dealing with multi-class imbalanced problems. We considered five variations of preprocessing methods and four dynamic selection methods. Our experiments conducted on 26 multi-class imbalanced problems show that the dynamic ensemble improves the F-measure and the G-mean as compared to the static ensemble. Moreover, data preprocessing plays an important role in such cases.**

***Index Terms*—Imbalanced learning, multi-class imbalanced, ensemble of classifiers, dynamic classifier selection, data preprocessing**

I. Introduction

Class-imbalance [1] refers to classification problems in which many more instances are available for certain classes than for others. Particularly, in a two-class scenario, one class contains the majority of instances (the *majority class*), while the other (the *minority class*) contains fewer instances. Imbalanced datasets may originate from real life problems including the detection of fraudulent bank account transactions, telephone calls, biomedical diagnosis, image retrieval and so on.

One of the biggest challenges in imbalanced learning is dealing with multi-class imbalanced problems [2]. Multi-class imbalanced classification is not as well developed as the binary case, with only a few papers handling this issue [3, 4, 5]. It is also considered as a more complicated problem, since the relation among the classes is no longer obvious. For instance, one class may be the majority one when compared to some classes, and minority when compared to others. Moreover, we may easily lose performance on one class while trying to gain it on another [4].

One way of dealing with imbalanced distributions is to use ensemble learning. As shown in [6], a diverse ensemble can better cope with imbalanced distribution. In particular, Dynamic selection (DS) techniques is seen as an alternative to deal with multi-class imbalance as it explores the local competence of each base classifier according to each new query sample [7, 2, 8]. Only the base classifiers that attained a certain competence level, in the given local region, are selected to predict the label of the query sample.

A key factor in dynamic selection is the estimation of the classifiers' competences according to each test sample. Usually the estimation of the classifiers competences are based on a set of labeled samples, called the dynamic selection dataset (DSEL). However, As reported in [9], dynamic selection performance is very sensitive to the distribution of samples in DSEL. If the distribution of DSEL itself becomes imbalanced, then there is a high probability that the region of competence for a test instance will become lopsided. Thus, the dynamic selection algorithms might end up biased towards selecting base classifiers that are experts for the majority class. With this in mind, we propose the use of data preprocessing methods for training a pool of classifiers as well as balancing the class distribution in DSEL for the DS techniques.

Hence, in this paper, we perform a study on the application of dynamic selection techniques and data preprocessing for handling with multi-class imbalance. Five data preprocessing techniques and four DS techniques as well as static ensemble combination are considered in our experimental analysis. Experiments are conducted using 26 multi-class imbalanced datasets with varying degrees of class imbalance. The following research questions are studied in this paper:

1) Does data preprocessing play an important role in the performance of dynamic selection techniques?
2) Which data preprocessing technique is better suited for dynamic and static ensemble combination?
3) Do dynamic ensembles present better performance than static ensembles?

This paper is organized as follows: Section II presents the related works on dynamic selection and describes the DCS and DES methods considered in this analysis. Data preprocessing techniques for imbalance are presented in Section III. Experiments are conducted in Section IV. Conclusion and future works are presented in the last section.

II. Dynamic selection

A dynamic selection (DS) enables the selection of one or more base classifiers from a pool, given a test instance. This is based on the assumption that each base classifier is

an expert in a different local region in the feature space. Therefore, the most competent classifiers should be selected in classifying a new instance. The notion of competence is used in DS as a way of selecting, from a pool of classifiers, the best classifiers to classify a given test instance. Usually, the competence of a base classifier is estimated based on a small region in the feature space surrounding a given test instance, called the *region of competence*. This region is formed using the k-nearest neighbors (KNN) technique, with a set of labeled samples, which can be either the training or validation set. This set is called the Dynamic Selection dataset (DSEL) [7]. To establish the competence, given a test instance and the DSEL, the literature reports a number of measures classified into individual-based and group-based measures [7]. Implementation of several DS techniques can be found on GitHub: https://github.com/Menelau/DESlib [10].

Among the categories, we focus on the individual-based measures, which consider individual base classifier accuracy for the region of competence. However, the competency measures are calculated differently by different methods in this category. For example, we consider methods in which the competency is measured by pure accuracy [11], by ranking of classifiers [12] or using oracle information [13].

Instead of grouping DS strategies by competence measure, we may also group them by selection methodology. Currently, there are two kinds of selection strategies: dynamic classifier selection (DCS) and dynamic ensemble selection (DES). DCS selects a single classifier for a test instance, whereas DES selects an ensemble of classifiers (EoC) to classify a test instance. Both these strategies have been studied in recent years, and some papers are available examining them [14, 15]. In this paper, we evaluate two DCS and two DES strategies:

- **The Modified Classifier Rank (RANK)** [12] is a DCS method that exploits ranks of individual classifiers in the pool for each test instance. The rank of a classifier is based on its local accuracy within a neighborhood of the test instance. More formally, given a test instance assigned to class C_i by a classifier, the ranking of the classifier is estimated as the number of consecutive nearest neighbors assigned to class C_i that have been correctly labeled. The most locally accurate classifier has the highest rank and is selected for classification.
- **The Local Class Accuracy (LCA)** [11] estimates the classifier accuracy in a local region around the given test instance and then uses the most locally accurate classifier to classify the test instance. The local accuracy is estimated for each base classifier as the percentage of correct classifications within the local region, but considering only those examples where the classifier predicted the same class as the one it gave for the test instance. The classifier presenting the highest local accuracy is used for the classification of the query sample.
- **The KNORA-Eliminate technique (KNE)** [13] explores the concept of Oracle, which is the upper limit of a DCS technique. Given the region of competence θ_j, only the classifiers that correctly recognize all samples belonging to the region of competence are selected. In other words, all classifiers that achieved a 100% accuracy in this region (i.e., that are local Oracles) are selected to compose the ensemble of classifiers. Then, the decisions of the selected base classifiers are aggregated using the majority voting rule. If no base classifier is selected, the size of the region of competence is reduced, and the search for the competent classifiers is restarted.
- **The KNORA-Union technique (KNU)** [13] selects all classifiers that are able to correctly recognize at least one sample in the region of competence. This method also considers that a base classifier can participate more than once in the voting scheme when it correctly classifies more than one instance in the region of competence. The number of votes of a given base classifier c_i is equal to the number of samples in the region of competence, for which it predicted the correct label. For instance, if a given base classifier c_i predicts the correct label for three samples belonging to the region of competence, it gains three votes for the majority voting scheme. The votes collected by all base classifiers are aggregated to obtain the ensemble decision.

These DS techniques are based on different criterion to estimate the local competence of the base classifiers. For example, while both the RANK and the LCA are DCS strategies, the former measures the competence based on ranking, and the latter based on classifier accuracy. On the other hand, the two DES strategies (KNE and KNU) are based on Oracle information. These techniques were selected as they were among the best performing DS methods according to the experimental analysis conducted in [7].

Nevertheless, a crucial aspect in the performance of the dynamic selection techniques is the distribution of the dynamic selection dataset (DSEL), as the local competence of the base classifiers are estimated based on this set. Hence, preprocessing techniques can really benefit DS techniques as they can be employed to edit the distribution of DSEL, prior to performing dynamic selection.

III. Data preprocessing

Changing the distribution of the training data to compensate for poor representativeness of the minority class is an effective solution for imbalanced problems, and a plethora of methods are available in this regards. Branco et al. [16] divided such methods into three categories, namely, stratified sampling, synthesizing new data, and combinations of the two previous methods. While the complete taxonomy is available in [16], we will center our attention on the methods that have been used together with ensemble learning [6].

One important category is under-sampling, which removes instances from the majority class to balance the distribution. Random under-sampling (RUS) [17] is one such method. RUS has been coupled with boosting (RUSBoost) [18] and with Bagging [17]. A major drawback of RUS is that it can discard potentially useful data which can be a problem when using dynamic selection approaches.

The other strategy is the generation of new synthetic data. Synthesizing new instances has several known advantages [19], and a wide number of proposals are available for building new synthetic examples. In this context, a famous method that uses interpolation to generate new instances is SMOTE [19]. SMOTE over-samples the minority class by generating new synthetic data. A number of methods have been developed based on the principle of SMOTE, such as, Borderline-SMOTE [20], ADASYN [21], RAMO [22] and Random balance [23]. Furthermore, Garcia et al. [24] observed that over-sampling consistently outperforms under-sampling for strongly imbalanced datasets.

Hence, in this work we considered three over-sampling techniques. Similar to [3], the class with the highest number of examples is considered the majority class, while all others are considered minority classes. Then, the over-sampling techniques are applied to generate synthetic samples for each minority class.

- **Synthetic Minority Over-sampling Technique (SMOTE)** [19], which creates artificial instances for the minority class. The process works as follows: Let $\boldsymbol{x}_i$ be an instance from the minority class. To create an artificial instance from $\boldsymbol{x}_i$, SMOTE first isolates the k-nearest neighbors of $\boldsymbol{x}_i$, from the minority class. Afterward, it randomly selects one neighbor and randomly generates a synthetic example along the imaginary line connecting $\boldsymbol{x}_i$ and the selected neighbor.
- **Ranked Minority Over-sampling (RAMO)** [22], which performs a sampling of the minority class according to a probability distribution, followed by the creation of synthetic instances. The RAMO process works as follows: For each instance $\boldsymbol{x}_i$ in the minority class, its k_1 nearest neighbors (k_1 is a user defined neighborhood size) from the whole dataset are isolated. The weight r_i of $\boldsymbol{x}_i$ is defined as:

$$r_i = \frac{1}{1 + exp(-\alpha.\delta_i)}, \tag{1}$$

  where δ_i is the number of majority cases in the k-nearest neighborhood. Evidently, an instance with a large weight indicates that it is surrounded by majority class samples, and thus difficult to classify.

  After determining all weights, the minority class is sampled using these weights to get a sampling minority dataset G. The synthetic samples are generated for each instance in G by using SMOTE on k_2 nearest neighbors where k_2 is a user-defined neighborhood size.
- **Random Balance (RB)** [23], which relies on the amount of under-sampling and over-sampling that is problem specific and that has a significant influence on the performance of the classifier concerned. RB maintains the size of the dataset, but varies the proportion of the majority and minority classes, using a random ratio. This includes the case where the minority class is over represented and the imbalance ratio is inverted. Thus, repeated applications of RB produce datasets having a large imbalance ratio variability, which promotes diversity [23]. SMOTE and random under-sampling are used to respectively increase or reduce the size of the classes to achieve the desired ratios.

  Given a dataset S, with minority class S_P and majority class S_N, the RB procedure can be described as follows:
  1) The modified size, $newMajSize$, of the majority class, is defined by a random number generated between 2 and $|S|-2$ (both inclusive). Accordingly, the modified size, $newMinSize$, of the minority class becomes $|S| - newMajSize$.
  2) If $newMajSize < |S_N|$, the majority class S'_N is created by RUS the original S_N so that the final size $|S'_N| = newMajSize$. Consequently, the new minority class S'_P is obtained from S_P using SMOTE to create $newMinSize - |S_P|$ artificial instances.
  3) Otherwise, S'_P is the class created by RUS S_P. On the other hand, S'_N is the class that includes artificial samples generated using SMOTE on S_N. Thus, finally, $|S'_P| = newMinSize$ and $|S'_N| = newMajSize$.

IV. Experiments

A. Datasets

A total of 26 multi-class imbalanced datasets taken from the Keel repository [25] was used in this analysis. The key features of the datasets are presented in Table I. The IR is computed as the proportion of the number of the majority class examples to the number of minority class examples. In this case, the class with maximum number of examples is the majority class, and the class with the minimum number of examples is the minority one. We grouped the datasets according to their IRs using the group definitions suggested by [26]. Datasets with low IR ($IR < 3$) are highlighted with dark gray, whereas datasets with medium IR ($3 < IR < 9$) are in light gray.

B. Experimental setup

The Weka 3.8 along with Matlab 8.4.0 was used in the experiments. Results were obtained with a 5×2 stratified cross-validation. Performance evaluation is conducted using the multi-class generalization of the AUC, F-measure and G-mean, as the standard classification accuracy is not suitable for imbalanced learning [6].

The pool size for all ensemble techniques was set to 100. The classifier used as a base classifier in all ensembles was J48, which is the Java implementation of Quinlan's C4.5, available in Weka 3.8. Here, C4.5 was used with Laplace smoothing at the leaves, but without pruning and collapsing as recommended in [6].

All preprocessing techniques were combined with Bagging during the pool generation phase. Table II lists such combinations. The preprocessing techniques, RAMO and SMOTE, have user-specified parameters. In the case of RAMO, we used $k_1 = 10$, $k_2 = 5$ and $\alpha = 0.3$. For SMOTE and RB, the number of nearest neighbors was 5. These parameter settings

TABLE I

CHARACTERISTICS OF THE 26 MULTI-CLASS IMBALANCED DATASETS TAKEN FROM THE KEEL REPOSITORY. COLUMN #E SHOWS THE NUMBER OF INSTANCES IN THE DATASET, COLUMN #A THE NUMBER OF ATTRIBUTES (NUMERIC/NOMINAL), #C SHOWS THE NUMBER OF CLASSES IN THE DATASET, AND COLUMN IR THE IMBALANCE RATIO.

Dataset	#E	#A	#C	IR	Dataset	#E	#A	#C	IR
Vehicle	846	(18/0)	4	1.09	CTG	2126	(21/0)	3	9.40
Wine	178	(13/0)	3	1.48	Zoo	101	(16/0)	7	10.25
Led7digit	500	(7/0)	10	1.54	Cleveland	467	(13/0)	5	12.62
Contraceptive	1473	(9/0)	3	1.89	Faults	1941	(27/0)	7	14.05
Hayes-Roth	160	(4/0)	3	2.10	Autos	159	(16/10)	6	16.00
Column3C	310	(6/0)	3	2.50	Thyroid	7200	(21/0)	3	40.16
Satimage	6435	(36/0)	7	2.45	Lymphography	148	(3/15)	4	40.50
Laryngeal3	353	(16/0)	3	4.19	Post-Operative	87	(1/7)	3	62.00
New-thyroid	215	(5/0)	3	5.00	Wine-quality red	1599	(11/0)	11	68.10
Dermatology	358	(33/0)	6	5.55	Ecoli	336	(7/0)	8	71.50
Balance	625	(4/0)	3	5.88	Page-blocks	5472	(10/0)	5	175.46
Flare	1066	(0/11)	6	7.70	Abalone	4139	(7/1)	18	45.93
Glass	214	(9/0)	6	8.44	Nursery	12690	(0/8)	5	2160.00

were adopted from [6]. Finally, for all the dynamic selection methods, we used 7 nearest neighbors to define the region of competence as in [15, 7].

TABLE II

PREPROCESSING METHODS USED FOR CLASSIFIER POOL GENERATION.

Bagging based methods		
Abbr.	Name	Description
Ba	Bagging	Bagging without preprocessing
Ba-RM100	Bagging+RAMO 100%	RAMO to double the minority class
Ba-RM	Bagging+RAMO	RAMO to make equal size for both classes
Ba-SM100	Bagging+SMOTE 100%	SMOTE to double the minority class
Ba-SM	Bagging+SMOTE	SMOTE to make equal size for both classes
Ba-RB	Bagging+RB	RB to randomly balance the two classes

The complete framework for a single replication is presented in Figure 1. The original dataset was divided into two equal halves. One of them was set aside for testing, while the other half was used to train the base classifiers and to derive the dynamic selection set. Let us now highlight the process of setting up the DSEL. Here, instead of dividing the training set, we augment it using the data preprocessing, to create DSEL. Moreover, the Bagging method is applied to the training set, generating a bootstrap with 50% of the data. Then, the preprocessing method is applied to each bootstrap, and the resulting dataset is used to generate the pool of classifiers. Since we considered a single training dataset, the DSEL dataset has an overlap with the datasets used during Bagging iterations. However, the randomized nature of the preprocessing methods allows the DSEL not to be exactly the same as the training datasets. Thus, avoiding overfitting issues.

C. Results according to data preprocessing method

In this section, we compare the performance of each preprocessing method with respect to each ensemble technique. Tables III, IV and V show the average rank for the AUC, F-measure and G-mean, respectively. The best average rank is in bold. We can see that the SM and SM100 obtained the best results. Furthermore, the configuration using only Bagging always presented the highest average rank.

The Finner's [27] step-down procedure was conducted at a 95% significance level to identify all methods that were equivalent to the best ranked one. The analysis demonstrates that considering the F-measure and G-mean, the result obtained using preprocessing techniques is always statistically better when compared to using only Bagging.

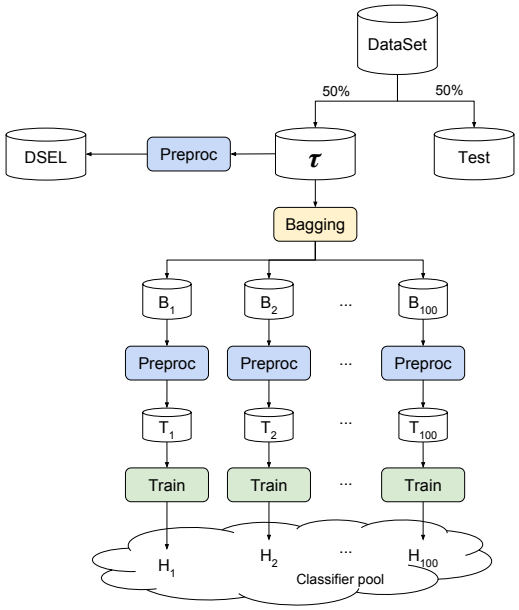


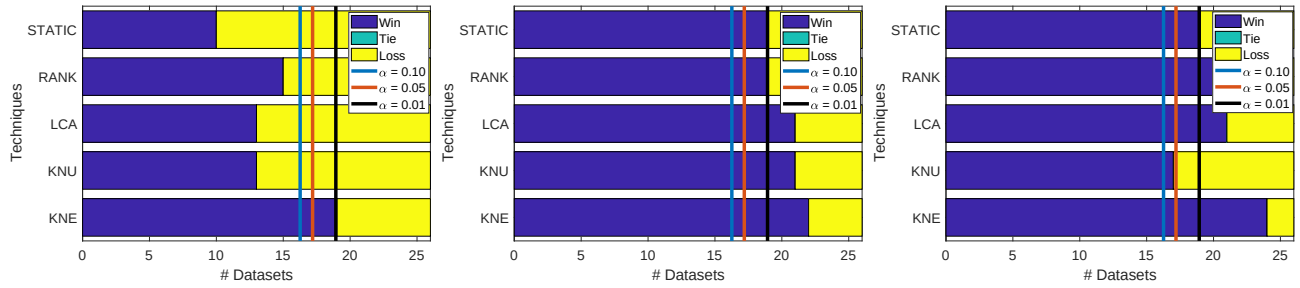
Fig. 1. The framework for training base classifiers and to prepare a DSEL for testing. Here, $\boldsymbol{\tau}$ is the training data derived from the original dataset, B_i is the dataset generated from the ith Bagging iteration, T_i is the dataset produced by preprocessing (*Preproc*) B_i and H_i is the ith base classifier.

TABLE III

AVERAGE RANKINGS ACCORDING TO AUC. METHODS IN BRACKETS ARE STATISTICALLY EQUIVALENT TO THE BEST ONE.

Algorithm	Bagging	RM	RM100	SM	SM100	RB
KNE	3.85	3.81	[2.88]	[3.46]	**2.62**	4.38
KNU	[3.35]	3.50	**2.88**	3.73	[3.27]	4.27
LCA	[3.23]	[3.19]	[3.42]	3.77	**2.77**	4.62
RANK	3.73	[3.38]	3.92	[3.42]	**2.88**	[3.65]
STATIC	[3.42]	3.50	**2.58**	3.81	[3.31]	4.38

Moreover, we conducted a pairwise comparison between each ensemble methods using data preprocessing with the same methods using only Bagging (baseline). For the sake of simplicity, only the best data preprocessing for each technique was considered (i.e., the best result of each row of Tables III,



(a) AUC (b) F-measure (c) G-mean

Fig. 2. Sign test computed over the wins, ties and losses. The vertical lines represents the critical value for at a significance level $\alpha = \{0.1, 0.05, 0.01\}$.

TABLE IV
AVERAGE RANKINGS ACCORDING TO F-MEASURE. METHODS IN BRACKETS ARE STATISTICALLY EQUIVALENT TO THE BEST ONE.

Algorithm	Bagging	RM	RM100	SM	SM100	RB
KNE	5.00	[3.35]	[3.15]	[3.23]	**2.65**	3.62
KNU	4.42	**3.08**	[3.42]	[3.35]	[3.19]	[3.54]
LCA	3.92	3.42	3.62	[3.00]	**2.38**	4.65
RANK	4.19	[3.46]	3.69	[3.38]	**2.54**	3.73
STATIC	4.00	[3.54]	[3.15]	[3.46]	**2.81**	4.04

TABLE V
AVERAGE RANKINGS ACCORDING TO G-MEAN. METHODS IN BRACKETS ARE STATISTICALLY EQUIVALENT TO THE BEST ONE.

Algorithm	Bagging	RM	RM100	SM	SM100	RB
KNE	5.42	[3.46]	[3.50]	**2.81**	[3.00]	[2.81]
KNU	5.00	[3.00]	4.00	**2.81**	[4.00]	2.19
LCA	4.38	[3.46]	4.08	**2.69**	[3.00]	[3.38]
RANK	4.73	[3.42]	3.92	[3.27]	**2.54**	[3.12]
STATIC	3.92	[3.50]	[3.50]	**3.15**	[3.54]	[3.38]

IV and V). The pairwise analysis is conducted using the Sign test, calculated on the number of wins, ties, and losses obtained by each method using preprocessing techniques, compared to the baseline. The results of the Sign test is presented in Figure 2.

The Sign test demonstrated that the data preprocessing significantly improved the results of these techniques according to the F-measure and G-mean. Considering these two metrics, all techniques obtained a significant number of wins for a significance level $\alpha = 0.05$. Moreover, three out of five techniques presented a significant number of wins for $\alpha = 0.01$. Hence, the results obtained demonstrate that data preprocessing techniques indeed play an important role when dealing with multi-class imbalanced problems.

Furthermore, DS techniques are more benefited from the application of data preprocessing (i.e., presented a higher number of wins). This results can be explained by the fact the data preprocessing techniques are applied in two stages: First, it is used in the ensemble generation stage in order to generate a diverse pool of classifiers. Then, they are also used in order to balance the distribution of the dynamic selection dataset for the estimation of the classifiers' competences.

D. Dynamic selection vs static combination

In this experiment we compare the performance of the dynamic selection approaches versus static ones. For each technique, the best performing data preprocessing technique is selected (i.e., best result from each row of Tables III, IV and V). Then, new average ranks are calculated for these methods. Table IV-D presents the average rank of the top techniques according to each metric.

TABLE VI
AVERAGE RANKS FOR THE BEST ENSEMBLE METHODS. (A) ACCORDING TO AUC, (B) ACCORDING TO F-MEASURE AND (C) ACCORDING TO G-MEAN. RESULTS THAT ARE STATISTICALLY EQUIVALENT TO THE BEST ONE ARE IN BRACKETS.

(a) AUC		(b) F-measure		(c) G-mean	
Methods	Rank	Method	Rank	Method	Rank
Ba-SM+KNU	2.04	Ba-RM+KNU	2.15	Ba-SM+KNE	2.23
Ba-SM100+KNE	[2.42]	Ba-SM100+KNE	[2.31]	Ba-SM+KNU	[2.42]
Ba-SM	2.50	Ba-SM100	[2.46]	Ba-SM	[2.62]
Ba-SM100+RANK	3.77	Ba-SM100+RANK	3.58	Ba-SM100+RANK	3.38
Ba-SM100+LCA	4.27	Ba-SM100+LCA	4.50	Ba-SM+LCA	4.35

Based on the average ranks, we can see that the DES techniques present a lower average rank when compared to that of the static combination for the three performance measures. Hence, DES techniques are suitable for dealing with multi-class imbalance. The performance of DES techniques (KNE and KNU) and the static combination were statistically equivalent considering the F-measure and G-mean, while the performance of the KNU was significantly better considering the AUC. On the other hand, the DCS techniques (LCA and RANK) presented a higher average rank when compared to the static ensemble, and may not be suitable to handle multi-class imbalanced problems.

V. CONCLUSION

In this work, we conducted a study on dynamic ensemble selection and data preprocessing for solving the multi-class imbalanced problems. A total of four dynamic selection techniques and five preprocessing techniques were evaluated in this experimental study.

Results obtained over 26 multi-class imbalanced problems demonstrate that the dynamic ensemble selection techniques studied (KNE and KNU) obtained a better result than static ensembles based on AUC, F-measure and G-mean. Moreover, the use of data preprocessing significantly improves the performance of DS and static ensembles. In particular, the SMOTE technique presented the best results. Furthermore,

DS techniques seems to benefit more of data preprocessing methods since they are applied not only to generate the pool of classifiers but also to edit the distribution of the dynamic selection dataset.

Future works would involve the definition of new preprocessing techniques specific to deal with multi-class imbalance as well as the definition of cost-sensitive dynamic selection techniques to handle multi-class imbalanced problems.

REFERENCES

[1] H. He and E. Garcia, "Learning from imbalanced data," *Trans. on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[2] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.

[3] L. Abdi and S. Hashemi, "To combat multi-class imbalanced problems by means of over-sampling techniques," *Trans. on Knowledge and Data Engineering*, vol. 28, no. 1, pp. 238–251, 2016.

[4] A. Fernández, V. López, M. Galar, M. J. Del Jesus, and F. Herrera, "Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches," *Knowledge-based systems*, vol. 42, pp. 97–110, 2013.

[5] F. Fernández-Navarro, C. Hervás-Martínez, and P. A. Gutiérrez, "A dynamic over-sampling procedure based on sensitivity for multi-class problems," *Pattern Recognition*, vol. 44, no. 8, pp. 1821–1833, 2011.

[6] J. F. Díez-Pastor, J. J. Rodríguez, C. I. García-Osorio, and L. I. Kuncheva, "Diversity techniques improve the performance of the best imbalance learning ensembles," *Information Sciences*, vol. 325, pp. 98 – 117, 2015.

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

[8] A. Roy, R. M. Cruz, R. Sabourin, and G. D. Cavalcanti, "A study on combining dynamic selection and data preprocessing for imbalance learning," *Neurocomputing*, 2018.

[9] R. M. O. Cruz, R. Sabourin, and G. D. C. Cavalcanti, "Prototype selection for dynamic classifier and ensemble selection," *Neural Comput. and Appl.*, vol. 29, no. 2, pp. 447–457, 2018.

[10] R. M. O. Cruz, L. G. Hafemann, R. Sabourin, and G. D. C. Cavalcanti, "DESlib: A dynamic ensemble selection library in python," *arXiv:1802.04967*, 2018.

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

[12] M. Sabourin, A. Mitiche, D. Thomas, and G. Nagy, "Classifier combination for hand-printed digit recognition," in *Int. Conf. on Document Analysis and Recognition*, 1993, pp. 163–166.

[13] A. Ko, R. Sabourin, and J. A. Britto, "From dynamic classifier selection to dynamic ensemble selection," *Pattern Recogn.*, vol. 41, no. 5, pp. 1718–1731, 2008.

[14] A. S. Britto, R. Sabourin, and L. E. S. Oliveira, "Dynamic selection of classifiers - a comprehensive review," *Pattern Recognition*, vol. 47, no. 11, pp. 3665–3680, 2014.

[15] 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.

[16] P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," *ACM Comput. Surv.*, vol. 49, no. 2, pp. 31:1–31:50, 2016.

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

[18] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: a hybrid approach to alleviating class imbalance," *Trans. Sys. Man Cyber. Part A*, vol. 40, no. 1, pp. 185–197, Jan. 2010.

[19] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, no. 1, pp. 321–357, 2002.

[20] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: A new over-sampling method in imbalanced data sets learning," in *Int. Conf. on Advances in Intelligent Computing (ICIC)*, 2005, pp. 878–887.

[21] H. e. a. He, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *Int. Joint Conf. on Neural Networks*, 2008, pp. 1322–1328.

[22] S. Chen, H. He, and E. A. Garcia, "RAMOBoost: ranked minority oversampling in boosting," *Trans. Neur. Netw.*, vol. 21, no. 10, pp. 1624–1642, 2010.

[23] J. F. Díez-Pastor, J. J. Rodríguez, C. I. García-Osorio, and L. I. Kuncheva, "Random balance: Ensembles of variable priors classifiers for imbalanced data," *Know.-Based Syst.*, vol. 85, pp. 96 – 111, 2015.

[24] V. García, J. S. Sánchez, and R. A. Mollineda, "On the effectiveness of preprocessing methods when dealing with different levels of class imbalance," *Know.-Based Syst.*, vol. 25, no. 1, pp. 13–21, Feb. 2012.

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

[26] A. Fernández, S. García, M. J. del Jesus, and F. Herrera, "A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets," *Fuzzy Sets and Systems*, vol. 159, no. 18, pp. 2378 – 2398, 2008.

[27] H. Finner, "On a monotonicity problem in step-down multiple test procedures," *Journal of the American Statistical Association*, vol. 88, no. 423, pp. 920–923, 1993.