Minimum Recall-Based Loss Function for Imbalanced Time Series Classification

Josu Ircio, Aizea Lojo, Usue Mori, Simon Malinowski, and Jose A. Lozano, *Senior Member, IEEE*

***Abstract*—This paper deals with imbalanced time series classification problems. In particular, we propose to learn time series classifiers that maximize the minimum recall of the classes rather than the accuracy. Consequently, we manage to obtain classifiers which tend to give the same importance to all the classes. Unfortunately, for most of the traditional classifiers, learning to maximize the minimum recall of the classes is not trivial (if possible), since it can distort the nature of the classifiers themselves. Neural networks, in contrast, are classifiers that explicitly define a loss function, allowing it to be modified. Given that the minimum recall is not a differentiable function, and therefore does not allow the use of common gradient-based learning methods, we apply and evaluate several smooth approximations of the minimum recall function. A thorough experimental evaluation shows that our approach improves the performance of state-of-the-art methods used in imbalanced time series classification, obtaining higher recall values for the minority classes, incurring only a slight loss in accuracy.**

***Index Terms*—Multivariate time series, imbalanced classification, minimum recall, loss function, neural networks.**

I. Introduction

An IMBALANCED classification problem can be broadly defined as a classification problem in which the distribution of classes is not uniform [1], [2], [3]. That is, among all the classes, there is one, or more than one, that has a higher probability than the rest of the classes. Those imbalanced classification problems in which the differences between the class probabilities are significantly high are considered particularly difficult. In these cases, the class-imbalance may cause traditional classifiers to be biased towards the majority classes, and, often, the real benefit in these problems is obtained by properly classifying the minority classes. Since this highly imbalanced classification problem is very common in real-world scenarios, it has been the subject of many classical studies [4], [5].

Imbalanced time series classification (ITSC) problems also appear frequently in real-world time series scenarios, such as in industrial component monitoring [6], hard drive failure prediction [7], earthquake prediction [8], or in human activity recognition [9]. Nevertheless, few research studies have been conducted for this specific type of problem [10].

The methods proposed to solve ITSC problems can be mainly divided into three groups: resampling methods, cost-sensitive methods and methods that apply specific loss functions to deal with class imbalance. Resampling methods attempt to re-balance the distribution of the classes by adding data samples (oversampling methods [11]) or removing data samples (undersampling methods [12]). Cost-sensitive techniques, on the contrary, consist of assigning higher costs to the miss-classification of the instances of the minority classes, penalizing the bias towards the majority class that many classifiers incur. Finally, the third group of methods focuses on defining new loss functions that adapt the learning process of the classifiers to take into account class imbalance.

Within the resampling methods, there are some approaches in that, despite not being specific for time series, have been frequently applied to solve ITSC problems, for example, SMOTE [13], ADASYN [14] or RUS [12]. However, in ITSC problems, data are time-ordered and this temporal information contained in the series may be essential for the classification. Therefore, there are methods such as SPO [10], INOS [8] and MOGT [15] that try to maintain and transfer this temporal information to the oversampled instances [16]. In general, the main advantages of the resampling methods are that they are easy to understand, and that can be combined with all types of classifiers. Nevertheless, these existing oversampling methods specific for time series make assumptions about the distribution of the observations that often do not hold for real-life data. Moreover, their high computational complexity limits the application of these methods to large time series. In addition, the risk of suffering from overfitting may also increase due to the lack of variety in the generated synthetic samples. With regard to the undersampling methods, the main disadvantage is that useful

Manuscript received 6 April 2022; revised 26 February 2023; accepted 8 April 2023. Date of publication 20 April 2023; date of current version 15 September 2023. The work of Jose A. Lozano was supported in part by the Basque Government under Grants BERC 2022-2025 and IT1504-22, and in part by the Ministry of Science and Innovation: BCAM Severo Ochoa accreditation under Grants CEX2021-001142-S/MICIN/AEI/10.13039/501100011033 and PID2019-104966GB-I00. This work was supported in part by Elkartek program under Grants KK-2022/00048, KK-2022/00049, KK-2022/00106, KK-2021/00095 and KK-2021/00065. Recommended for acceptance by X. Chen. *(Corresponding author: Josu Ircio.)*

Josu Ircio and Aizea Lojo are with the Ikerlan Technology Research Centre, Basque Research and Technology Alliance (BRTA), 20500 Arrasate-Mondragón, Spain (e-mail: jircio@ikerlan.es; alojo@ikerlan.es).

Simon Malinowski is with the IRISA UMR 6074, Univ Rennes, 35700 Rennes, France (e-mail: simon.malinowski@irisa.fr).

Usue Mori is with the Intelligent Systems Group, Department of Computer-Science and Artificial Intelligence, University of the Basque Country UPV/EHU, 20018 San Sebastian, Spain (e-mail: usue.mori@ehu.eus).

Jose A. Lozano is with the Intelligent Systems Group, Department of Computer-Science and Artificial Intelligence, University of the Basque Country UPV/EHU, 20018 San Sebastian, Spain, and also with Basque Center for Applied Mathematics (BCAM), 48009 Bilbao, Spain (e-mail: ja.lozano@ehu.eus).

This article has supplementary downloadable material available at https://doi.org/10.1109/TKDE.2023.3268994, provided by the authors.

Digital Object Identifier 10.1109/TKDE.2023.3268994

information for classification may be lost when eliminating instances [16], [17], [18].

Regarding cost-sensitive techniques, the most basic methods modify the loss function of the classifiers by introducing class-dependent weights [17], [18]. These techniques cannot be combined with all classifiers because not all classifiers define an explicit loss function that can be modified. Furthermore, the results obtained with these methods depend on the weights assigned to each class, and the best choice for these weights can vary across datasets [18]. To address this, some recent methods utilize dynamically changing missclassification costs by updating the weights during the training process as if they were additional parameters to be learned [16], [19], [20]. Nevertheless, adding more parameters can increase the complexity of the training process. Recently, new approaches have been developed that use performance metrics suitable for imbalanced scenarios such as F1 [21], [22] and AUC [23], [24], [25] as targets when training classifiers [21]. To this end, different loss functions based on these metrics are designed with the aim of adapting the learning process of the classifiers to account for class imbalance [26]. However, these loss functions are usually non-differentiable and this is a requirement for applying the traditional gradient-based learning methods. Consequently, surrogates are used to enable the use of common learning methods.

In addition to the particular shortcomings associated with each type of method, most of the existing approaches for dealing with ITSC share the same disadvantage of not being applicable to all the TSC problems without exception. Some methods are restricted to unidimensional time series and cannot be applied to multidimensional time series [8], [27]. Others are restricted to binary classification problems and cannot be applied to multiclass problems [11], [16], [19], [21], [22], [23], [24], [25]. Finally, others have both restrictions at the same time [12], [15], [28].

In this paper, based on the work presented in [7] for predicting hard disk failures and using the approach followed by the third group of methods, we develop a novel method to address ITSC problems. Specifically, we propose to use the minimum recall of the classes as the function to maximize when learning a neural network classifier for time series. Since the minimum recall function is not differentiable, we propose to use different smooth approximations (differentiable) of the minimum recall of the classes as the function to maximize when learning a neural network classifier. Thanks to this, we manage to obtain classifiers that, despite the class-imbalance, tend to maximize at the same time the recall value of all the classes [29]. It should be noted that the proposed method is independent of the characteristics of the ITSC problem, so it can be applied not only to unidimensional or binary problems, but also to multidimensional or multiclass problems. In addition, it does not require modifications in the training data to balance the distribution of the classes such as resampling methods nor learning specific parameters for balancing the cost of miss-classification such as cost-sensitive methods.

The rest of this paper is structured as follows. In Section II, the proposed method for addressing the ITSC problems is presented. In Section III, the experimental framework to evaluate the different variations of our method, and compare it to other state-of-the-art solutions is introduced, followed by an extensive experimental evaluation and discussion in Section IV. Finally, in Section V, the conclusions and future work are presented.

II. Proposed Method

A. Problem Statement

The most common classifiers are designed based on the assumption of an equal class distribution. Specifically, they are trained for maximizing the classification accuracy, or in other words, they try to minimize the classification error. Therefore, in ITSC problems, classifiers are often biased toward the majority classes and tend to completely ignore the minority classes [2]. It follows that accuracy is not a proper measure for learning classifiers in imbalanced scenarios [18].

Thus, the goal in ITSC problems is to design classifiers that, despite the class-imbalance, are focused on properly classifying the minority classes, but without overly affecting the accuracy of the majority classes [1].

B. Minimum Recall to Deal With the Class-Imbalance

The proposed approach is an extension of the idea presented in [7]. Specifically, we propose to maximize the minimum recall of the classes when training a classifier rather than the accuracy which is what most of the traditional classifiers are designed to do. To this end, we choose neural network classifiers because they explicitly define the loss function which allows us to modify their learning process to maximize the minimum recall.

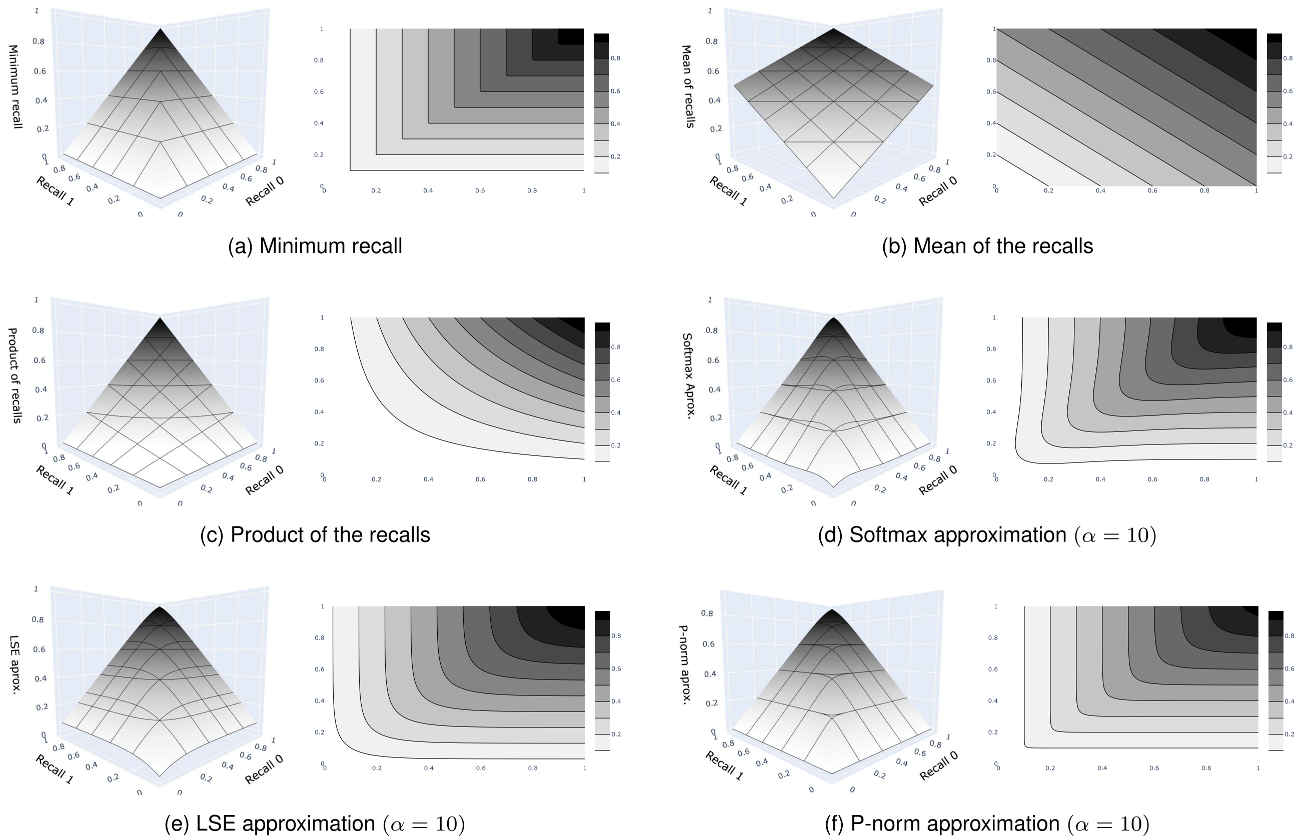
Unfortunately, traditional efficient gradient-based learning methods for neural networks require a differentiable loss function in order to be applied [30]. Since the minimum recall function is not differentiable, we cannot directly learn a classifier to maximize it while using a gradient based method of optimization.

In [7], to address the non-differentiability of the minimum recall, a new framework for learning neural network classifiers was developed. It consists of two genetic algorithms (GAs) that heuristically optimize the network architecture and its weights for maximizing the minimum recall of the classes. Although competitive results were obtained for predicting hard drive failures, applying two GAs for learning the classifier is time-consuming and inefficient.

Consequently, we propose a new approach to learn neural network classifiers that maximizes the minimum recall. Specifically, we propose to explore the use of different smooth (differentiable) approximations of the minimum recall function which have been proven effective in other works [31].

C. Smooth Approximations of the Minimum Recall Function

Let us define a class-imbalanced time series classification problem with m different classes $\{C_1, \ldots, C_m\}$ and with a subset of N time series, $\mathbf{TS} = \{ts_1, ts_2, \ldots, ts_N\}$, and their respective class labels, $\mathbf{C} = \{c_{ts_1}, c_{ts_2}, \ldots, c_{ts_N}\}$. Among these N instances, n_i are the instances of the class C_i, being $N = \sum_{i=1}^{m} n_i$. Without loss of generality, we consider that there is a subset of classes L, with $|L| < m$, for which the class



(a) Minimum recall

(b) Mean of the recalls

(c) Product of the recalls

(d) Softmax approximation ($\alpha = 10$)

(e) LSE approximation ($\alpha = 10$)

(f) P-norm approximation ($\alpha = 10$)

Fig. 1. Approximations of the minimum recall function for a binary classification problem. For each approximation, a surface plot and a contour plot are presented with the values taken by each approximation in terms of the recalls of both classes.

probability is significantly lower than for the rest of the classes, that is, $n_i \ll n_j$ for all $C_i \in L$ and $C_j \notin L$.

The objective is to build a classifier f that is able to predict the class label c_{ts} of any time series ts, $f(ts) = c_{ts}$, as accurately as possible.

For each class, we denote with TS_j the subset of time series from TS that belong to class C_j ($TS_j = \{ts \,|\, c_{ts} = C_j\}$). Given this, the recall value obtained by the classifier for class C_i, $recall_{C_i}$, is defined as the proportion of instances of class C_i that it correctly classifies

$$recall_{C_i} = \frac{\sum_{ts \in TS_i} 1_{C_i}(f(ts))}{n_i},$$

being 1_{C_i} the Indicator function of the class C_i. Therefore, the minimum recall is calculated as

$$\text{Minimum recall} = \min(recall_{C_1}, recall_{C_2}, \ldots, recall_{C_m}).$$

With this in mind, we first explore the most obvious approximations of the minimum recall function, such as the mean and the product of the recalls. Then, as more complex and precise approximations of the minimum recall, we also explore a different parametric family of functions F_α. A parametric family of functions F_α is defined as a set of functions such that for every $\alpha \in \{1, \ldots, \infty\}$, the function f_α contained in F_α is smooth, and when α tends to ∞, f_α converges to a function f, in our case the minimum recall (for $f_\alpha \in F_\alpha,\ f_\alpha \to f,$ when $\alpha \to \infty$) [32]. It should be noted that when α is high, the parametric approximations f_α of the minimum recall could be almost non-differentiable. This can cause numerical problems with the gradient-descent learning methods, such as the exploding gradient problems [33].

The approximations of the minimum recall analyzed are the following:

- *Mean of the recalls*

$$\text{Mean recall} = \frac{\sum_{i=1}^{m} recall_{C_i}}{m}.$$

The mean of the recalls gives the same importance to all the class recalls. However, the mean of the recalls is not as sensitive as the minimum recall when there are only some classes with low recalls. As can be observed in Fig. 1(b), for a binary problem, the approximation obtained with the mean of the recalls does not resemble the minimum recall function when the recall values of the classes are far apart.

- *Product of the recalls*

$$\text{Product recall} = \prod_{i=1}^{m} recall_{C_i}.$$

Since the recall values are in $[0, 1]$, the product of them will always be significantly lower than the minimum of the recalls (see Fig. 1(c)). Furthermore, the greater the number of classes, the greater the difficulty of obtaining high values for the product of the recalls. This can make the learning process with the resulting loss function slow and can fail to find suitable solutions.

- *Softmax approximation [32], [34]*

$$\text{Softmax}(\alpha) = \frac{\sum_{i=1}^{m} recall_{C_i} \cdot \exp(-\alpha \cdot recall_{C_i})}{\sum_{i=1}^{m} \exp(-\alpha \cdot recall_{C_i})}.$$

As can be seen in Fig. 1(d), the softmax function resembles the minimum recall, except when the recalls of all classes are values close to each other. In these cases, the softmax approximation obtains higher values than the minimum recall. Setting α to a high value increases the weight assigned to the class with the lowest recall and decreases the weight assigned to the other classes. Therefore, as mentioned before, by adjusting the value of α, we can control the level of similarity between the softmax approximation and the minimum recall.

- *LogSumExp (LSE) approximation [32], [34]*

$$\text{LSE}(\alpha) = \frac{1}{-\alpha} \log\left(\sum_{i=1}^{m} \exp(-\alpha \cdot recall_{C_i})\right).$$

When the recalls of some classes are low, the LSE approximation correctly fits the minimum recall function. However, as the recalls of all classes increase, the slope of the LSE tends to decrease and differs from the minimum recall function (see Fig. 1(e) for a binary problem). This situation can prevent the learning process from obtaining high values for all recalls at once.

- *P-norm approximation:* the L^{∞}-norm or maximum norm is the limit of the L^{p}-norms for $p \to \infty$, and in the case that $p \to -\infty$, it is equivalent to the minimum function [32]. Therefore, the minimum recall can be approximated with the following function:

$$\text{P-norm}(\alpha) = \left(\sum_{i=1}^{m} |recall_{C_i}|^{(-\alpha)}\right)^{\frac{1}{(-\alpha)}}.$$

This function resembles the minimum recall function, nevertheless, when the recall values are low, it could be almost non-differentiable (see Fig. 1(f) for a binary problem). As previously mentioned, this can cause problems with the gradient-descent learning methods.

The defined smooth approximations are used to replace the minimum recall when plugging it as the function to optimize when learning the classifier. Therefore, each of these smooth approximations will generate a differentiable loss function. From here on, for simplicity, we will talk about the generated loss functions instead of the approximations.

III. Experimental Framework

In this section, the objectives of the experiments, the benchmark datasets and the neural network classifiers used in all the experiments are presented.

A. *Objectives of the Experiments*

The first objective of our experimentation is to find the best smooth approximation of the minimum recall among those presented in Section II-C. Then, the second objective is to compare the performance of this loss function with other state-of-the-art methods for imbalanced problems. Finally, the third objective is to study the sensitivity of the α parameter of the proposed parametric smooth loss functions in classification results.

B. *Benchmark Datasets*

To carry out the experimentation, the datasets provided by the UEA & UCR time series classification repository [35] are used. Since, among these datasets, there are some that are multiclass, we use the imbalance-degree (ID) [4] as a measure of imbalance, instead of the typically used imbalance ratio (IR), which is only useful for binary classification problems. The ID is a number whose whole part represents the number of minority classes minus one, and the decimal part represents the estimated level of imbalance of the class distribution. Given a classification problem with m classes, let $\boldsymbol{\zeta} = (\zeta_1, \zeta_2, \ldots, \zeta_m)$ be defined as the empirical distribution of the classes. Each class probability ζ_i is estimated by just using the relative frequency of the class C_i in the dataset. Moreover, let Z^m be defined as the set containing all the possible empirical distributions $\boldsymbol{\zeta}$ and let $Z_k^m \subset Z^m$, $k \in \{0, 1, \ldots, m-1\}$ be a subset containing all the empirical class distributions containing exactly k minority classes. Formally, $Z_k^m \triangleq \{\boldsymbol{\zeta} \in Z^m \ : \ k = \sum_{i=1}^{m} \mathbb{1}(\zeta_i < 1/m)\}$. In addition, the equiprobability case is denoted as $\mathbf{e} = (e_1, e_2, \ldots, e_m)$ where $e_i = 1/m$. In this context, the ID of the problem is defined as

$$ID(\boldsymbol{\zeta}) = \frac{d_{\triangle}(\boldsymbol{\zeta}, \mathbf{e})}{d_{\triangle}(\boldsymbol{\iota}_k, \mathbf{e})} + (k-1),$$

where k is the number of minority classes, $d_{\triangle}$ is the chosen distance/similarity function to instantiate ID and $\boldsymbol{\iota}_k$ the distribution showing exactly k minority classes with the highest distance to the equiprobability ($\arg\max_{\boldsymbol{\zeta} \in \mathbf{Z}_k^m} d_{\triangle}(\boldsymbol{\zeta}, \mathbf{e})$). In our case, to compute the ID, the Hellinger distance is used as recommended by the authors. Subsequently, the selected datasets are those with a decimal part of the ID higher or equal to 0.2 because it indicates a considerable estimated level of imbalance between the distribution of the classes.

Additionally, the three datasets created in [7] from the open data provided by the Backblaze[1] company are also used (HD-1%, HD-3% and HD-5%). The properties of all the selected datasets are summarized in Table I.

C. *Neural Networks Classifiers Used for the Experiments*

The proposed approach is independent of the neural network classifier used. The unique conditions that the classifiers to be used must fulfill are that they must explicitly define a loss function which can be modified and that they must be able to

[1] [Online]. Available: https://www.backblaze.com/b2/hard-drive-test-data.html

deal with univariate and multivariate time series classification problems. Therefore, we select, without loss of generality, two of the best time series neural network classifiers [36], [37]: the Multivariate Long Short Term Memory Fully Convolutional Network (MLSTM-FCN) [9] and the Residual network (Resnet) [38]. However, other relevant classifiers such as InceptionTime [39] and ROCKET [40] could also be considered. All the selected classifiers are trained using the Adam stochastic optimizer [41] and a maximum of 60 epochs, which may be interrupted if during 10 epochs the classifier does not improve. When the network is trained to maximize the accuracy, the binary cross entropy (BCE) loss and the cross entropy (CE) loss are used depending on whether the classification problem is binary or multiclass. Based on preliminary experiments, the batch size for the MLSTM-FCN classifier is set to the minimum between 128 and the number of instances of the training set. For the Resnet classifier, we use the same batch size as in the original paper [38], which is the minimum between 64 and (the number of training set instances) /10. Moreover, the learning rate is initialized to 0.001, decreased to 0.0001 during the epochs from 35 to 42 and decreased again to 0.00001 during the last epochs up to 60 for the MLSTM-FCN classifier. For the ResNet classifier, we used a total of 1,500 epochs.

IV. Results and Discussion

Considering the stochastic nature of the MLSTM-FCN and Resnet classifiers, as well as the related work used for comparison purposes, each method is run 10 times for each dataset. The results of the experiments are measured using the average and the standard deviation of both the accuracy (*acc*) and the minimum of the recalls (*minRec*), of all runs. On the one hand, the *acc* allows us to control the overall performance of the classifier and whether the classification of the majority classes is being affected when trying to predict the minority classes. On the other hand, the *minRec* indicates whether the minority classes are being properly classified. To compare the performance of the different methods over multiple datasets, we analyze if there are significant differences in terms of the minimum recall and the accuracy. In order to do that, as recommended in [42], [43], the Wilcoxon signed-rank statistical test with the Shaffer correction (or Shaffer dynamic correction when there are less than 10 methods for comparison) is applied with a significance level of 0.05. The results obtained with these tests are represented in diagrams in which the classifiers are grouped by a thick bold line if there are no significant pairwise differences between them.

A. Comparison of the Performance of the Different Minimum Recall-Based Loss Functions

The goals to be achieved with this experimentation are 1) to verify that, for ITSC problems, training a classifier with a minimum recall-based loss function obtains higher minimum recall values than when training it in order to maximize the accuracy, but without significantly decreasing the obtained overall accuracy and 2) to analyze the performance of the different smooth loss functions presented in Section II-C and determine which is the best.

TABLE I
Properties of the Benchmark Datasets

Dataset	Train size	Test size	Ndim	Length	Nclass	ID train
Phal.Out.Co.	1800	858	1	80	2	0.20
OSULeaf	200	242	1	427	6	2.21
Haptics	155	308	1	1092	5	0.24
Prox.Phal.Out.Co.	600	291	1	80	2	0.24
ECG200	100	100	1	96	2	0.25
ChlorineConc.	467	3840	1	166	3	1.25
Worms	181	77	1	900	5	3.25
EigenWorms	128	131	6	17984	5	3.26
Heartbeat	204	205	61	405	2	0.30
Prox.Phal.Out.AGr.	400	205	1	80	3	0.30
ElectricDevices	8926	7711	1	96	7	3.31
Mid.Phal.Out.AGr.	400	154	1	80	3	1.31
Mid.Phal.TW	399	154	1	80	6	3.34
FiftyWords	450	455	1	270	50	35.38
Dist.Phal.Out.AGr.	400	139	1	80	3	1.41
Earthquakes	322	139	1	512	2	0.45
Dist.Phal.TW	400	139	1	80	6	3.47
Prox.Phal.TW	400	205	1	80	6	3.47
LSST	2459	2466	6	36	14	9.49
MedicalImages	381	760	1	99	10	7.50
Wafer	1000	6164	1	152	2	0.61
HD-5%	3891	1217	12	200	2	0.71
ECG5000	500	4500	1	140	5	2.73
HD-3%	3792	1185	12	200	2	0.79
HD-1%	3715	1162	12	200	2	0.88

For each dataset, the number of instances of the training and test sets, the number of dimensions (Ndim), the lengths of the series, the number of classes (Nclass) and the ID of the training set are presented.

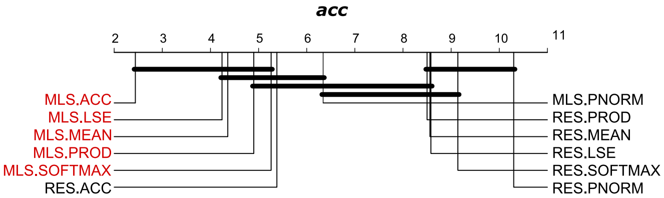


Fig. 2. Diagram of the pairwise differences in terms of the accuracy with a significance level of 0.05 for both classifiers (MLSTM-FCN (MLS) and Resnet (RES)) learned with the proposed loss functions and *acc* based loss function.

To do this, we train the MLSTM-FCN and Resnet classifiers with the proposed loss functions (see Section II) and with the traditional accuracy-based loss functions for all benchmark datasets. To try to avoid possible learning problems resulting from obtaining functions that are almost non-differentiable, we establish the α parameter of all the parametric approximations to 10.

In Table II, the average and the standard deviation of the *acc* and *minRec* for all the runs and all the datasets are presented (because the results for the Resnet classifier are worse, they are only shown in the Appendix) (available online). For each dataset, the loss functions with the best results in terms of *acc* and *minRec* are highlighted. Additionally, in Figs. 2 and 3, diagrams of the pairwise statistical differences obtained for the *acc* and *minRec* measures are displayed respectively.

On the one hand, Table II shows that, in terms of *acc*, the MLS learned with the accuracy-based loss function obtains the best results for the majority of the datasets. However, in this case, the differences with the MLS learned with the LSE, the mean, the product and the softmax loss functions are not statistically significant (See Fig. 2).

TABLE II
TABLE WITH THE OBTAINED RESULTS FOR THE MLSTM-FCN CLASSIFIER LEARNED WITH THE PROPOSED LOSS FUNCTIONS AND THE ACCURACY-BASED LOSS FUNCTION

	MLSTM-FCN											
	ACC		MEAN		PROD		SOFTMAX		LSE		PNORM	
Dataset	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec
Phal.Out.Co.	**0.79±0.01**	0.64±0.05	0.75±0.02	**0.72±0.01**	0.76±0.02	**0.72±0.02**	0.75±0.02	**0.72±0.01**	0.74±0.01	**0.72±0.01**	0.75±0.02	0.62±0.04
OSULeaf	0.73±0.02	0.00±0.00	**0.79±0.03**	0.56±0.05	0.76±0.02	0.61±0.01	0.74±0.01	0.59±0.01	0.76±0.02	**0.63±0.02**	0.72±0.02	0.57±0.01
Haptics	0.40±0.02	0.04±0.04	0.38±0.01	0.00±0.00	0.39±0.02	0.23±0.02	0.40±0.03	0.24±0.02	**0.41±0.03**	**0.27±0.01**	0.37±0.02	0.18±0.02
Prox.Phal.Out.Co.	**0.86±0.02**	0.65±0.05	0.66±0.00	0.53±0.00	**0.86±0.02**	0.74±0.05	0.83±0.02	**0.75±0.05**	0.84±0.02	**0.75±0.03**	0.82±0.02	0.71±0.07
ECG200	**0.85±0.01**	0.69±0.00	0.82±0.01	**0.80±0.01**	0.81±0.01	**0.80±0.01**	0.81±0.01	0.76±0.01	0.80±0.01	0.77±0.02	0.80±0.01	0.77±0.01
ChlorineConc.	**0.58±0.00**	0.01±0.02	0.43±0.01	0.37±0.01	0.46±0.01	**0.38±0.01**	0.41±0.02	0.37±0.01	0.45±0.01	0.37±0.00	0.42±0.01	0.37±0.01
Worms	0.61±0.01	0.00±0.00	0.54±0.06	0.27±0.08	0.66±0.02	0.36±0.05	0.65±0.02	**0.42±0.06**	0.66±0.02	**0.42±0.07**	**0.67±0.03**	0.39±0.03
EigenWorms	0.73±0.14	0.29±0.12	**0.82±0.03**	0.47±0.12	0.78±0.07	0.49±0.05	0.81±0.03	**0.51±0.07**	0.79±0.03	0.44±0.12	0.60±0.23	0.25±0.15
Heartbeat	**0.74±0.02**	0.43±0.04	0.70±0.03	0.57±0.04	0.71±0.03	0.56±0.04	0.68±0.04	**0.62±0.04**	0.68±0.05	**0.62±0.05**	0.71±0.03	0.60±0.04
Prox.Phal.Out.AGr.	0.75±0.13	0.50±0.03	0.82±0.01	0.70±0.01	0.82±0.00	**0.71±0.00**	0.83±0.02	0.60±0.11	0.82±0.03	0.62±0.07	**0.84±0.01**	0.63±0.05
ElectricDevices	0.63±0.17	0.19±0.08	**0.74±0.01**	0.32±0.01	0.72±0.01	0.34±0.02	0.62±0.01	0.33±0.03	0.66±0.06	0.32±0.03	0.66±0.04	**0.35±0.01**
Mid.Phal.Out.AGr.	**0.58±0.10**	0.18±0.07	0.49±0.08	0.31±0.07	0.52±0.08	0.33±0.08	0.51±0.03	0.36±0.03	0.51±0.02	**0.39±0.04**	0.56±0.05	0.28±0.03
Mid.Phal.TW	**0.55±0.01**	0.00±0.00	0.54±0.01	0.00±0.00	0.40±0.02	**0.22±0.00**	0.27±0.02	0.00±0.00	0.26±0.04	0.09±0.08	0.24±0.02	0.00±0.00
FiftyWords	0.28±0.00	0.00±0.00	0.34±0.02	0.00±0.00	0.04±0.03	0.00±0.00	0.42±0.01	0.00±0.00	**0.47±0.01**	0.00±0.00	0.05±0.02	0.00±0.00
Dist.Phal.Out.AGr.	**0.73±0.02**	0.58±0.02	0.68±0.04	0.57±0.03	0.71±0.02	0.63±0.04	0.72±0.01	0.61±0.02	**0.73±0.01**	**0.67±0.02**	0.39±0.17	0.05±0.17
Earthquakes	**0.78±0.01**	0.15±0.03	0.66±0.01	0.57±0.03	0.66±0.03	0.51±0.03	0.67±0.01	**0.64±0.01**	0.66±0.02	0.61±0.02	0.66±0.02	0.62±0.01
Dist.Phal.TW	**0.69±0.01**	0.00±0.00	0.59±0.02	0.01±0.03	0.58±0.01	0.14±0.03	0.58±0.04	0.20±0.06	0.59±0.02	**0.22±0.04**	0.56±0.01	0.17±0.04
Prox.Phal.TW	**0.82±0.00**	0.00±0.00	0.72±0.01	0.00±0.00	0.69±0.03	0.16±0.03	0.58±0.01	**0.49±0.01**	0.63±0.02	0.40±0.05	0.77±0.01	0.02±0.03
LSST	**0.60±0.01**	0.00±0.00	0.54±0.04	0.01±0.02	0.11±0.04	0.00±0.00	0.28±0.07	0.04±0.02	0.36±0.07	**0.07±0.06**	0.13±0.03	0.00±0.00
MedicalImages	**0.57±0.01**	0.00±0.00	0.41±0.01	0.00±0.00	0.26±0.17	0.00±0.00	0.50±0.01	0.42±0.01	0.56±0.01	**0.48±0.01**	0.40±0.07	0.14±0.12
Wafer	**0.99±0.00**	0.96±0.01	**0.99±0.00**	**0.97±0.00**	**0.99±0.00**	**0.97±0.00**	**0.99±0.00**	0.96±0.01	**0.99±0.01**	**0.97±0.00**	**0.99±0.01**	0.96±0.01
HD-5%	**0.98±0.00**	0.61±0.01	0.97±0.00	**0.63±0.00**	0.97±0.00	**0.63±0.00**	0.96±0.04	**0.63±0.01**	0.97±0.00	**0.63±0.01**	0.97±0.00	**0.63±0.00**
ECG5000	**0.93±0.00**	0.00±0.00	0.91±0.03	0.20±0.04	0.85±0.04	**0.32±0.05**	0.45±0.21	0.11±0.08	0.75±0.11	0.28±0.06	0.48±0.10	0.00±0.00
HD-3%	**0.99±0.00**	0.57±0.01	0.98±0.00	0.61±0.02	0.98±0.00	0.63±0.02	0.98±0.00	**0.66±0.00**	0.98±0.00	0.65±0.01	0.80±0.39	0.24±0.20
HD-1%	**0.99±0.00**	0.48±0.05	**0.99±0.00**	0.66±0.03	**0.99±0.00**	**0.67±0.00**	**0.99±0.00**	**0.67±0.00**	**0.99±0.00**	**0.67±0.00**	0.93±0.13	0.24±0.21
Times the best	**18**	**0**	**5**	**4**	**3**	**9**	**2**	**10**	**5**	**14**	**3**	**2**

For each dataset and each loss function, 10 classifiers are learned. Then, for these 10 classifiers, the average and the standard deviation of the resulting *acc* and *minrec* are presented.

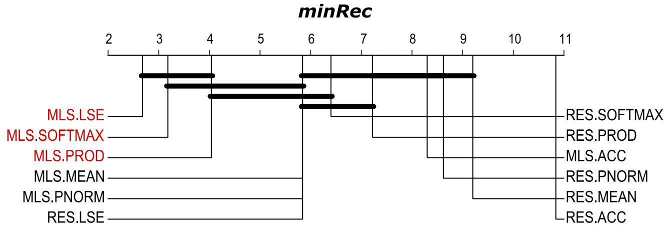


Fig. 3. Diagram of the pairwise differences in terms of the *minRec* with a significance level of 0.05 for both classifiers (MLSTM-FCN (MLS) and Resnet (RES)) learned with the proposed loss functions and accuracy-based loss function.

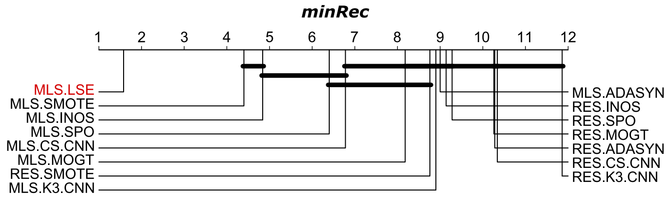


Fig. 4. Diagram of the pairwise differences in terms of the *minRec* with a significance level of 0.05 for the related work with both classifiers (MLSTM-FCN (MLS) and Resnet (RES)) and the best proposed method.

On the other hand, for the *minRec* columns, the LSE loss function is the most highlighted loss function followed by the softmax and the product loss functions. This is confirmed by Fig. 3 in which we can observe that, in terms of the *minRec*, the MLS classifier with the LSE loss function obtains the best results with significant differences with all the rest except for the MLS classifier with the softmax and the product loss functions.

In addition, in both figures, it can be seen that the MLSTM-FCN (MLS) classifier obtains better results than the Resnet (RES) for almost all the loss functions in terms of both *minRec* and *acc*.

These results suggest that the proposed approach meets our expectations. Specifically, the MLS classifier trained with minimum recall-based loss functions is able to obtain higher *minRec* results for the majority of the datasets compared to training it to maximize the accuracy. Moreover, for multiple datasets, the *minRec* value obtained by the MLS classifier trained with the accuracy-based loss function is 0.00±0.00, which means that the classifier has not been able to correctly predict any test instances at least for one class. In contrast, the MLS classifier trained with minimum recall-based loss functions manages to obtain a *minRec* value higher than 0 for considerably more datasets than the accuracy-based loss function, and without a significant decrease in the obtained *acc* values.

It should be noted that, even if the P-norm loss function closely resembles the minimum recall function, it does not obtain the best results. We believe that the reason for this phenomenon is the near non-differentiability of the P-norm approximation when the recalls of the classes are close. As previously explained, this non-differentiability can cause difficulties during training with gradient-based optimization algorithms, resulting in slower convergence and suboptimal solutions.

Apart from that, the LSE, softmax and product loss functions are the most effective loss functions to deal with ITSC problems,

TABLE III
TABLE WITH THE OBTAINED RESULTS FOR THE MLSTM-FCN CLASSIFIER LEARNED WITH THE BEST PROPOSED LOSS FUNCTION AND WITH THE STATE-OF-THE-ART METHODS FOR DEALING WITH THE IMBALANCE

	MLSTM-FCN																	
	LSE		SMOTE		ADASYN		SPO		INOS		MOGT		CS-CNN		K3-CNN	
Dataset	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec	acc	minRec
Phal.Out.Co.	0.74±0.01	**0.72±0.01**	0.72±0.01	0.70±0.01	0.72±0.02	0.43±0.09	0.77±0.03	0.61±0.16	**0.78±0.02**	0.68±0.08	0.77±0.02	0.58±0.14	0.73±0.02	0.65±0.03	**0.78±0.02**	0.68±0.01
OSULeaf	0.76±0.02	**0.63±0.02**	0.78±0.03	0.55±0.04	0.36±0.04	0.00±0.00	**0.79±0.03**	0.52±0.07	0.67±0.02	0.22±0.12	0.68±0.04	0.25±0.18	0.69±0.08	0.19±0.11	0.56±0.02	0.00±0.00
Haptics	0.41±0.03	**0.27±0.01**	**0.42±0.01**	0.11±0.02	0.21±0.00	0.00±0.00	0.38±0.02	0.02±0.03	0.37±0.03	0.01±0.02	0.40±0.04	0.10±0.08	0.36±0.05	0.02±0.04	0.28±0.02	0.00±0.00
Prox.Phal.Out.Co.	0.84±0.02	**0.75±0.03**	0.83±0.05	0.74±0.02	0.84±0.02	0.66±0.12	**0.87±0.03**	0.67±0.12	0.86±0.02	0.73±0.09	0.86±0.02	0.67±0.08	0.77±0.05	0.64±0.09	0.77±0.08	0.61±0.09
ECG200	0.80±0.01	0.77±0.02	0.83±0.03	0.68±0.12	0.80±0.04	0.55±0.17	**0.84±0.02**	0.72±0.08	**0.84±0.02**	0.75±0.07	0.68±0.19	0.40±0.37	0.74±0.07	0.53±0.10	0.83±0.01	**0.81±0.01**
ChlorineConc.	0.45±0.01	0.37±0.00	0.48±0.04	**0.38±0.02**	0.45±0.02	0.31±0.05	0.53±0.04	0.31±0.07	0.52±0.03	0.37±0.06	**0.56±0.02**	0.18±0.09	0.47±0.09	0.14±0.07	0.42±0.04	0.22±0.03
Worms	0.66±0.02	**0.42±0.07**	0.66±0.06	0.36±0.08	0.49±0.02	0.00±0.00	0.71±0.01	0.22±0.03	**0.72±0.02**	0.34±0.09	0.66±0.02	0.05±0.06	0.61±0.10	0.19±0.14	0.47±0.05	0.00±0.00
EigenWorms	0.79±0.03	**0.44±0.12**	**0.81±0.02**	0.38±0.06	0.50±0.06	0.01±0.02	0.68±0.10	0.26±0.08	0.69±0.13	0.32±0.15	–	–	0.71±0.18	0.36±0.16	0.54±0.09	0.05±0.08
Heartbeat	0.68±0.05	**0.62±0.05**	0.62±0.08	0.53±0.04	0.71±0.01	0.02±0.02	**0.72±0.07**	0.38±0.08	0.70±0.03	0.50±0.10	**0.72±0.03**	0.45±0.08	0.60±0.05	0.48±0.03	0.64±0.04	0.56±0.04
Prox.Phal.Out.AGr.	0.82±0.03	0.62±0.07	0.83±0.01	0.58±0.06	0.82±0.02	0.64±0.07	**0.85±0.01**	0.55±0.09	0.82±0.02	**0.67±0.04**	0.84±0.02	0.63±0.08	0.69±0.18	0.27±0.26	0.67±0.18	0.41±0.02
ElectricDevices	0.66±0.06	**0.32±0.03**	0.52±0.21	0.20±0.03	0.64±0.03	0.04±0.02	**0.74±0.02**	0.17±0.04	0.72±0.01	0.16±0.06	0.73±0.01	0.22±0.07	0.59±0.20	0.23±0.07	0.56±0.14	0.25±0.09
Mid.Phal.Out.AGr.	0.51±0.02	**0.39±0.04**	0.55±0.03	0.29±0.02	0.58±0.02	0.24±0.00	**0.61±0.02**	0.23±0.01	0.60±0.02	0.24±0.01	0.59±0.02	0.24±0.01	0.54±0.09	0.25±0.02	0.54±0.07	0.09±0.07
Mid.Phal.TW	0.26±0.04	**0.09±0.08**	0.49±0.01	0.00±0.00	0.51±0.03	0.00±0.00	**0.57±0.02**	0.00±0.00	0.53±0.03	0.03±0.06	0.53±0.02	0.02±0.04	0.46±0.04	0.00±0.00	0.56±0.03	0.00±0.00
FiftyWords	**0.47±0.01**	0.00±0.00	–	–	–	–	–	–	–	–	–	–	0.22±0.01	0.00±0.00	0.16±0.00	0.00±0.00
Dist.Phal.Out.AGr.	**0.73±0.01**	**0.67±0.02**	0.60±0.05	0.48±0.08	0.66±0.08	0.44±0.18	0.64±0.07	0.33±0.21	0.65±0.09	0.44±0.21	0.68±0.07	0.38±0.20	0.69±0.03	0.58±0.02	0.67±0.06	0.47±0.08
Earthquakes	0.66±0.02	**0.61±0.02**	0.57±0.11	0.36±0.04	0.75±0.00	0.00±0.00	**0.76±0.03**	0.23±0.18	0.74±0.06	0.22±0.18	0.65±0.20	0.12±0.18	0.72±0.03	0.46±0.04	0.58±0.08	0.43±0.06
Dist.Phal.TW	0.59±0.02	**0.22±0.04**	0.64±0.02	0.17±0.06	0.60±0.02	0.12±0.03	0.67±0.02	0.19±0.07	0.65±0.02	0.20±0.08	0.64±0.03	0.09±0.07	**0.68±0.01**	0.00±0.00	**0.68±0.00**	0.00±0.00
Prox.Phal.TW	0.63±0.02	**0.40±0.05**	0.73±0.03	0.00±0.00	0.72±0.02	0.00±0.00	0.78±0.02	0.05±0.06	0.76±0.04	0.10±0.06	0.73±0.02	0.00±0.02	**0.80±0.01**	0.00±0.00	0.44±0.02	0.00±0.00
LSST	0.36±0.07	0.07±0.06	0.48±0.08	0.00±0.00	0.35±0.03	0.00±0.01	**0.60±0.02**	0.00±0.00	0.58±0.02	**0.10±0.04**	0.58±0.01	0.00±0.00	0.37±0.03	0.00±0.00	0.35±0.02	0.00±0.00
MedicalImages	0.56±0.01	**0.48±0.01**	0.61±0.01	0.45±0.01	0.66±0.03	0.04±0.03	0.72±0.01	0.28±0.06	**0.73±0.02**	0.39±0.07	0.72±0.01	0.28±0.04	0.32±0.04	0.01±0.03	0.16±0.00	0.00±0.00
Wafer	**0.99±0.01**	**0.97±0.00**	**0.99±0.01**	0.96±0.02	0.97±0.02	0.81±0.15	**0.99±0.02**	0.92±0.11	**0.99±0.01**	0.93±0.07	0.97±0.07	0.90±0.09	0.98±0.01	0.88±0.07	0.69±0.01	0.65±0.01
HD-5%	**0.97±0.00**	**0.63±0.01**	**0.97±0.00**	**0.63±0.01**	0.80±0.36	0.43±0.23	0.55±0.47	0.29±0.28	0.80±0.36	0.39±0.22	–	–	**0.97±0.00**	**0.63±0.00**	0.06±0.00	0.00±0.00
ECG5000	0.75±0.11	**0.28±0.06**	–	–	**0.93±0.00**	0.00±0.00	**0.93±0.01**	0.16±0.03	0.92±0.01	0.16±0.05	**0.93±0.01**	0.13±0.05	0.43±0.12	0.03±0.03	0.63±0.03	0.00±0.00
HD-3%	**0.98±0.00**	**0.65±0.01**	**0.98±0.00**	0.61±0.01	0.68±0.45	0.33±0.25	0.77±0.41	0.38±0.25	0.72±0.43	0.36±0.25	–	–	0.96±0.06	0.60±0.01	0.12±0.29	0.04±0.12
HD-1%	**0.99±0.00**	**0.67±0.00**	0.90±0.14	0.58±0.01	0.55±0.50	0.15±0.18	0.55±0.50	0.14±0.16	0.46±0.50	0.11±0.15	–	–	**0.99±0.00**	0.57±0.05	0.22±0.39	0.09±0.16
Times the best	**6**	**20**	**5**	**2**	**1**	**0**	**12**	**0**	**5**	**2**	**3**	**0**	**4**	**1**	**2**	**1**

Each method is run 10 times for each dataset. Then, for these 10 runs, the average and the standard deviation of the resulting *acc* and *minrec* are presented.

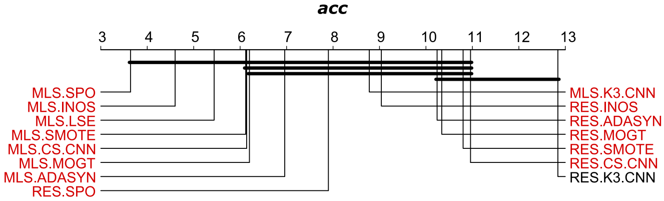


Fig. 5. Diagram of the pairwise differences in terms of the *acc* with a significance level of 0.05 for the related work with both classifiers (MLSTM-FCN (MLS) and Resnet (RES)) and the best proposed method.

TABLE IV
PROPERTIES OF THE GENERATED FACE-DETECTION DATASETS ORDERED BY ID

Dataset	Train size	Train class 0	Train class 1	ID train
FD-Ø	5890	2945	2945	0.0
FD-1	5111	2945	2166	0.1
FD-2	4525	2945	1580	0.2
FD-3	4078	2945	1133	0.3
FD-4	3735	2945	790	0.4
FD-5	3473	2945	528	0.5
FD-6	3273	2945	328	0.6
FD-7	3126	2945	181	0.7
FD-8	3025	2945	80	0.8
FD-9	2965	2945	20	0.9

The ID values vary between 0 and 0.9. For each generated dataset, the number of train instances, the number of train instances of each class, and the ID of the train set are presented.

because, for both performance metrics, *minRec* and *acc*, they do not have significant differences with the best performing loss function.

Furthermore, the MLS classifier with the LSE loss function obtains the highest result for the *minRec* as well as for the *acc* for more datasets than the other two. Therefore, we can consider it as the best proposed method.

B. Comparison With the State-of-the-Art Methods

The objective of this experiment is to analyze whether the best proposed method, the MLS classifier learned with the LSE loss function, is an effective solution to the ITSC problem compared with other state-of-the-art solutions. Specifically, we select relevant methods in the literature that have addressed the ITSC problem, and which are sufficiently detailed to be replicated. They are grouped as follows:

- Oversampling methods not specific for time series: SMOTE [13] and ADASYN [14] (we use the Imbalanced-learn Python package for running both methods [44]). They are not designed to deal with multidimensional time series. Therefore, for multidimensional datasets, all the dimensions are concatenated into a single vector, such that the original multidimensional instances are converted to unidimensional instances.
- Oversampling methods specially designed for time series: SPO [10], INOS [8], MOGT [15]. They are not designed to deal with multidimensional time series nor for multiclass problems. So, first, to be able to apply them to multidimensional datasets, the same process described above for the non-specific oversampling methods is followed. Second, to apply these methods to multiclass ITSC problems, the one-versus-all strategy is used. It consists of creating binary problems by selecting one of these less populated classes as the minority class and creating the majority class by grouping the rest of the classes. In this way, these oversampling methods, exclusively designed for binary problems, can be applied to oversample this minority class until it has the

same number of instances as the most populated class of the original problem. This strategy is applied to all the classes with a lower number of instances than the most populated class (majority class) in order to independently resample them, and thus, obtain an equal number of instances for all the classes. At the end (after applying this strategy to all minority classes), the classifier is learned in this resampled dataset where all the classes will have the same number of instances.

- Cost-sensitive methods: CS-CNN [16] and K3-CNN [19]. They are restricted to binary classification problems. So, to be able to apply them to multiclass classification problems, first of all, the measures used by these methods for computing the missclassification cost, such as accuracy and G-means, must be calculated for multiclass problems. Then, the misclassification cost obtained is applied to all the classes except the most sampled class(es). For the K3-CNN, instead of the parameters specified in Section III-C for learning the classifiers, those that provide the best results in the original work [19] are used. That is, the batch size is set to the minimum between 512 and the number of instances of the training set, the learning rate is fixed to 0.001 and the eps parameter of the Adam optimizer is set to $1e^{-8}$.

As in the previous experiment, we present in a table (Table III) the obtained results disaggregated by dataset (since the results for the Resnet classifier are worse, they are only shown in the Appendix), (available online). In addition, the diagrams of the obtained pairwise differences between all the methods for the *acc* and the *minRec* measures are presented in Figs. 4 and 5.

In Table III, it can be observed that, for the *minRec* column, the proposed approach is highlighted for the majority of the datasets. This is confirmed by Fig. 4, which shows that, in terms of the *minRec*, the proposed approach obtains the best results with significant differences with the rest of the state-of-the-art methods. On the contrary, in terms of *acc*, there are no significant differences among almost any of the methods (see Fig. 5).

Given these results, we can conclude that, in general, the MLS classifier learned with the LSE loss function, despite the imbalance, is able maximize the recall value of all the classes, and at the same time obtain high values for both the overall accuracy and the minimum recall.

C. Sensitivity of the Learned Classifiers to Parameter α

As previously stated, the parametric smooth approximations of the minimum recall function (Softmax, LSE and P-norm functions) depend on a parameter α. This parameter determines the degree of similarity of these functions to the minimum recall function and, hence, the learned classifiers are affected by it as well. The higher the α, the better the approximation. However, when α is high, these functions could be almost non-differentiable, causing numerical problems with the gradient-descent learning methods, such as the exploding gradient problem. With this experiment we want to analyze the sensitivity of this parameter α within our approach. Furthermore, we also want to analyze whether the influence of the α values vary depending on the ID of the datasets.

To accomplish these objectives, we perform experiments with α values of 1,5,10,20,40,60,80,100,120. Additionally, we select the binary and balanced multidimensional time series Face-detection (FD) dataset [35], and we generate a set of datasets with increasing ID. To do this, we randomly undersample instances from one of the two classes (in our case, class 1) from the original dataset. The properties of all the generated datasets are summarized in Table IV.

For each α value and each dataset, we apply our methodology.

The obtained results are presented in Fig. 6. For each dataset, a different line graph is presented. Each line graph shows the average values of the accuracy and the minimum recall that are obtained for the different α values (α values are ordered on the x-axis from lowest to highest). The thickness of each point of the line graph indicates the average of the standard deviation obtained for that measurement. The thicker the point, the higher the average standard deviation.

From Fig. 6 we can infer the following:

- As the ID of the datasets increases, the average of the minimum recall for all α values decreases. However, the accuracy is kept almost constant for all the IDs.
- In all the datasets, the accuracy is higher for low α values than for high values. In terms of the minimum recall, for low ID datasets, the best results are obtained with low α values. However, as the ID increases, this tendency is reverted and the best results are obtained with the highest α values.
- The standard deviation of the accuracy and minimum recall is higher for high α values than for low values. However, this standard deviation is lower for high ID datasets than for low ID datasets.

We can conclude that the most appropriate α parameter will depend on the ID of the dataset. Specifically, for the datasets with the lowest ID, the best minimun recall results are obtained with low α values. Conversely, as the ID of the datasets increases, the differences between the results obtained with the highest and the lowest α values are reduced. Finally, for the datasets with the highest ID values, this trend is reversed and the best results are obtained with the highest α values. However, if the α value chosen is too high, the accuracy could decrease and the variability of the obtained results will be high.

Finally, from this exploratory analysis we can guess that when the problems are highly imbalanced, the functions that best fit the minimum recall function (functions with high α values) obtain more balanced results in terms of the recalls of the classes. This can be because the more they are similar to the minimum recall function, the stronger they penalize the classifier for obtaining a low recall for the minority class. Thus, the classifier is forced to learn to classify the minority classes accurately in order to reduce the loss function. As a result, the classifier obtains higher minimum recall values than when using low α values.

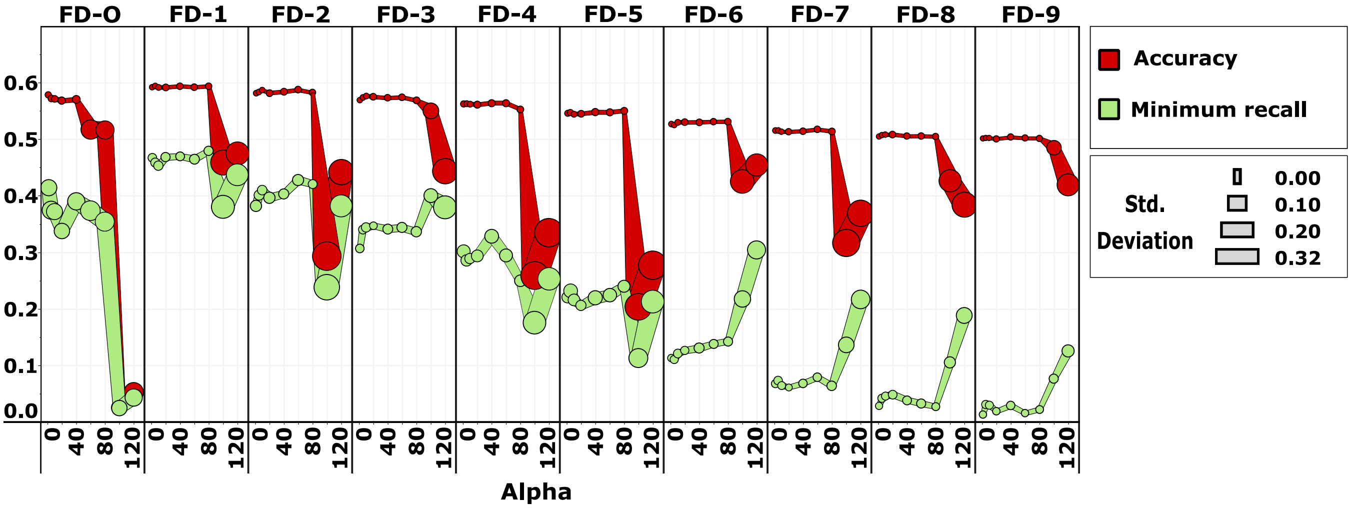


Fig. 6. Set of line graphs with the results of the MLST-FCN classifier with the LSE loss function learned for different α parameters and for datasets with different ID. For each dataset generated from the FD dataset, a line graph with the accuracy and the minimum recall scores obtained when varying the values of the α parameter from 1 to 120 is presented. The thickness of each point of the line graph indicates the average value of the standard deviation obtained for that measurement.

V. Conclusion

The objective of this paper is to develop a method for addressing ITSC problems which can be applied to not only unidimensional or binary problems, but also to multidimensional or multiclass problems. With this in mind, we propose to learn neural network classifiers for time series that maximize the minimum recall of the classes, instead of the accuracy which is what most of the traditional classifiers do. Nevertheless, traditional gradient-based learning methods for learning neural network classifiers require a differentiable loss function [30]. Since the minimum recall is not differentiable, we cannot straightforwardly apply these methods. To solve this issue, we propose to replace the minimum recall function with different smooth approximations (differentiable) of it. Thus, the resulting functions could be used as the functions to optimize when learning a neural network classifier. Thanks to this, we manage to obtain classifiers that, despite the class-imbalance, tend to maximize at the same time the recall value of all the classes [29].

The proposed smooth approximations are experimentally evaluated and compared with each other and with other state-of-the-art methods to deal with the imbalanceness on datasets with different imbalance degree. The results obtained show that the MLSTM-FCN classifier with the LSE loss function successfully achieves our objective and outperforms the other methods in most cases, obtaining the highest minimum recall values without resulting in a decrease of the overall accuracy. In addition, since the LSE loss function is generated from a parametric approximation of the minimum recall function, the sensitivity of this parameter α on the results of the learned classifiers is analyzed. We conclude that, the higher the ID of the dataset, higher values for the parameter α are required for obtaining adequate results. However, excessively high values of the parameter α could increase the variability of the results and decrease the accuracy.

Consequently, future work could focus on developing a method for selecting the best parameter α for the LSE parametric approximation function in order to obtain classifiers that properly classify the minority classes, but without overly affecting the accuracy of the majority classes. Moreover, it should be noted that, although it is presented specifically for ITSC problems, this approach is applicable to other problems and types of data apart from time series. Consequently, another possible research line could be to try to apply our approach to other scenarios with imbalance data, such as, imbalanced (medical) image classification problems.

References

[1] B. Krawczyk, "Learning from imbalanced data: Open challenges and future directions," *Prog. Artif. Intell.*, vol. 5, no. 44, pp. 221–232, 2016.

[2] M. M. Botezatu, I. Giurgiu, J. Bogojeska, and D. Wiesmann, "Predicting disk replacement towards reliable data centers," in *Proc. 22nd ACM Int. Conf. Knowl. Discov. Data Mining*, San Francisco, CA, USA, 2016, pp. 39–48.

[3] H. Yi, Q. Jiang, X. Yan, and B. Wang, "Imbalanced classification based on minority clustering smote with wind turbine fault detection application," *IEEE Trans. Ind. Informat.*, vol. 17, no. 9, pp. 5867–5875, Sep. 2021.

[4] J. Ortigosa-Hernández, I. Inza, and J. A. Lozano, "Measuring the class-imbalance extent of multi-class problems," *Pattern Recognit. Lett.*, vol. 98, pp. 32–38, 2017.

[5] I. Cordón, S. García, A. Fernández, and F. Herrera, "Imbalance: Oversampling algorithms for imbalanced classification in R," *Knowl.-Based Syst.*, vol. 161, pp. 329–341, 2018.

[6] M. Canizo, E. Onieva, A. Conde, S. Charramendieta, and S. Trujillo, "Real-time predictive maintenance for wind turbines using Big Data frameworks," in *2017 IEEE Int. Conf. Prognostics Health Manage.*, 2017, pp. 70–77.

[7] J. Ircio, A. Lojo, J. A. Lozano, and U. Mori, "A multivariate time series streaming classifier for predicting hard drive failures [application notes]," *IEEE Comput. Intell. Mag.*, vol. 17, no. 1, pp. 102–114, Feb. 2022.

[8] H. Cao, X.-L. Li, D. Y.-K. Woon, and S.-K. Ng, "Integrated oversampling for imbalanced time series classification," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 12, pp. 2809–2822, Dec. 2013.

[9] F. Karim, S. Majumdar, H. Darabi, and S. Harford, "Multivariate LSTM-FCNs for time series classification," *Neural Netw.*, vol. 116, pp. 237–245, 2019.

[10] H. Cao, X.-L. Li, Y.-K. Woon, and S.-K. Ng, "SPO: Structure preserving oversampling for imbalanced time series classification," in *Proc. IEEE 11th Int. Conf. Data Mining*, 2011, pp. 1008–1013.

[11] Z. Gong and H. Chen, "Model-based oversampling for imbalanced sequence classification," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, 2016, pp. 1009–1018.

[12] G. Liang, "An effective method for imbalanced time series classification: Hybrid sampling," in *Proc. Australas. Joint Conf. Artif. Intell.*, Springer, 2013, pp. 374–385.

[13] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.

[14] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IEEE World Cong. Comput. Intell.)*, 2008, pp. 1322–1328.

[15] H. Cao, V. Y. Tan, and J. Z. Pang, "A parsimonious mixture of gaussian trees model for oversampling in imbalanced and multimodal time-series classification," *IEEE Trans. Neural Netw. Learn.*, vol. 25, no. 12, pp. 2226–2239, Dec. 2014.

[16] Y. Geng and X. Luo, "Cost-sensitive convolution based neural networks for imbalanced time-series classification," 2018, *arXiv:1801.04396*.

[17] Z.-H. Zhou and X.-Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 1, pp. 63–77, Jan. 2006.

[18] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning From Imbalanced Data Sets*, vol. 11. Berlin, Germany: Springer, 2018.

[19] V. Raj, S. Magg, and S. Wermter, "Towards effective classification of imbalanced data with convolutional neural networks," in *Proc. IAPR Workshop Artif. Neural Netw. Pattern Recognit.*, Springer, 2016, pp. 150–162.

[20] Y. Ho and S. Wookey, "The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling," *IEEE Access*, vol. 8, pp. 4806–4813, 2019.

[21] C. Zhang, G. Wang, Y. Zhou, and J. Jiang, "A new approach for imbalanced data classification based on minimize loss learning," in *Proc. IEEE 2nd Int. Conf. Data Sci. Cyberspace*, 2017, pp. 82–87.

[22] M. A. Rahim and H. M. Hassan, "A deep learning based traffic crash severity prediction framework," *Accident Anal. Prevention*, vol. 154, 2021, Art. no. 106090.

[23] Z. Yang, Q. Xu, X. Cao, and Q. Huang, "Learning personalized attribute preference via multi-task AUC optimization," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 5660–5667.

[24] G. Wang, K. W. Wong, and J. Lu, "AUC-based extreme learning machines for supervised and semi-supervised imbalanced classification," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 51, no. 12, pp. 7919–7930, Dec. 2021.

[25] X. Xu et al., "MSCS-DeepLN: Evaluating lung nodule malignancy using multi-scale cost-sensitive neural networks," *Med. Image Anal.*, vol. 65, 2020, Art. no. 101772.

[26] N. Nasalwai, N. S. Punn, S. K. Sonbhadra, and S. Agarwal, "Addressing the class imbalance problem in medical image segmentation via accelerated tversky loss function," in *Proc. Pacific-Asia Conf. Knowl. Discov. Data Mining*, Springer, 2021, pp. 390–402.

[27] Q. Yan and Y. Cao, "Optimizing shapelets quality measure for imbalanced time series classification," *Appl. Intell.*, vol. 50, no. 2, pp. 519–536, 2020.

[28] G. Liang and C. Zhang, "An efficient and simple under-sampling technique for imbalanced time series classification," in *Proc. 21st ACM Int. Conf. Inf. Knowl. Manage.*, Maui, HI, USA, 2012, pp. 2339–2342.

[29] J. Ortigosa-Hernández, I. Inza, and J. A. Lozano, "Towards competitive classifiers for unbalanced classification problems: A study on the performance scores," 2016, *arXiv:1608.08984*.

[30] S. Sun, Z. Cao, H. Zhu, and J. Zhao, "A survey of optimization methods from a machine learning perspective," *IEEE Trans. Cybern.*, vol. 50, no. 8, pp. 3668–3681, Aug. 2020.

[31] M. Cuturi and M. Blondel, "Soft-DTW: A differentiable loss function for time-series," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2017, pp. 894–903.

[32] M. Lange, D. Zühlke, O. Holz, T. Villmann, and S.-G. Mittweida, "Applications of lp-norms and their smooth approximations for gradient based learning vector quantization," in *Proc. Eur. Symp. Artif. Neural Netw.*, 2014, pp. 271–276.

[33] A. H. Ribeiro, K. Tiels, L. A. Aguirre, and T. Schön, "Beyond exploding and vanishing gradients: Analysing RNN training using attractors and smoothness," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2370–2380.

[34] B. Gao and L. Pavel, "On the properties of the softmax function with application in game theory and reinforcement learning," 2017, *arXiv:1704.00805*.

[35] A. J. Bagnall, J. Lines, W. Vickers, and E. J. Keogh, "The UEA & UCR time series classification repository," [Online]. Available: www.timeseriesclassification.com

[36] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller, "Deep learning for time series classification: A review," *Data Mining Knowl. Discov.*, vol. 33, no. 4, pp. 917–963, 2019.

[37] J.-S. Ang, K.-W. Ng, and F.-F. Chua, "Modeling time series data with deep learning: A review, analysis, evaluation and future trend," in *Proc. IEEE 8th Int. Conf. Inf. Technol. Multimedia*, 2020, pp. 32–37.

[38] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *Proc. Int. Joint Conf. Neural Netw.*, 2017, pp. 1578–1585.

[39] H. Ismail Fawaz et al., "InceptionTime: Finding alexnet for time series classification," *Data Mining Knowl. Discov.*, vol. 34, no. 6, pp. 1936–1962, 2020.

[40] A. Dempster, F. Petitjean, and G. I. Webb, "ROCKET: Exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining Knowl. Discov.*, vol. 34, no. 5, pp. 1454–1495, 2020.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, San Diego, CA, USA, 2015.

[42] A. Benavoli, G. Corani, and F. Mangili, "Should we really use post-hoc tests based on mean-ranks?," *J. Mach. Learn. Res.*, vol. 17, no. 5, pp. 1–10, 2016.

[43] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci.*, vol. 180, no. 10, pp. 2044–2064, 2010.

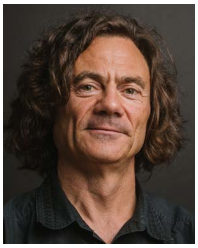
[44] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning," *J. Mach. Learn. Res.*, vol. 18, no. 17, pp. 1–5, 2017.

Josu Ircio received the bachelor's degree in mathematics and the MSc degree in mathematical modelling and research, statistics and computing from the University of the Basque Country, in 2014 and 2015 respectively. He is currently working toward the PhD degree with the Ikerlan Research Center, Spain. His research interests include data analytics, supervised classification and multivariate time series analysis.

Aizea Lojo received the MS degree in Advanced Computer Systems (UPV/EHU), and the PhD degree in computer science in web mining and web recommendation/adaptation systems from the University of the Basque Country, in 2015, as a Basque Government grant holder. She is the author of numerous scientific publications in journals with high impact. She has been with IKERLAN since 2015. In the few last years, her research activities have been focused on CBM, Machine Learning, data mining applied to real industrial problems. Currently, she is working in the group of Data Analytics and Artificial Intelligence (DAI) within IKERLAN, participating in several industrial real projects, involved in several European Research projects leading work packages and tasks and also taking part in several R&D projects funded by the Spanish/Basque Governments.

Usue Mori received the MSc degree in mathematics, and the PhD degree in computer science from the University of the Basque Country UPV/EHU, Spain, in 2010 and 2015, respectively. Since 2015, she has been working as a lecturer with the Department of Computer Science and Artificial Intelligence, University of the Basque Country UPV/EHU. Her main research interests include clustering and classification of time series.

Simon Malinowski received the BS and PhD degrees in telecommunications from the University of Rennes, Rennes, France, in 2005 and 2008, respectively. He was then a postdoctoral researcher with INESC TEC, Porto, Portugal, and FEMTO-ST, Besançon, France, and an assistant professor with the AgroCampus Ouest, Rennes, France. He is currently an assistant professor with the Institut de Recherche en Informatique et Systèmes Aláatoires, University of Rennes 1, Rennes, France. His research interests include signal processing, data mining, and, more particularly, time-series analysis and mining. He is a regular reviewer for international journals related to the fields of pattern recognition and knowledge discovery.



Jose A. Lozano (Senior Member, IEEE) received the PhD degree in computer science from the University of the Basque Country UPV/EHU, Donostia, Spain, in 1998. Since 2008, he has been a full professor with the University of the Basque Country UPV/EHU, where he currently leads the Intelligent Systems Group. He is the co-author of more than 100 ISI journal publications and a co-editor of three books. His current research interests include machine learning, pattern analysis, evolutionary computation, datamining, metaheuristic algorithms, and real-world applications. He is an associate editor of the *IEEE Trans. on Neural Networks and Learning Systems*.