A Multivariate Time Series Streaming Classifier for Predicting Hard Drive Failures

Josu Ircio, Aizea Lojo, Usue Mori, and Jose A. Lozano,

***Abstract*—Digital data storage systems such as hard drives can suffer breakdowns that cause the loss of stored data. Due to the cost of data and the damage that its loss entails, hard drive failure prediction is vital. In this context, the objective of this paper is to develop a method for detecting the beginning of hard drive malfunction using streaming SMART data, allowing the user to take actions before the breakdown occurs. This is a challenging task for two main reasons. First, there are not usually many examples of failed hard drives. Second, in these few available examples, hard drives are only identified and labeled as failed after complete breakdown occurs, but the exact moment when they begin to malfunction is usually unknown. Both these aspects significantly complicate the supervised learning of hard drive failure prediction models. To cope with these issues, the problem is addressed as a multidimensional time series streaming classification problem based on sliding windows. Moreover, as a solution to the highly imbalanced situation, the learned classifier is optimized to maximize the minimum recall of classes. Experimental results using the Backblaze benchmark dataset show that the proposed method reliably anticipates hard drive failures and obtains a higher balance between the recall values of both classes, failed and correct disks, compared to other state-of-the-art solutions.**

I. Introduction

WITH the digital transformation of information and the popularization of the cloud computing platforms, data storage centers have become essential. Data centers are mainly composed of hard drives, which are the devices used for data storage. These hard drives can suffer breakdowns that cause the loss of the stored data and, in many cases, a chain reaction of the rest of the equipment failures which can result in the downtime of the entire data center. According to a study reported in [1], the average cost of a data center downtime can be up to $740,357 including the costs derived from the data loss or corruption, productivity loss, equipment damage and recovery, and legal and reputation repercussions. Therefore, predicting upcoming failures in hard drives is crucial: it allows users to take actions before a failure occurs, such as moving their data to other storage systems. With this in mind, the objective of this paper is to develop an effective method to detect malfunctions in hard drives so that the user can take preventive measures.

Due to the importance of anticipating breakdowns in hard drives, the Self-Monitoring and Reporting Technology (SMART) system has been developed [2]. It is a system implemented within hard drives which reports information about their functioning. The SMART system provides attributes such as the number of uncorrectable software read errors, the count of aborted operations and the temperature of the device. A change in the values of these reported attributes could indicate defects of the disk and/or problems in the mechanical subsystem. Consequently, hard drive manufacturers use a simple algorithm that raises an alarm whenever any of the SMART attributes exceeds a predefined threshold. Unfortunately, this method is not able to obtain high failure prediction rates [3]. Therefore, in order to try to improve its results, several methods and techniques have been implemented over the past decade [4]–[7].

There are three major approaches that have been applied for hard drive failure prediction: the first is to try to predict the remaining useful life (RUL) of the drives as a forecasting problem [8]–[11]. Specifically, current and past observations are used to learn a regression model that predicts the expected lifetime of the disk. However, both the shortage of failure disk examples and the fact that the life expectancy depends directly on the environment and conditions of the operation, make it difficult to generalize and learn a unique model for different disks.

The second approach is to predict the health status of the drive as a multi-class classification problem [11]–[13]. To this end, the multiple classes are defined as different health degrees, which are established based on the remaining time until failure. In particular, the highest degree indicates that the disk is working properly, and for the remaining grades, the lower the grade, the closer failure is, with the lowest grade indicating imminent failure. It should be noted that this strategy is equivalent to discretizing the previously described approach and it therefore has the same difficulties. In addition, since the failure occurrence is uncommon, the percentage of correct and failed disks is very imbalanced and, by generating more classes, this problem is exacerbated.

Finally, the third approach, and the most common one, models the failure prediction problem as a binary classification problem on the hard drive status (correct or failed) [14]–[17]. In this case, the disk is classified as failed if signs of deterioration, which indicate that it will fail in the near future, are detected. This is the approach adopted in this paper.

Although this third approach is the most common one, some weaknesses can also be found among the existing methods. For instance, in [3], [15], [17], [18], the lead-time to failure is set in advance and it is considered that all the drives show signs of malfunction with the exact same anticipation. Finding an appropriate lead-time that is suitable for all the hard drives is problematic because life expectancy depends directly on the

J. Ircio and A. Lojo, Ikerlan Technology Research Centre, Spain.
Jose A. Lozano and U. Mori, University of the Basque Country, Spain.
Jose A. Lozano, Basque Center for Applied Mathematics (BCAM), Spain.
Manuscript received November 23, 2020;

conditions and the environment in which the disk operates and these are not usually the same. Consequently, a method is designed that tries to identify the first signs of malfunction in each disk in an automatic way, without pre-setting a unique lead time for all disks.

Another shortcoming is that some methods do not take into account the temporal correlation between the measurements taken by the SMART system [3], [17], [19]. They classify each observation (timestamp) individually as malfunctioning or correct, without considering the temporal nature of the data at hand. In order to solve this, there are methods that group the observations using sliding windows. Then, they apply strategies such as majority voting to decide whether a data-window is showing signs of deterioration [14]–[16], [20]. However, the classification of each observation is still carried out individually, without considering the adjacent observations and the temporal relationship that can exist between them.

In the proposed method, the measurements of the SMART system are treated as multivariate time series (MTS). In this way, it is taken advantage of the temporal information contained in the signal. Additionally, in a real context, the full MTS is not available from the beginning. That is, the complete operating data from when the hard drive is put into operation until it breaks down is not known beforehand. On the contrary, at each instant of time, new observations (MTS time steps) arrive that gradually gather the MTS. This means that, in a real context, data is a MTS stream and specific methods to cope with this type of data will be required. Therefore, preprocessing the training MTS data is proposed, in which the complete operation of the hard drives is already collected, using sliding windows. Each window captures a subsequence of the MTS, so it still has a temporal nature and can be considered as a MTS. The objective is to learn a binary classifier that identifies whether a window of observations shows signs of malfunction (class 1) or not (class 0). In this way, the problem is reduced to a MTS classification problem which takes into account the temporal correlation and the streaming nature of the observations.

Learning a classifier in this context is complicated for two main reasons. Firstly, the number of windows with signs of malfunction are limited because the failure occurrence is uncommon. This accounts for very imbalanced class frequencies, which makes the learning process hard. Secondly, even if the failed disks are labeled as such, the signs of future failure do not appear throughout the lifetime of the disks. Thus, failed disks can be composed of correct windows (usually at the beginning of the operation) and malfunction windows (usually sometime before the failure). The exact window in which the disk begins to show signs of malfunction is usually not known and is difficult to identify: a windows labeling process is necessary in order to train a supervised window-based classifier.

With this in mind, a framework that deals with each of these challenges is developed:

- Identification of windows with signs of upcoming failure. A double-round learning methodology is proposed. A first classifier is built using only the windows which can be labeled reliably (i.e., the last window in the case of failed disks and the first window in the case of correct disks). This classifier is used to label all the windows of all the training streams, which are then used to build a second and final classifier.
- Imbalance. Using the minimum recall of the classes as the function to maximize when learning a classifier is proposed, rather than the accuracy that is typically used. Thanks to this, classifiers are obtained which, in spite of the imbalance, tend to equilibrate the recall value of both classes (malfunction and correct).

The proposed method is applied to the Backblaze[1] datasets. The results obtained show that the method is able to balance the prediction of correct and failed drives, obtaining high true positive rates with almost the same ratio of false positives as the competing algorithms.

The rest of this paper is structured as follows. In Section II, the proposed approach for solving the hard drive failure prediction problem and the challenges found are detailed. In Section III, the experimental framework is introduced, followed by the obtained results and discussion in Section IV. Finally, in Section V, the conclusions and future work are presented.

II. Proposed Method

The problem of anticipating hard disk failures is addressed as shown in the scheme presented in Figure 1. The observations from the hard drives are treated as MTS streams and they are analyzed using sliding windows. The objective is to predict when the disk works correctly or malfunctions in a particular window. However, the uncertainty regarding the location of the malfunctioning windows and the imbalance between the classes complicate the learning of this classifier. This section presents the proposed framework for predicting hard drive failures and dealing with these two issues. Finally, an explanation is given as to how the resulting classifier is deployed in a streaming manner to identify when new unlabeled hard drives begin to malfunction.

A. Double-Round Learning Methodology

As stated previously, the objective is to learn a window-based classifier which will predict whether a window of the MTS, which represents the current status of a hard drive, is showing signs of malfunction or not. To this end, a training database of windows with known class labels is needed. However, what is normally available is a training set of hard drives which are labeled as correct (they do not fail throughout the time of analysis) or failed (they break down some time during the analysis), and not the window-based labels. In the disks with no recorded failure (correct disks), it can be assumed that all the windows are correct. Nevertheless, in a failed disk, it can be confidently considered that when it is initially put into operation it works correctly for a while. Then, at a certain moment, it begins to show signs of malfunction until it suffers a breakdown and stops working. Consequently, it can be assumed that the first window (when the disk is

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

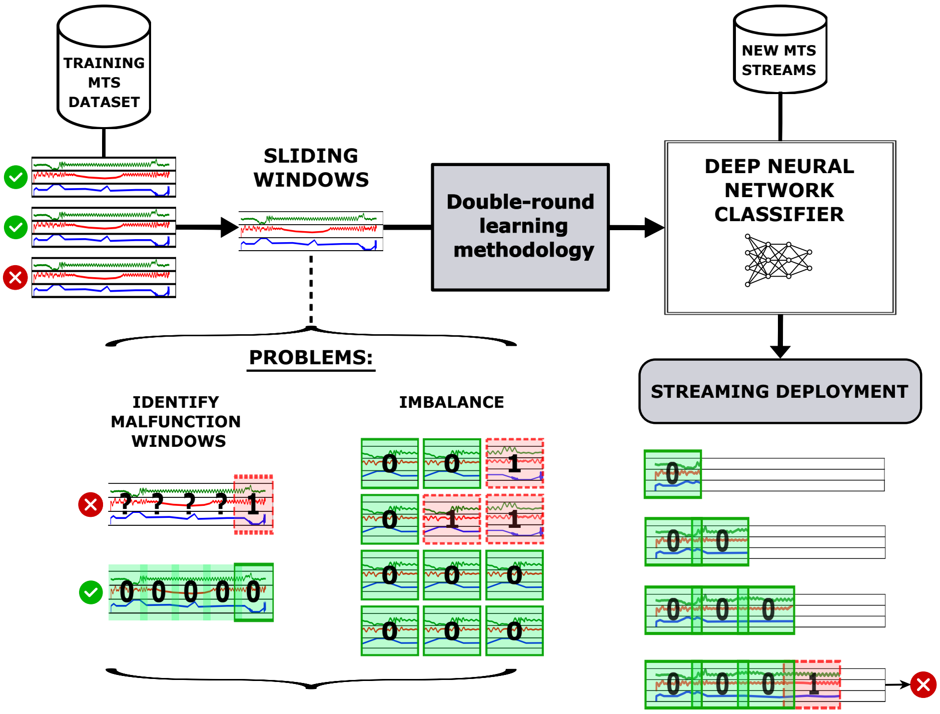


Fig. 1. Graphical description of the failure prediction problem.

put into operation) is a correct window (class 0) and that the last window, just before the breakdown, shows signs of malfunction and it is a malfunction window (class 1) (see the problems highlighted in Figure 1). Since the moment when the signs of malfunction begin to appear is unknown, the label of the rest of the windows is unknown. However, in order to learn a classifier which is able to detect windows with signs of malfunction, it is mandatory to have examples of these kind of windows. Therefore, for failed disks, a labeling process is necessary to identify those windows.

A double-round learning methodology is proposed to solve this problem. It is thus named because the training process is done in two stages (See Figure 2), which are explained below:

1.1 **Generate a training dataset with one window from each hard drive:** In this first step, the purpose is to generate a dataset in which the correct and malfunction windows are clear cases of correct operation and malfunction. It is assumed that this occurs when the correct windows are selected from the beginning of the operation of the disk and when the malfunction windows are selected from just before the failure occurs. Therefore, for correct hard drives, the first window of observations is selected and, in the case of failed hard drives, the last window of observations is chosen. Consequently, the generated training dataset comprises only one window of each disk available and will be referred to as One-window dataset.

1.2 **Train the first classifier:** The first classifier is trained with the dataset created in the previous step, the One-window dataset, (the details about the classifiers used are explained in Section II-C).

2.1 **Apply the classifier 1 to all the windows of the failed disks and create a new training dataset:** The classifier trained in the previous step is applied to all the windows of the failed disks of the training dataset. Contrary to other proposals in the literature, which preset a common lead time value for all disks, this is done to attempt to automatically detect when the disks begin to malfunction and label the windows of the failed disks. Specifically, the purpose of this step is to identify representative windows that show signs of malfunction and are not at the end of the MTS.

For the failed disks in which the last window is not correctly classified (is not classified as 1) by the classifier trained in the previous step, all the windows classified as malfunction are not considered for selection (see disk 3 of Step 2.1 in Figure 2). Since, in these cases, the classifier is not able to correctly classify the last window before the breakdown, which it is previously assumed has clear signs of malfunction, selecting the other windows classified as malfunction is avoided just in case the classifier is also failing. However, although the first classifier does not correctly classify these last windows, they are added as malfunction windows so as not to lose the information they contain (see disk 3 of Step 2.1 in Figure 2).

In contrast, for the failed drives in which the last window is classified as 1, all consecutive windows that have been classified as malfunctioning starting backwards from the last window are selected as malfunction windows for the new training set (see disks 1 and 2 of Step 2.1 in Figure 2). Moreover, considering that the failed drives should also operate correctly at the beginning, for these

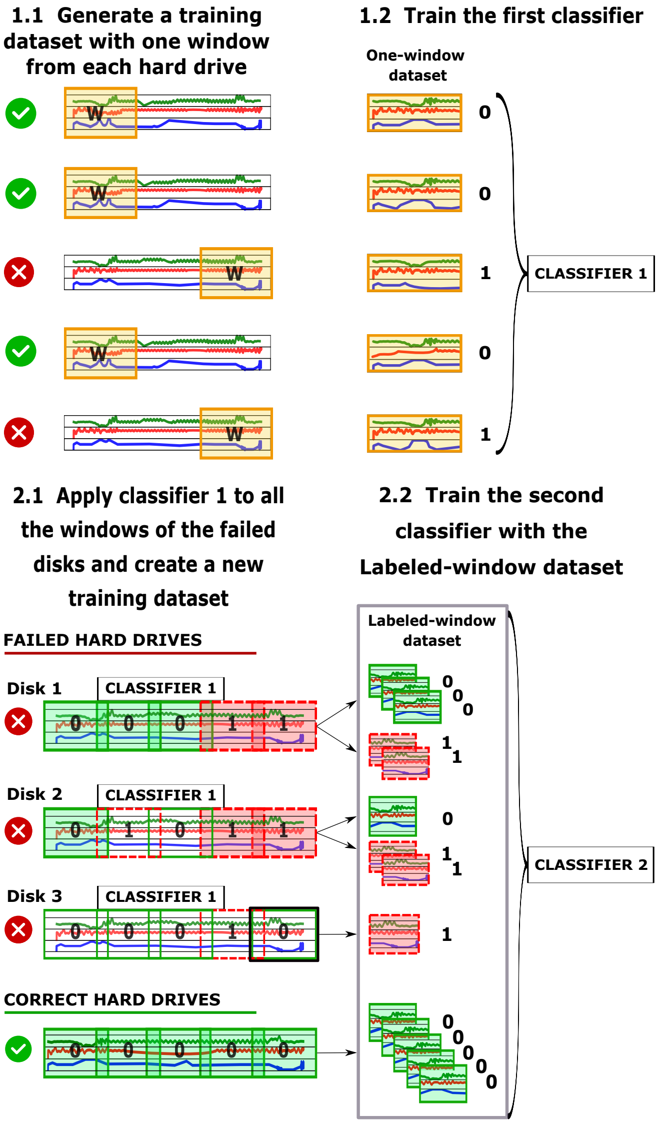

1.1 Generate a training dataset with one window from each hard drive
W
1.2 Train the first classifier
One-window dataset
0
0
1
0
1
CLASSIFIER 1
2.1 Apply classifier 1 to all the windows of the failed disks and create a new training dataset
FAILED HARD DRIVES
Disk 1
CLASSIFIER 1
0 0 0 1 1
Disk 2
CLASSIFIER 1
0 1 0 1 1
Disk 3
CLASSIFIER 1
0 0 0 1 0
CORRECT HARD DRIVES
0 0 0 0 0
2.2 Train the second classifier with the Labeled-window dataset
Labeled-window dataset
0 0 0
1 1
0
1 1
1
0 0 0 0 0
CLASSIFIER 2

Fig. 2. Diagram representing the steps of the proposed double-round learning methodology.

disks, all the windows consecutively classified as correct windows located before the first window classified as malfunctioning are also selected for the new dataset (see disks 1 and 2 of Step 2.1 in Figure 2). In this case, the correct class (0) is assigned to them. In order to discard possible windows in which the classifier may be failing, windows that are between the first malfunction window and the last correct window inclusive are discarded (see the second window in disk 2 of Step 2.1 in Figure 2).

Correct hard drives do not need to undergo any window-labeling process because all the windows are considered correct windows. To include different examples of correct windows (not only from the beginning of the disks operation) from the correct hard drives, other correct windows are also added to the new dataset. However, by taking all the sliding windows from the correct disks, the original imbalance rate between failed and correct disks excessively increases. Therefore, to maintain the original imbalance rate, correct hard drives are randomly sampled and then all their windows are added to the new dataset.

This new dataset will be called Labeled-window dataset.

2.2 **Train the second classifier with the Labeled-window dataset:** The second classifier is trained with the dataset generated in the previous step, the Labeled-window dataset. This dataset, besides the correct and malfunction windows which are assumed to be correctly labeled, has windows that are identified as malfunction windows which are not at the end of the MTS and the windows identified as correct windows which are at the beginning of the failed hard drives. Thanks to this, the new classifier should be able to detect the beginning of the malfunction. Consequently, it will be the classifier used to classify windows in streaming mode.

B. Minimum Recall Based Function to Deal with the Imbalanced Scenario

One of the main difficulties when predicting disk failures is that the correct disks are, by far, the largest portion of all the available disks, while the failed disks are the minority. As seen in the previous section, this problem is even more evident when using sliding windows. As a result, the most common classifiers, those based on maximizing the classification accuracy, either tend to over-fit the minority class or completely ignore it [21], [22].

		True label	
		Correct	Failed
Predicted label	Correct	TC	FC
	Failed	FF	TF

$$\textbf{Accuracy} = \frac{TC+TF}{TC+FC+TF+FF}$$

$$\textbf{Recall}_{\textbf{C}} = \frac{TC}{TC+FF}$$

$$\textbf{Recall}_{\textbf{F}} = \frac{TF}{TF+FC}$$

Fig. 3. Accuracy and recall measures for the failure prediction problem.

While many alternatives have been given in the literature to alleviate the imbalance problem, such as oversampling, undersampling or cost-sensitive classification [23]–[25], this work follows a completely new avenue. To deal with this situation, when training a classifier, it is proposed to maximize the minimum recall of the classes rather than the accuracy that is typically used (Figure 3 shows the definition of the accuracy and the recall) as suggested in [26]. Owing to this, equal importance is given to both classes, failed and correct disks, and classifier are obtained that tend to equilibrate the recall value of both classes. As a result, even if the accuracy decreases, more failed disks are predicted without resulting in the prediction of excessive false positives.

C. Deep Neural Network Classifier for MTS Genetically Optimized for Maximizing the Minimum Recall

It should be noted that learning most of the traditional classifiers 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 easily changed. With this in mind, in the following sections, the solution developed to learn a neural network classifier that maximizes the minimum recall is presented.

1) Network Architecture Optimization: The network architecture has a significant influence on the performance of the final classifier, and there are many alternatives and options that could be considered. For example, some factors that affect the obtained results are the depth of the architecture, the type of layers that compose the architecture or the parameters related to each specific type of layer. Consequently, designing them to fit a specific problem is not an easy task [27].

In [28], a Genetic Algorithm (GA) is proposed for automatically designing the network architecture. Specifically, each individual of the GA is a network architecture that is generated by a sequence of blocks of different types of layers. Moreover, each block is composed of one or more layers of the same type and each type of layer has its own parameters. On that account, each individual is defined by establishing (in this order) the number of blocks of the architecture, the type and the number of layers of each block and, finally, the value of the parameters for each layer in the blocks. All the blocks have two parameters, the input size and the output size. The first represents the dimensions of the input data and the second, the dimension of the output data after going through the layers of the block. If the architecture has more than one block, the output size of a block will have to be equal to the input size of the next one. Within the blocks, the layers that compose them must fulfill this same chain condition with their input and output sizes.

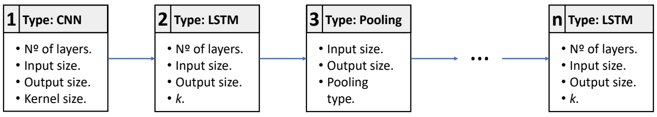


Fig. 4. Example of an individual of the GA used for optimizing the network architecture of the classifier. The individual is defined by a sequence of blocks of different types of layers (LSTM, CNN, pooling) and the necessary parameters for defining each one.

This approximation is taken as a baseline, but, in this case, the observations of hard drives are considered as MTS. Therefore, for building the network architecture, it is necessary to use specific layers for MTS [29]–[32] instead of the original layers that, in [28], were specifically for images. In particular, the layers that are established for designing the blocks of the network architecture will be Long Short-Term Memory Network (LSTM), Convolutional Neural Network (CNN) and Pooling layers (Figure 4 shows an example of an individual of this GA). In addition, there are certain specific parameters for each type of block that must also be defined:

- LSTM blocks need k, which is a parameter that determines the output size of each LSTM layer in the block.
- CNN blocks need the kernel size for all the convolutional layers.
- Pooling blocks need the type of pooling (maximization or averaging pooling) to be carried out. The window length and the stride for both pooling types will be equal to 2, hence the output size will be half of the input size.

Apart from the changes in the network architecture options, the main change from the original proposal is made in the fitness function used to evaluate and compare the individuals of the GA. Based on the proposed alternative to deal with the imbalanced scenario, the minimum recall is used as the fitness function rather than the accuracy. However, at the beginning of the training process, due to the highly imbalanced situation, it is common that many of the individuals of the GA define classifiers that simply assign all the disks to the majority class. Therefore, if one simply aims at maximizing the minimum recall of the classes, a great quantity of individuals will not be comparable because they will have the same fitness value (they will all have a minimum recall of 0). This greatly complicates the search of the optimal individual. The quality of these individuals thus needs to be ranked in order to guide the search of the GA from the beginning.

To address this problem, the use of a probabilistic recall that is an adaptation of the Brier score [33] is proposed. In a binary classification problem with N instances of which J are instances of class 0 and K are instances of class 1 ($N = J + K$), the probabilistic recall for each class is defined as follows:

$$\text{Prob. recall}_0 = \frac{1}{|J|}\sum_{j\in J}(f_j - 0)^2$$

$$\text{Prob. recall}_1 = \frac{1}{|K|}\sum_{k\in K}(f_k - 1)^2$$

where f_i is the predicted probability of failure for the i^{th} instance. Note that, contrary to recall, the greater the probabilistic recall is, the worse the prediction is. Because of this, to maintain the consistency with the objective of maximizing the minimum recall, the sign of the function is changed. Consequently, minus the probabilistic recall between the classes will be used in the proposed fitness function and, thus, the objective will be also to maximize the minimum of it. In summary, the proposed fitness function, f, for the GA is established as follows:

$$f = \begin{cases} \min\left(\text{Recall}_0, \text{Recall}_1\right) & \min\left(\text{Recall}_0, \text{Recall}_1\right) \neq 0 \\ \min\left(-\text{Prob. recall}_0, -\text{Prob. recall}_1\right) & \min\left(\text{Recall}_0, \text{Recall}_1\right) = 0 \end{cases} \tag{1}$$

As can be seen, the minimum recall will be used in the cases when this value is not 0, and for the individuals with a minimum recall of 0, the probabilistic recall will be used. Note that when maximizing this function, an individual with a positive minimum recall will always be preferred to an individual with a minimum recall of 0.

In addition, some minimal modifications are made in the original crossover and mutation processes of the GA to allow them to work with the new MTS specific block types and their parameters, but without altering the original operations. Finally, an elitist truncation method for the offspring selection process and the random selection method for the parents selection process are used.

During the search process of the GA, the weights of each individual need to be learned. As long as the loss function is differentiable, traditional gradient-based learning methods for deep neural networks [34] can be used for this task. Ideally, the proposed function (1) would be plugged into any of these common learning methods, but unfortunately, since it is not differentiable, this cannot be done. Instead, this first GA will only focus on optimizing the architecture with respect to function (1) and the weights will be optimized using the classical stochastic optimizer called Adam [35], which uses the cross entropy as a loss function.

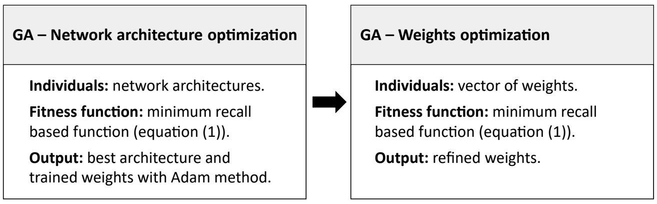
In conclusion, the network architecture obtained through the GA will be specific for time series and will be the one whose value for the proposed fitness function (function (1)) is the maximum of all those analyzed by the GA.

2) Weights Optimization: Similar to accuracy, the cross entropy used in the GA of the previous step for learning the weights of the selected network architecture is not designed for imbalanced scenarios. Therefore, it is proposed to post-optimize the weights of the selected network architecture using function (1) which is more suitable for this highly imbalanced problem. Since, as mentioned above, function (1) is not differentiable, it is referred again to GAs.

In this second GA, the individuals will be vectors of weights. The process of generating the initial population of the GA consists of taking the vector of weights learned in the previous step for the best architecture, and applying multiple random perturbations to it. In this way, the weight vectors are explored, which are similar to those obtained previously with the Adam optimizer (resulting from the first GA), but which will attempt to maximize the function (1).

Regarding the rest of the genetic operations, the following methods are used: the one-point crossover method [36], the multiple random perturbations method (which is previously applied to generate the initial population) as mutation operation, the elitist selection method for the offspring selection process and the random selection method for the parents selection process.

After applying the described learning method, which is comprised of the two GAs, both the obtained network architecture and its weights will be heuristically optimized for maximizing the minimum recall. The schema of this learning method is presented in Figure 5. It should be noted that the two classifiers that are learned in the double-round learning methodology (see Section II-A) will be trained by following this learning method.


GA – Network architecture optimization

Individuals: network architectures.

Fitness function: minimum recall based function (equation (1)).

Output: best architecture and trained weights with Adam method.

➡

GA – Weights optimization

Individuals: vector of weights.

Fitness function: minimum recall based function (equation (1)).

Output: refined weights.

Fig. 5. The schema of the proposed classifier. The first GA selects the best network architecture and the second GA refines the weights of the network architecture selected in the first GA.

D. Deployment of the Proposed Method

In this section, how the final classifier (the classifier 2 in Figure 2) is deployed in streaming mode in a real scenario is explained.

The training phase would be performed offline because it requires historical operation data from multiple hard drives and a certain amount of computational resources. The resulting classifier would be deployed in streaming mode. When the SMART system collects new daily data stream from the hard drive operation, the classifier is applied to the stream. If signs of malfunction are detected, an alert could be reported and the recovery mode of the hard drives could be automatically triggered to prevent data loss or the propagation of the malfunction to other hard drives in the data center.

As can be seen in Figure 6, sliding windows are taken from the beginning and their class is predicted. If the predicted class is 0, a correct window, it should wait to gather new observations, and then it is slid to the next window and the process is repeated. If the predicted class is 1, a malfunction window, the analysis of the disk is stopped. Consequently, it is determined that the disk has signs of malfunction and it is going to fail in the following days.

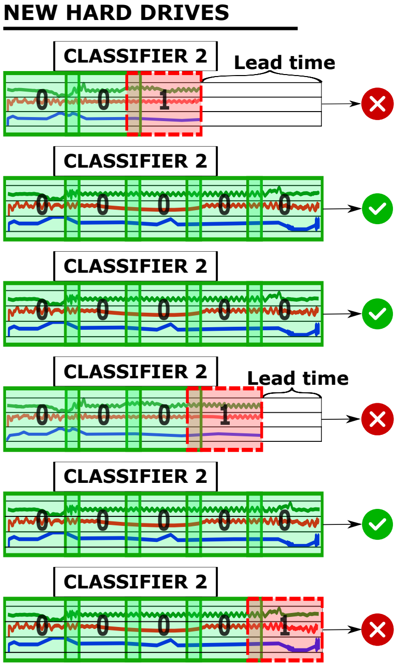


Fig. 6. Diagram of the deployment process followed to apply the final classifier on new hard drives.

III. Experimental Framework

This section describes the benchmark datasets that are used to test the approach, the parameters that need to be established and the experiments to compare it with other related work.

1) ***Benchmark Datasets:*** To evaluate the proposed method, the open data provided by the Backblaze[2] company is used.

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

Backblaze has thousands of spinning hard drives in their cloud storage ecosystem spread across data centers. Every day, a snapshot of each operational hard drive is taken. This snapshot includes basic drive information along with the SMART statistics reported by that drive. As previously stated, each daily snapshot is considered a time step of the corresponding hard drive streaming MTS.

To design the classifier, the set of SMART metrics that, according to the experience of experts, indicate imminent failure[3] are chosen. The selected attributes are listed below:

- SMART 5: Reallocated Sector Count. Represents a count of the malfunctioning sectors that have been found and remapped. A drive which has had reallocations is significantly more likely to fail in the immediate months.
- SMART 187: Reported Uncorrectable Errors. Counts the errors that could not be recovered using the internal error correcting mechanism.
- SMART 188: Command Timeout. Counts the number of aborted operations due to timeout. Values larger than zero may indicate power supply or data cable connection problems.
- SMART 189: High Fly Writes. Counts the errors detected when a recording head is flying out of its normal operating range.
- SMART 197: Current Pending Sector Count. Counts the ”unstable” sectors (waiting to be remapped, because of unrecoverable read errors).
- SMART 198: Offline Uncorrectable. Represents the total count of uncorrectable errors when reading/writing a sector. A rise in the value of this attribute indicates defects of the disk surface and/or problems in the mechanical subsystem.

For each attribute, the raw and the normalized values are reported, which means that every MTS stream contains 12 dimensions.

Backblaze classifies a disk as failed (class 1), when it completely stops working (does not respond to console commands or can not be read or written), or when irrefutable evidence of imminent failure is shown. After the failure, the drive is removed from the list. Otherwise, if the disk operates correctly during all the periods of time available, the class associated to the MTS is “0”. Additionally, new units are introduced regularly into operation.

Reported data for the same SMART attribute can vary in meaning and values, depending on the hard drive model and the manufacturer [13], [22], [37], [38]. Therefore, to ensure that the behavior is homogeneous for all of the disks in the dataset, a single hard drive model is chosen. In particular, all the units of the ST4000DM000 model are selected because it is one of the models with the highest number of units available and, at the same time, it has a high number of recorded failures in comparison with the rest of the models.

Furthermore, the most recent hard drives for the study are chosen. Specifically, the disks that started working in the period between the beginning of 2016 and the end of 2019 are selected. There are also small periods of time when measurements are not recorded. These missing time steps are ignored. Finally, it should also be considered that there are disks that stop recording data even if their failure is not recorded. These hard drives are discarded for not knowing the reason for their removal from Backblaze.

TABLE I
PROPERTIES OF BENCHMARK DATASETS.

Dataset	Train/Valid/Test	Imbalance rate	Correct disks	Failed disks
Dataset 1	$64\%/16\%/20\%$	5%	5748	333
Dataset 2		3%	5748	177
Dataset 3		1%	5748	58

In total, there are 333 failed disks and 5748 correct disks (the imbalance rate is about 5%). To analyze the performance of the method for higher imbalance rates, two new datasets with imbalance rates of 3% and 1% are also created. The procedure for generating these new datasets is the same as that for the original dataset, but the number of failed hard drives is reduced. The reduction is made randomly selecting from the total the number of failed hard drives needed to obtain the desired imbalance rate. For the 3% imbalance rate dataset, there are 177 failed disks; for the 1% imbalance rate dataset, there are 58 failed disks. For each dataset, a stratified partition is carried out by selecting 80% of the hard drives for the training and validation subsets and the remaining 20% for the testing subset. Then, the previously selected 80% is divided into 80% for the training subset and the remaining 20% for the validation subset. The properties of the datasets are summarized in Table I.

Finally, the MTS streams are divided into subsequences through the use of sliding windows. The window size is set to 200 time steps (days) and the slide to 5 time steps (days).

2) *Parameter Settings:*

In order to perform any experiments, it is necessary to define the parameters of the two GAs included in the proposed framework. Recall that each of the GAs is applied twice: once with the initial dataset that comprises only one window for each disk (One-window dataset), and one that is built after the double round labeling of the windows (Labeled-window dataset).

For the first GA, a usual scheme in GAs is followed (see Section II-C1). Specifically, the probabilities of the crossover and the mutation are 0.95 and 0.75, respectively. Each mutation type (addition, alteration and removal) has the same probability of occurrence. Additionally, the population size and the maximum number of generations are set at 20. Each individual is trained by the Adam stochastic optimizer [35] using the cross entropy as a loss function and a maximum of 45 epochs which may be interrupted if the classifier does not improve during 3 epochs. Depending on the dataset to which the method is applied, based on preliminary experiments, different batch sizes and learning rates are used:

- For the One-window dataset, the batch size is set to 60 and the learning rate is initialized to 0.01, decreased to

[3] https://www.backblaze.com/blog/what-smart-stats-indicate-hard-drive-failures/

0.001 during the 36 to 43 epochs and decreased again to 0.0001 during the last two epochs.

- The Labeled-window dataset is larger and more complex than the previous one. Therefore, the batch size is set to 128 and the learning rate is initialized to 0.01, decreased to 0.001 during the 6 to 30 epochs, decreased again to 0.0001 during the 31 to 43 epochs, and for the last two epochs it is set to 0.0001.

Regarding the definition of the individuals, the number of LSTM and CNN blocks is limited to a maximum of 3 (for each type) and to a maximum of 2 for the pooling blocks. In addition, for the convolutional blocks, the number of layers is restricted to values between 1 and 8, the available choices for the output size are 32, 64, 128, 256 or 512 and the kernel size can be 1, 3 or 5 based on the settings used for CNNs in other state-of-the-art works. For the LSTM blocks, the number of layers is restricted to values between 1 and 4 and the output size can take the following values: 64, 128 or 256. All these restrictions are applied to limit the computational resources.

For the second GA, the same schema as for the first one is followed. The parameters for the GA are the same, except for the population size, which is 50 individuals instead of 20.

*3) **Experiments to Compare the Proposed Approach with Other Methods**:* The objective of the experimentation is to validate the proposed method. To this end, the results obtained in the original dataset, Dataset 1 (see Table I), are compared with those obtained in other related works.

In general, the approaches used in the literature are very different in nature, in the type of data they require, in the evaluation metrics they apply, etc. In this sense, carrying out a fair comparison is not always possible [4], [13]. Taking this into account, relevant and recent works that have also addressed the problem as a binary classification problem and are sufficiently detailed to be replicated are selected. The methods chosen for comparison are those described below:

- In [17], each daily vector of SMART measurements is considered an instance of the classification problem. Nine raw SMART features highly correlated to failure events are selected. Then, they define the lead-time as a hyperparameter to optimize and all the observations inside this lead-time are labeled as failed samples. In order to reduce the impact of the class imbalance, they propose using the SMOTE oversampling technique. Finally, logistic regression, SVM, random forest (RF) and gradient boosted tree (GBT) classifiers are applied. They obtain the best results with a lead-time of 20 observations (days) together with the SVM with the selected features and without using the SMOTE, and with the RF and the GBT with all the features and without using SMOTE. These three approaches are applied to the previously described dataset.
- In [16], each daily vector of SMART measurements is considered an instance of the classification problem. As in the previous approach, the lead time is preset, but in this case, sliding windows and a majority voting strategy are implemented for classification. They apply RF, feed-forward neural networks and the k-means clustering algorithm for failure prediction. The best results are achieved with the RF classifier with 7-observations (days) lead-time and with windows of 3 observations for the majority voting strategy.
- In [22], the measurements of the SMART attributes over time are considered a MTS. To address the problem of failure prediction they follow this process: first, they find the subset of SMART attributes indicative of disk replacements by identifying those dimensions of the MTS that have significant change points. After that, they compute a compact representation of the selected dimensions of the full-length MTS stream using exponential smoothing. Each selected dimension of the MTS is compacted into a single point and consequently, each MTS is compacted into a vector. The class assigned to each vector will be the class of the corresponding disk. In order to deal with the imbalanced dataset, they perform down-sampling of the correct hard drives via k-means clustering. Only correct disks that are the nearest neighbors of the k-centroids are selected. Finally, the regularized greedy forests (RGF) classifier is trained.

The best performing parameter combinations and features that are reported in the original papers are directly adopted and applied to the benchmark dataset. For all the methods, the evaluation is done at the disk level, establishing the criterion that a disk is going to fail when they predict an instance or a set of instances as malfunctioning.

*4) **Experiments to Analyze the Performance of the Proposed Method in Datasets with High Imbalance Rates**:* To verify that the proposed framework is effective in highly imbalanced scenarios, it is also applied to the datasets created with imbalance rates of 3% and 1% (see above in Section III-1).

In addition, the double-round learning methodology is applied, but the specific function based on the minimum recall (function (1)) is replaced with accuracy in all the steps in which it was used. Thus, the classifier will be learned to maximize the accuracy rather than maximize the minimum recall. The objective is to verify that, for imbalanced situations, training a classifier to maximize the minimum recall of the classes obtains more balanced results than training it to maximize the accuracy. Henceforth, to abbreviate, the proposed double-round learning methodology with the specific function based on minimum recall (function (1)) is referred to as DRL-Rec and the proposed double-round learning methodology but with the accuracy as DRL-Acc.

IV. Results and discussion

In this section, the obtained results after running the experiments are discussed. It should be noted that, although the proposed approach classifies malfunction and correct windows, the objective is to predict whether the disk is going to fail in the following days or not (see Section II-D). Therefore, the results of the experiments are measured in terms of how well the hard drives are predicted as correct or failed drives.

TABLE II
THE OBTAINED RESULTS BY THE PROPOSED METHOD AND THE STATE-OF-THE-ART METHODS. EACH METHOD IS RUN 10 TIMES. FOR EACH METHOD, THE AVERAGE AND THE STANDARD DEVIATION OF THE ACCURACY, THE RECALLS, THE PRECISION, THE FALSE POSITIVE RATE, THE AVERAGE LEAD TIME OF THE FAILURES AND THE AVERAGE LEAD TIME OF THE FAILURES RESTRICTED TO CORRECTLY PREDICTED HARD DISKS FOR ALL THE RUNS ARE PRESENTED.

Method	**ACC**	**Rec$_0$**	**Rec$_1$**	**Prec.**	**FPR**	**LT**	**RLT**
DRL-Rec	0,94±0,01	0,96±0,00	0,59±0,06	0,46±0,04	0,04±0,00	65±04	112±13
SVM [17]	0,95±0,00	1,00±0,00	0,04±0,02	1,00±0,00	0,00±0,00	03±03	40±43
RF [17]	0,95±0,00	1,00±0,00	0,14±0,01	0,68±0,03	0,00±0,00	07±01	48±08
GBT [17]	0,95±0,00	1,00±0,00	0,07±0,00	1,00±0,00	0,00±0,00	03±00	34±00
RF [16]	0,95±0,00	1,00±0,00	0,06±0,01	0,74±0,08	0,00±0,00	02±01	30±14
RGF [22]	0,49±0,11	0,47±0,12	0,73±0,05	0,08±0,02	0,53±0,12	224±57	305±63

Considering the stochastic nature of the proposed algorithm as well as the related work, each method is run 10 times. For each run, the accuracy (ACC), the recalls of both classes (Rec$_0$ and Rec$_1$), the precision (Prec, defined as FP/(TP + FP)), the false positive rate (FPR, defined as FP/(TN + FP)) are reported. Additionally, the lead-time with which the failure is predicted is defined as the number of observations (days) from the last observation used to predict the failed disk to the observation when it really fails (see Figure 6). In contrast, if all the windows are predicted as 0 up to the last available one, the disk is classified as a correct disk. The average lead time of the failures for all the failed drives (LT) and this same metric restricted to the failed drives that are correctly classified (RLT) are calculated. Note that the lead time for incorrectly classified disks is 0.

A. Comparison with The State-of-The-Art

The objective of the first analysis is to compare the proposed method, DRL-Rec, with other related works and analyze whether it is an effective solution to the problem of hard drive failure prediction.

The obtained results for the DRL-Rec and the related work are presented in Table II. For each method, the average and the standard deviation of the measures described above for all the runs are presented. In addition, in Figure 7, a scatter plot is displayed where, for all the runs of each method, the recall of the class 0 and the recall of the class 1 are depicted (the further to the top-right, the higher the recalls).

As Table II shows, the three methods that are proposed in [17] and the RF model proposed in [16] obtain the highest accuracy, an average of 95%. These methods correctly classify all the correct disks as the value of 1 for the recall of this class indicates. However, these four methods are hardly able to correctly classify any failed disks, obtaining a very low recall for class 1. That is problematic because, even if no false positives are reported, the majority of the failed disks will be missed, resulting in a high cost.

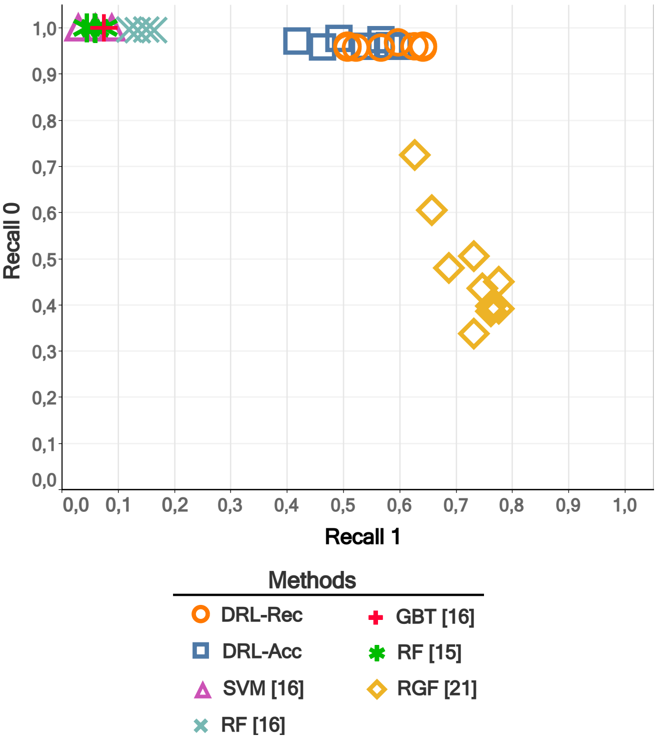


Fig. 7. Scatter plot with the recall of the class 0 and recall of the class 1 for all the runs of each method. The further to the top-left, the higher the recalls. Each method has a different shape and color.

On the contrary, the RGF model proposed in [22] obtains the highest values for the recall of the class 1. However, as can be observed in the table, although an average of more than 70% of the failed disks is detected, the accuracy is very low due to the high FPR obtained. Therefore, this model detects most of the failed disks at the cost of incorrectly predicting more than half of the correct ones. This would also mean a high economic cost because many disks would be replaced by new ones even though they did not show any signs of imminent failure.

Finally, the DRL-Rec method obtains almost the same accuracy and recall values for the class 0 as the first 4 methods and it is able to identify nearly 60% of the failed disks (significantly more than the first 4 methods). Moreover, when compared with RGF, although the proposed method does not reach the recall values for the class 1 of this method, it does not incur in a high FPR and it maintains a high recall value for the majority class (significantly higher than the RGF model). In conclusion, DRL-Rec achieves the most balanced results. This is confirmed by Figure 7, which shows that for the DRL-Rec, the points are more to the top-right corner than for the rest of the methods.

In addition, according to the lead time of the failure predictions, DRL-Rec manages to detect signs of malfunction that indicate future failure an average of 66 days before it occurs for all the failed disks and an average of about 112 days for the correctly predicted failed disks. This provides the user enough

TABLE III
THE OBTAINED RESULTS BY DRL-REC AND DRL-ACC WHEN APPLIED TO DATASET 2 AND 3 WITH IMBALANCE RATES OF 3% AND 1% RESPECTIVELY. EACH METHOD IS RUN 10 TIMES. FOR EACH METHOD, THE AVERAGE AND THE STANDARD DEVIATION OF THE ACCURACY, THE RECALLS, THE PRECISION, THE FALSE POSITIVE RATE, THE AVERAGE LEAD TIME OF THE FAILURES AND THE AVERAGE LEAD TIME OF THE FAILURES RESTRICTED TO CORRECTLY PREDICTED HARD DRIVES FOR ALL RUNS ARE PRESENTED.

Dataset 1 **5 %**	**ACC**	**Rec$_0$**	**Rec$_1$**	**Prec.**	**FPR**	**LT**	**RLT**
DRL-Rec	0,94±0,01	0,96±0,00	0,59±0,06	0,46±0,04	0,04±0,00	65±04	112±13
DRL-Acc	0,94±0,01	0,97±0,01	0,52±0,07	0,47±0,06	0,03±0,01	52±11	99±11

Dataset 2 **3 %**	**ACC**	**Rec$_0$**	**Rec$_1$**	**Prec.**	**FPR**	**LT**	**RLT**
DRL-Rec	0,95±0,01	0,96±0,01	0,57±0,03	0,33±0,04	0,04±0,01	31±11	55±18
DRL-Acc	0,96±0,01	0,98±0,01	0,30±0,12	0,32±0,08	0,02±0,01	10±05	36±15

Dataset 3 **1%**	**ACC**	**Rec$_0$**	**Rec$_1$**	**Prec.**	**FPR**	**LT**	**RLT**
DRL-Rec	0,96±0,01	0,96±0,01	0,60±0,08	0,14±0,02	0,04±0,01	61±01	111±15
DRL-Acc	0,97±0,01	0,98±0,01	0,40±0,11	0,17±0,07	0,02±0,01	45±09	117±27

time to take preventive actions. In comparison with the other methods, it obtains higher lead time values than the first 4 methods and lower values than the RGF model. One reason why the RGF model obtains higher lead time values may be that this method is strongly biased towards predicting class 1, and hence it detects the windows of this class the earliest, even if it missclassifies many correct disks in this process.

B. Comparison of the Proposed Methods in High Imbalance Rates

The results obtained when applying DRL-Rec and DRL-Acc to Datasets 1, 2 and 3 are shown in Table III. As for the previous table, the average and the standard deviation of the measures described above for all the runs are presented.

It can be observed that DRL-Acc obtains the same accuracy as DRL-Rec for the Dataset 1 and a slightly higher accuracy for the rest of the datasets. This makes sense because the objective of DRL-Acc is to maximize the accuracy. Due to the imbalance of the class frequencies, it achieves these values by correctly predicting almost all the correct disks, obtaining somewhat higher values for the recall of the majority class than the DRL-Rec.

In contrast, DRL-Rec obtains a notably higher recall for the minority class than DRL-Acc, but classifies the class 0 almost as well as it. Moreover, for DRL-Acc, the standard deviation of the recall of class 1 and of the precision is higher than for DRL-Rec, especially in Dataset 2. This shows that DRL-Rec is the most robust method (it has less variability) for correctly predicting failed disks. Consequently, it can be determined that by using the proposed function (1), the importance of both classes is balanced and, even when the accuracy slightly decreases, more failed disks are predicted with almost no loss in the predictions of the majority class. This becomes more evident when the original imbalance rate is increased from 5% to 3% or 1%. Finally, DRL-Rec manages to detect signs of malfunction that indicate future failure more days in advance than DRL-Acc (except for the RLT in the Dataset 1 which is very similar for both methods).

These results validate the main adaptation proposed to address the challenge of the imbalanced scenario.

V. CONCLUSIONS AND FUTURE WORK

The objective of this paper is to develop a method for addressing the hard drive failure prediction problem. The method is capable of effectively detecting when the hard drive begins to malfunction, allowing the user to take preventive actions. This goal is successfully achieved by designing a double-round learning methodology. It allows training a classifier which can automatically identify the time series window in which the hard drive starts to show signs of malfunction instead of setting a predefined lead time of malfunction, as in previous state-of-the-art proposals.

In addition, to deal with the imbalanced nature of the problem, a classifier for MTS that is designed to maximize a minimum recall based function is proposed rather than the accuracy-based functions, which are typically used. Specifically, a deep neural network classifier for MTS is used. Since traditional gradient-based learning methods for learning neural network classifiers require a differentiable loss function to be applied and the minimum recall is not differentiable, a framework comprised of two consecutive genetic algorithms (GAs) is proposed that heuristically optimizes the network architecture and its weights respectively for maximizing the minimum recall of the classes. Due to this, equal importance is given to both classes, failed and correct disks, and classifiers are obtained that tend to equilibrate the recall value of both classes.

The experimental results obtained when applying the proposed method to the open data provided by the Backblaze company are compared with the results obtained with other state-of-the-art methods, showing that the proposed method obtains the most balanced results in terms of the recall of both classes. Moreover, in order to analyze the performance of the proposed method when the imbalance rate is increased, two additional datasets with higher imbalance rates than the original rate are generated. The results of this experimentation highlight the effectiveness of the proposed classifier to deal with high imbalance rates.

It should be noted that, although it is presented specifically for the hard drive failure prediction problem, the developed learning methodology is applicable to other MTS streaming classification problems. Consequently, future work could focus on applying it to similar problems that are also highly unbalanced. In addition, another possible research line could be to try to improve the training process of the classifier investigating how to approximate the proposed minimum recall based function with differentiable functions. This would allow the application of classical techniques such as gradient methods (instead of applying GAs) for learning the weights.

Finally, if the classifier is operating for a long period of time, it could be possible that the operating conditions could change and new types of hard drives could be added to the data center. As a result, the effectiveness of the classifier could decrease [39]. An interesting future research topic is to propose methods to discover when classifier degrades its performance sufficiently to require intervention.

Acknowledgments

The authors wish to express their thanks to the Basque Government for their financial support of this research through the Elkartek program under the DIGITAL project (Grant agreement No. KK-2019/00095) and under the 3KIA project (Grant agreement No. KK-2020/00049). Any opinions, findings and conclusions expressed in this article are those of the authors and do not necessarily reflect the views of funding agencies. Jose A. Lozano is partially supported by the Basque Government through the BERC 2018-2021 program and IT1244-19 and by the Spanish Ministry of Science, Innovation and Universities: BCAM Severo Ochoa accreditation SEV-2017-0718 and PID2019-104966GB-I00.

References

[1] PonemonInstitute, "Cost of data center outages: Data center performance benchmark series," Emerson Network Power, Ponemon Institute LLC., Tech. Rep., 2016. [Online]. Available: https://www.ponemon.org/research/ponemon-library/security/2016-cost-of-data-center-outages.html

[2] B. Allen, "Monitoring hard disks with smart," *Linux Journal*, no. 117, pp. 74–77, 2004.

[3] J. F. Murray, G. F. Hughes, and K. Kreutz-Delgado, "Machine learning methods for predicting failures in hard drives: A multiple-instance application," *Journal of Machine Learning Research*, vol. 6, pp. 783–816, 2005.

[4] M. Garcia, V. Ivanov, A. Kozar, S. Litvinov, A. Reznik, V. Romanov, and G. Succi, "Review of techniques for predicting hard drive failure with smart attributes," *International Journal of Machine Intelligence and Sensory Signal Processing*, vol. 2, no. 2, pp. 159–172, 2018.

[5] J. Yu, "Hard disk drive failure prediction challenges in machine learning for multi-variate time series," in *Proceedings of the 2019 3rd International Conference on Advances in Image Processing*, 2019, pp. 144–148.

[6] D. Jauk, D. Yang, and M. Schulz, "Predicting faults in high performance computing systems: An in-depth survey of the state-of-the-practice," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2019, pp. 1–13.

[7] L. Zhu, X. Li, W. Tong, C. Zhang, and B. Li, "A survey on default prediction of cloud storage service," in *2019 IEEE International Conference on Computation, Communication and Engineering (ICCCE)*. IEEE, 2019, pp. 13–16.

[8] F. D. dos Santos Lima, F. L. F. Pereira, I. C. Chaves, J. P. P. Gomes, and J. de Castro Machado, "Evaluation of recurrent neural networks for hard disk drives failure prediction," in *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*. IEEE, 2018, pp. 85–90.

[9] F. D. S. Lima, F. L. F. Pereira, L. G. Leite, J. P. P. Gomes, and J. C. Machado, "Remaining useful life estimation of hard disk drives based on deep neural networks," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–7.

[10] S. Basak, S. Sengupta, and A. Dubey, "Mechanisms for integrated feature normalization and remaining useful life estimation using lstms applied to hard-disks," in *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE, 2019, pp. 208–216.

[11] M. Züfle, C. Krupitzer, F. Erhard, J. Grohmann, and S. Kounev, "To fail or not to fail: Predicting hard disk drive failure time windows," in *Measurement, Modelling and Evaluation of Computing Systems - 20th International GI/ITG Conference, MMB 2020, Saarbrücken, Germany, March 16-18, 2020, Proceedings*, 2020, pp. 19–36.

[12] C. Xu, G. Wang, X. Liu, D. Guo, and T.-Y. Liu, "Health status assessment and failure prediction for hard drives with recurrent neural networks," *IEEE Transactions on Computers*, vol. 65, no. 11, pp. 3502–3508, 2016.

[13] D. Liu, B. Wang, P. Li, R. J. Stones, T. G. Marbach, G. Wang, X. Liu, and Z. Li, "Predicting hard drive failures for cloud storage systems," in *International Conference on Algorithms and Architectures for Parallel Processing*. Springer, 2019, pp. 373–388.

[14] L. P. Queiroz, F. C. M. Rodrigues, J. P. P. Gomes, F. T. Brito, I. C. Chaves, M. R. P. Paula, M. R. Salvador, and J. C. Machado, "A fault detection method for hard disk drives based on mixture of gaussians and nonparametric statistics," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 542–550, 2016.

[15] J. Li, R. J. Stones, G. Wang, X. Liu, Z. Li, and M. Xu, "Hard drive failure prediction using decision trees," *Reliability Engineering & System Safety*, vol. 164, pp. 55–65, 2017.

[16] M. Thandapani and V. Krishnan, "A stable model to predict the hard disk failure," Ph.D. dissertation, Dublin, National College of Ireland, 2017.

[17] N. Aussel, S. Jaulin, G. Gandon, Y. Petetin, E. Fazli, and S. Chabridon, "Predictive models of hard drive failures based on operational data," in *16th IEEE International Conference on Machine Learning and Applications, ICMLA 2017, Cancun, Mexico, December 18-21, 2017*, 2017, pp. 619–625.

[18] W. Yang, D. Hu, Y. Liu, S. Wang, and T. Jiang, "Hard drive failure prediction using big data," in *34th IEEE Symposium on Reliable Distributed Systems Workshop, SRDS 2015 Workshop, Montreal, QC, Canada, September 28 - October 1, 2015*, 2015, pp. 13–18.

[19] C. A. R. C., J. Pâris, R. Vilalta, A. M. K. Cheng, and D. D. E. Long, "Disk failure prediction in heterogeneous environments," in *International Symposium on Performance Evaluation of Computer and Telecommunication Systems, SPECTS 2017, Seattle, WA, USA, July 9-12, 2017*, 2017, pp. 1–7.

[20] J. Xiao, Z. Xiong, S. Wu, Y. Yi, H. Jin, and K. Hu, "Disk failure prediction in data centers via online learning," in *Proceedings of the 47th International Conference on Parallel Processing*, 2018, pp. 1–10.

[21] W. Pei, B. Xue, L. Shang, and M. Zhang, "Developing interval-based cost-sensitive classifiers by genetic programming for binary high-dimensional unbalanced classification [research frontier]," *IEEE Computational Intelligence Magazine*, vol. 16, no. 1, pp. 84–98, 2021.

[22] M. M. Botezatu, I. Giurgiu, J. Bogojeska, and D. Wiesmann, "Predicting disk replacement towards reliable data centers," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 39–48.

[23] Z. Gong and H. Chen, "Model-based oversampling for imbalanced sequence classification," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 2016, pp. 1009–1018.

[24] G. Liang and C. Zhang, "A comparative study of sampling methods and algorithms for imbalanced time series classification," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2012, pp. 637–648.

[25] S. Roychoudhury, M. Ghalwash, and Z. Obradovic, "Cost sensitive time-series classification," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2017, pp. 495–511.

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

[27] K. C. Tan, L. Feng, and M. Jiang, "Evolutionary transfer optimization-a new frontier in evolutionary computation research," *IEEE Computational Intelligence Magazine*, vol. 16, no. 1, pp. 22–33, 2021.

[28] Y. Sun, B. Xue, M. Zhang, G. G. Yen, and J. Lv, "Automatically designing cnn architectures using the genetic algorithm for image classification," *IEEE Transactions on Cybernetics*, 2020.

[29] F. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[30] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks," *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 5, pp. 361–378, 2017.

[31] M. Canizo, I. Triguero, A. Conde, and E. Onieva, "Multi-head cnn–rnn for multi-time series anomaly detection: An industrial case study," *Neurocomputing*, vol. 363, pp. 246–260, 2019.

[32] C. Yin, S. Zhang, J. Wang, and N. N. Xiong, "Anomaly detection based on convolutional recurrent autoencoder for iot time series," *IEEE*

Transactions on Systems, Man, and Cybernetics: Systems, pp. 1–11, 2020.

[33] G. W. Brier, "Verification of forecasts expressed in terms of probability," *Monthly Weather Review*, vol. 78, pp. 1–3, 1950.

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

[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[36] A. Umbarkar and P. Sheth, "Crossover operators in genetic algorithms: a review." *ICTACT journal on soft computing*, vol. 6, no. 1, 2015.

[37] A. R. Mashhadi, W. Cade, and S. Behdad, "Moving towards real-time data-driven quality monitoring: a case study of hard disk drives," *Procedia Manufacturing*, vol. 26, pp. 1107–1115, 2018.

[38] X. Sun, K. Chakrabarty, R. Huang, Y. Chen, B. Zhao, H. Cao, Y. Han, X. Liang, and L. Jiang, "System-level hardware failure prediction using deep learning," in *Proceedings of the 56th Annual Design Automation Conference 2019, DAC 2019, Las Vegas, NV, USA, June 02-06, 2019*, 2019, p. 20.

[39] J. Riihijarvi and P. Mahonen, "Machine learning for performance prediction in mobile cellular networks," *IEEE Computational Intelligence Magazine*, vol. 13, no. 1, pp. 51–60, 2018.