Challenges and Benefits for Detecting Soon-to-Fail Drives in Industry 4.0 *

Marko Krstic, Nicolas Nicolaou, and Efstathios Stavrakis

Algolysis Ltd, Cyprus
https://www.algolysis.com/
marko@algolysis.com,nicolas@algolysis.com,
stathis@algolysis.com

Abstract. Data storage and processing is an integral component of Industry 4.0 applications. Failures of storage devices may lead to system corruptions and malfunctioning of physical infrastructures. In this work we explore a methodology to monitor the storage devices that are used in an Industry 4.0 environment, and investigate mechanisms for early detection of failing devices. In particular, we describe the usage of Machine Learning models over data that describe the physical condition of the storage devices to determine whether a device is ill-functioning. We support our proposed models with experimental results over a large dataset, and we provide an analysis on the performance of our algorithms.

Keywords: Proactive monitoring · Storage devices · Machine learning · Industry 4.0.

1 Introduction

Computing and automation in product manufacturing have been key drivers of the 3rd industrial revolution. Building on that foundation, the 4th industrial revolution aims at devising and deploying smarter autonomous systems that are powered by data, analytics and Artificial Intelligence.

These cyber-physical systems are no longer considered in isolation; they are seen as part of a distributed, sometimes decentralized, infrastructure for product manufacturing. This design lifts the barriers imposed by physical proximity and enables generation of knowledge through continuous monitoring, data acquisition and processing from thousands of devices, even in disparate locations. The flow of information from these cyber-physical systems into data analytics and Machine Learning pipelines are essential for designing adaptive system controls that can respond to the dynamic requirements of production processes, improve operational efficiency, and increase the level of automation in industrial and critical infrastructures [1].

* This work was partially supported by the EU H2020 Innovation Associate grant PREFAIL (957149) and by the Cyprus Research and Innovation Foundation under the grant agreement POST-DOC/0916/0090.

Since data are at the core of such systems, applications in Industry 4.0 rely heavily on reliable, accessible and dependable data storage. Failures in storage devices of industrial infrastructures may prove catastrophic: computing devices will fail to collect data and perform the necessary computations, thus disrupting the infrastructure control loop. Although the main focus of research in Technology ecosystem of Industry 4.0 are usually on communication technologies (e.g. Zigbee, Bluetooth 4.0, Sigfox), IoT devices and computing approaches (e.g. Cloud and Fog computing) [2, 3], it is crucial to develop system for proactive monitoring, detection and mitigation of storage device failures - in order to avoid extensive, and costly disruptions.

In this work we examine the proactive detection of failing storage devices (or drives), often referred to as *soon-to-fail* drives, by utilizing hardware metrics (i.e., Self-Monitoring, Analysis and Reporting Technology (S.M.A.R.T) measurements [4]) and Machine Learning (ML) algorithms. S.M.A.R.T attributes [4] were introduced by storage vendors as a means to examine the condition and usage of drive hardware. These metrics are typically a snapshot of the current status of various attributes of a storage device (e.g., Temperature) or cumulative characteristics (e.g., Power On Hours). Although a snapshot of those metrics may not be sufficient to provide enough insight about the state of a device, sequence of historical data may reveal patterns that indicate issues on a storage device. Such patterns can be learned by utilizing Machine Learning models trained with appropriate data.

These metrics can be hard to obtain and oftentimes due to proprietary technologies used on embedded devices extracting useful device attributes may not be possible. Furthermore, in the context of Industry 4.0, datasets for training ML models are hardly available. To circumvent those issues and make progress, it is feasible to design models and methodologies for designing such data storage device monitoring and failure prediction systems by drawing from ICT sector. We utilize large scale datasets of metrics obtained from storage devices in datacenter infrastructures that have become available (i.e. BackBlaze[5]) and are equivalent in breadth and depth to the data a modern manufacturing facility may generate in the field. We present the methodology of how to pre-process that data and train a number of ML models in Tensorflow framework [6] to proactively detect soon-to-fail drives.

Using the predictive power of these algorithms we aim to the timely detection of soon-to-fail storage devices in order to: (i) improve the reliability of existing storage systems by proactively migrating the data to healthy devices before outages, and (ii) prevent performance degradation by ill-functioning storage components.

Background. The early approaches was trying to predict hard drive failures by setting thresholds for different S.M.A.R.T attributes, however threshold conservative selection process led to detection of only 3% - 10% of failed drives [7].

The usage of machine learning algorithms for prediction gave significantly better results even when simple techniques are used. For example, supervised naive Bayes classifier trained on small dataset of 1,936 drives (with only 9 failed

ones) achieved prediction accuracy of 33% with 0.67% false alarm rate [8]. Furthermore, in the early phase of the disk failure prediction research it is shown that more complex algorithms like Support Vector Machine (SVM), can provide predictions without false alarms, while detecting 50.6% of failed drives on small dataset with 369 drives [7]. However, the more representative results emerged with increased availability of large datasets [5, 9, 10] and interests of large companies (like Alibaba) which recognized business values of these prediction systems.

With the increased size of datasets and heterogeneity of considered drives, it becomes possible to develop prediction systems that can be used in the production, however the design of such system is not a trivial task. For example, decision tree model trained on one year Backblaze data for all hard drive models available in it can accurately recognize 52% of failed drives, but that comes with a price of 60% false alarms [11]. Sequential modeling of disk failure process by using Deep Learning models can further improve these results. By using this approach, according to [9], it is possible to achieve not only better prediction performance for set of disks in Data Centers on which combination of Convolutional Neural Network (CNN) and Long Short-Term Memory Network (LSTM) is trained, but also the model has much better generalization capabilities on monitoring new Data Centers (not used in training). Furthermore it is shown that additional information about disk location and system performance (i.e. number of normal/temp files written successfully), could significantly improve performance as CNN LSTM model trained on Wayne State University dataset can achieve F1 measure score of 58% when using only S.M.A.R.T attributes, whereas in case when all available information are used this value go to 95%. Thus, definition of performance metrics that adequately describes functioning of Industry 4.0 storage systems can be quite beneficial for application of proactive monitoring in these environments. On the other hand, some approaches combine sequential modeling of drive failure process with static information about drives (like manufacturer and model). The small CNN model with only 6 layers (2 convolutional, 2 maxpooling, and 2 dense) can achieve F1 measure score of 71% (with false alarms in 25% of cases) if information about drives manufacturer is added into first dense layer and S.M.A.R.T attributes normalization uses historical ratio of failed and healthy drives for each vendor [12]. However, these information will not always be available and/or behavior of some specific drives used in Industry 4.0 could deviate significantly from regular data centre drives thus using mean values for vendor could introduce significant bias.

In cooperation with academic community, Alibaba as industry partner organized PAKDD (Pacific-Asia Conference on Knowledge Discovery and Data Mining) 2020 AI Ops Competition in order to tackle problem of large-scale disk failure prediction [13]. The general methodology adopted by the most of competitors was comprised of data preprocessing, training sample generation, feature engineering, and modeling [14]. The most commonly used methods for preprocessing was simple ones like dropping the samples with missing data directly, filling them with some constant value (i.e. zero), or interpolating missing data by forward filling methods [15]. The main goal of training sample generation

was to reduce class imbalance for what most of participants adopted, either, upsampling methods for generation of synthetic samples (i.e. SMOTE [16]) or downsampling methods to randomly select a subset of samples [17] that describe healthy disk functioning. In the feature engineering phase, participants, predominately used sliding-windows with various window lengths, different statistical measures such as difference, mean, variance, and exponentially weighted moving average values to create features from S.M.A.R.T attributes, whereas some of them also apply feature selection procedure to remove those features that are not correlated with disk health status. As from the generally used methodology in AI Ops Competition, that includes feature engineering phase, it can be easily concluded, the focus in modelling part was on shallow Machine Learning algorithms. The disk failure prediction was commonly considered as binary classification problem and tree-based ensemble models are trained by using either LightGBM [18] or Xgboost [19] framework. The selection of those frameworks is justified by their short execution time as well as low memory requirements what are very important aspects for applications like this when large amount of data is available.

In addition to mainstream approaches, a few novel methods in each phase of identified general methodology of PAKDD2020 Alibaba AI Ops Competition emerged [14]. For example, it is shown that usage of the cubic spline interpolation method to deal with missing data problem can increase F1-score by more than 3% [20]. The same authors proposed also application of Generative Adversarial Network (GAN) for training sample generation that showed as promising new direction to tackle problem of class imbalance [20]. In the feature engineering phase, a couple of teams proposed feature construction methods based on analysis of the distance of failure occurrences, distributions of disk lifetime, and data missing ratio, which individually but also in the combination led to the improvement of failure prediction [14]. At the end, possible ways to improve performance in modeling phase is to consider failure prediction as multi-class classification or regression problem [21].

Contributions. The main goal of this work is to demonstrate the possibilities of utilizing ML algorithms for monitoring and detecting soon-to-fail drives in an Industry 4.0 deployment. To capture a well rounded and useful outcome, we followed specific steps starting from the identification of challenges all the way to the experimental analysis of the used models. In particular, our contributions are the following:

1. We first identify and present the challenges to overcome for identifying soon to fail drives in the wild.
2. Provided the challenges we then present a methodology for data preparation, and propose ML approaches that may be suitable for this problem. More specifically, we consider four different ML models: LSTM and CNN-LSTM which have been used in previous attempts as well, and the ResNET and a variant of a conventional CNN model which are proposed firstly in this work for this application.

3. Due to the nature of the problem and the dataset produced by S.M.A.R.T metrics, we suggest a methodology for properly training the proposed models.
4. Finally we implement and tested our models and obtained experimental results and analytical outcomes for each and every method proposed.

Our results shed some light in the potential of using S.M.A.R.T metrics alone with powerfull ML models to predict any potential drive failures.

Paper Structure. The paper is organized in following way. Section 2 presents the challenges with which systems for detection of soon-to-fail drives are facing. Sections 3 and Section 4 describe data preprocessing procedure and prediction models suitable for application in proactive monitoring of storage devices in Industry 4.0, respectively. Training process of ML models is explained in Section 5. In Section 6 experimental results are presented, whereas finally, Section 7 concludes the work.

2 Challenges

Achieving high level of proactive prediction of drive failures is not a trivial problem due to many data-related challenges present in S.M.A.R.T measurements such as high level of noise, extremely class imbalanced distribution, concept drift phenomena, large number and heterogeneity of hard drives included. In more detail the main challenges to accurate drive failure prediction are the following:

High level of noise. The noise in datasets for disk failure prediction stems from the non-standardized values of S.M.A.R.T attributes as well as from the non-reliable and non-appropriate labeling procedure.

Different hard drive vendors may use specific S.M.A.R.T attributes for different purposes [22]. This introduces noise and makes it much harder for prediction models to learn how healthy drives should function and what are the characteristics that indicate soon-to-fail drives. Such inconsistencies, pushed many researchers to use sampling strategies selecting only specific types of disks for training their prediction models, and thus avoiding the risk for introducing additional noise through inclusion of many different hard drive models/vendors in training dataset.

On the other side, labeling procedure is equally important both in the context of confirming that reason why hard drive does not send information about S.M.A.R.T attributes is its failure, as well as that the number of days before failure for which disk is marked as soon-to-fail is chosen in appropriate way so from S.M.A.R.T attributes it is possible to recognize failing conditions. S.M.A.R.T measurements are not received also in situations when there are some communication problems or if disk itself does not have power supply (either when itself or device using it is turned off, or in case of power outage).

Extremely class imbalanced distributions. Due to low hard drive failure rate, the number of samples in which S.M.A.R.T attributes describes healthy conditions is much bigger than the number of samples which can be used to describe soon-to-fail and failed drives. Therefore, the datasets are characterized by extremely class imbalanced distributions. For example, in the last three years annualized hard drive failure rate in Backblaze dataset (shown on Fig. 1) has values between 1 and 2 percent [5]. Furthermore, if we take into account the fact that even for failed disks most of the samples corresponds to normal disk functioning conditions and that only during a short period of time before the failure S.M.A.R.T attributes indicate failures, it is clearly that without dealing with class imbalance, negative samples will have much bigger impact on the training process. Sampling techniques (like Synthetic Minority Oversampling Technique – SMOTE [16] that increase number of positive samples), cost sensitive learning [23] (that incorporate different weights for positive and negative samples in learning process), special loss functions (with different weights incorporated in them, like focal loss [24]) or anomaly detection approach [25] (that learns how to recognize healthy conditions and detect anomalies) can be used to improve prediction for positive samples (soon-to-fail drives).

Fig. 1. Annualized hard drive failure rate for Backblaze dataset [5]

Concept drift. Datasets are not only characterized by extremely class imbalanced distributions, but also by concept drift phenomena. In particular, statistical patterns vary in time due to the addition of new and the removal of old or failed drives from the monitored drive pool. To deal with this problem, *change point* detection algorithms can be used to decide when there is a need for retraining or updating the prediction models [26].

Large number and heterogeneity of hard drives in a dataset. The problem of hard drive failure prediction can be also considered as Big Data problem due to the large number of hard drives in publicly available datasets, as well as in production systems of big infrastructure providers (like Alibaba) which monitors large Data Centers. For example Backblaze dataset for first quartal of 2021 contains S.M.A.R.T measurements for 175 443 drives from four data centers

on two continents [5], whereas Wayne State University (WSU) made available dataset that contains information about 380,000 hard disks over a period of two months across 64 sites [9]. Because of this characteristic of datasets, special attention should be paid that prediction models can be trained in reasonable time on computing resources available to the company that develop this type of product. This directly influences which prediction models can be used, the way in which training data is selected, as well as lead to batch-by-batch training process.

In addition to large number of drives, representative datasets (like Backblaze and WSU dataset) are also characterized by large heterogeneity. This further introduces more complexity in failure prediction problem as it is not unusual that different drives models have different operating characteristics.

3 Data processing

To the best of our knowledge there does not exist dataset on metrics on storage devices readily available in Industry 4.0. Therefore, for this study we utilize a set of datasets offered by Backblaze [22], a leading company in the field of data backup, containing S.M.A.R.T data obtained from a large number and heterogeneous drives, allowing us to capture a big subset of devices that may be used in Industry 4.0. Such data may align perfectly with recent strategies that move computing toward the "edge", forming the the so-called edge data centers in Smart factories [27].

Backblaze collects S.M.A.R.T reports from their devices daily. We focused on the data collected in the period between January 1st 2018, and December 31st 2019. This dataset consists of 142138 different drives.

Backblaze reports a drive as failed when it is removed from storage or replaced due to one of the following reasons:

1. The drive has stopped working: This means that it won't power up, doesn't respond to console commands or the RAID system alerts that the drive can't execute read or write operations.
2. The drive is about to fail: Empirical evidence has lead to the decision to remove a drive before it fails catastrophically.

In the rest of the paper we refer to a *failed* drive when this appears in the period we investigate and it is marked as failed by Backblaze before the end of the period; otherwise the drive is *healthy*.

3.1 Dataset Format

Each yearly dataset consists of 365 daily snapshots, which are stored as CSV (Comma Separated Values) files. Each file contains a header row, and a report from each monitored drive for each subsequent row. A drive report represents a daily snapshot of the respective drive. The columns of the dataset represent the following information:

- **Date**: The date of the file in yyyy-mm-dd format.
- **Serial Number**: The manufacturer-assigned serial number of the drive.
- **Model**: The manufacturer-assigned model number of the drive.
- **Capacity**: The drive capacity in bytes.
- **Failure**: Contains a "0" if the drive is healthy. Contains a "1" if this is the last day the drive was operational before failing.
- **Attributes**: 90 columns of S.M.A.R.T attributes and their normalized values, each associated with an identifier.

3.2 Feature Selection

The sequence of raw S.M.A.R.T attributes given in Table 1, are used as inputs to our prediction models. The identifiers of S.M.A.R.T attributes corresponds to those used in [9], with the main difference that we use the *raw value* of those attributes which can provide insights in disk functioning without vendor-based normalization. This type of normalization can prevent our ML system to recognize different drive models, what can be really important in the environments with heterogeneous hard drives. Instead, we apply min-max normalization, which transforms the S.M.A.R.T attributes values into the range $[0, 1]$, in order to improve convergence speed of neural networks [28]. For a smart attribute α, the value 0 corresponds to the minimum value of α in our data set, whereas value 1 corresponds to the the maximum value of α.

Table 1. List of S.M.A.R.T attributes used.

ID	Name	Description
1	Read Error Rate	Frequency of errors during read operations.
3	Spin-Up Time	Time required a spindle to spin up to operational speed.
4	Start/Stop Count	Raw value holds the actual number of spin-up/spin-down cycles.
5	Reallocated Sectors Count	The number of the unused spare sectors. When encountering a read/write/check error, a device remaps a bad sector to a "healthy" one taken from a special reserve pool.
9	Power-On Hours Count	The Raw value shows the actual powered-on time, usually in hours.
12	Power Cycle Count	The Raw value holds the actual number of power cycles.
194	Temperature	Temperature, monitored by a sensor somewhere inside the drive. Raw value typicaly holds the actual temperature (hexadecimal) in its rightmost two digits.

3.3 Test/Training/Validation Data Set Creation

We generate the testing dataset by selecting 30 healthy and 30 failed drives from each year at random. The remaining data is divided into training and validation dataset in proportion of 70% and 30%, respectively.

With the sequence length parameter it is possible to control how many previous days S.M.A.R.T measurements are added to the current to describe operational state of the drive. In this paper we adopted sequence length of 30 days as long enough to model process of disk failing.

Labeling determines whether a drive is about to fail "soon" or whether the drive remains healthy within a predefined time window, termed as *labeling window*. We followed the suggestions presented in [29], where authors analyzed the duration of prefail period for two specific hard drive models from two vendors and determined that 29 and 27 days, respectively, was a satisfying period. Thus, in our experiments we use a labeling window of 29 days and we label the records in our dataset in a binary fashion as follows:

- H: a drive record is labeled as *healthy* at a date D, if the drive is not marked as failed in our dataset in any day d s.t., $D < d \leq D + 29$ or we have reached the end of the dataset (whichever is first).
- F: a drive record is labeled as *soon-to-fail* (or failed) at date D, if the drive is marked as failed in the dataset in any day d s.t., $D < d \leq D + 29$.

Based on the label of the drive record, we also classify drives' *operating characteristics*, i.e. the conditions under which a drive operates and which can be measured by the S.M.A.R.T attribute values. More precisely, if a drive is labeled as *soon-to-fail* (F) then the values of S.M.A.R.T attributes describe soon-to-fail operating characteristics; otherwise, the S.M.A.R.T values describe healthy drive operating characteristics.

In order to deal with the class imbalance problem, healthy drives are randomly selected from our dataset in such a way that the ratio between failed and healthy ones, in both training and validation set, was 1:2. This ratio is selected in order to reduce class imbalance but also to preserve diversity of drives for training the ML models. However, as each drive can be operating for different time duration, resulting into different number of samples per drive, additional *undersampling and cost sensitive learning methods* will be examined to further reduce class imbalance (see Section 5.2).

4 Prediction Models

In this section we present the ML models we consider and we substantiate their applicability in Industry 4.0 applications.

4.1 LSTM and CNN LSTM Models

Industry 4.0 environments, often include rare and non-typical computing devices which may use non-conventional hard drives as storage mediums. To this end, in order to provide viable proactive solutions for detecting soon-to-fail drives in such environments, we should examine solutions that model drive failures through sequence of historical operating characteristics, an approach that showed good

generalization capabilities especially in the case of LSTM and CNN LSTM models [9]. LSTM model learns directly from S.M.A.R.T attributes which patterns corresponds to soon-to-fail conditions. On the other hand, CNN LSTM model uses CNN for features extraction and LSTM to support sequence classification. An illustration of the structure of LSTM and CNN LSTM models is shown in Fig 2.

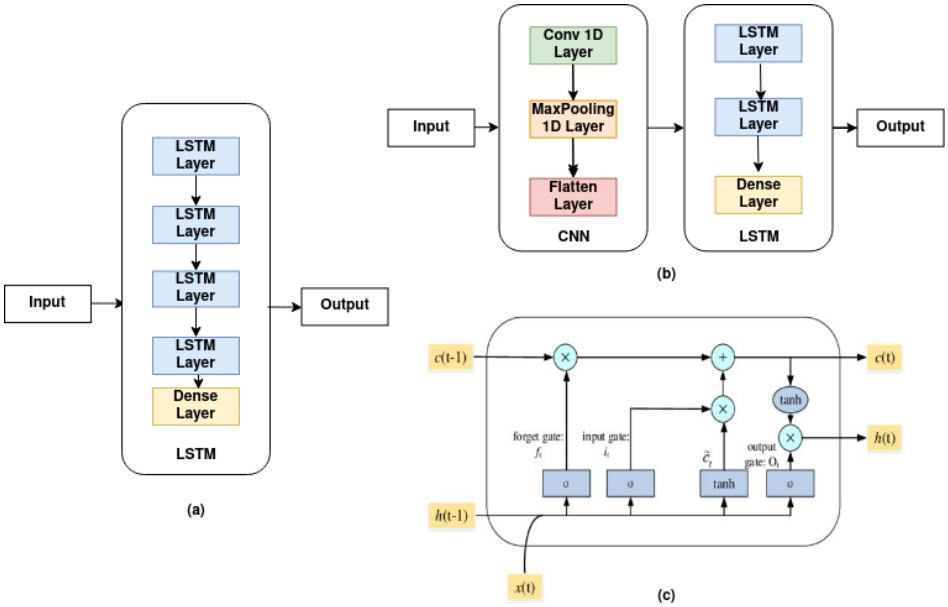

Fig. 2. a) LSTM model [9], (b) CNN LTSM model [9], and (c) LSTM Unit[30]

LSTM layers [31] are composed of LSTM units which can process data sequentially and save hidden state ($h(t)$) through time. The functioning of LSTM unit is described on Fig 2. With symbols $x(t)$ the input vector at timestep t, $h(t-1)$ the hidden state value in timestep $(t-1)$, $c(t)$ and $c(t-1)$ the cell state vectors in the current timestemp t and in the previous timestemp $(t-1)$, and o the Hadamard product. Initial values for cell state vector $c(0)$ and hidden state $h(0)$ are zeros, whereas the other time steps in sequence are computed according to Fig 2. LSTM unit contains input, forget and output gates which controls its behavior, and makes the propagation of unchanged gradients from previous time steps possible, in order to support deeper neural network architectures. On the top of the LSTM layers in LSTM model [9], fully-connected Dense layer [32] is added in order to support binary classification function (healthy or soon-to-fail drives), what is the main goal of the neural network as a whole.

On the other hand, CNN LSTM model [9] applies additional transformation before LSTM layers. Conv 1D layer [33] firstly calculate convolutions over time

dimension, then MaxPooling 1D [34] layer downsamples the input representation by taking the maximum value over spatial windows, whereas lastly Flatten layer [35] transforms its input vector to the shape [batch size, sequence length, number of features] that is expected by LSTM layers.

4.2 The ResNet Model

As in its essence modeling failures by sequence of historical operational characteristics translates disk failure prediction problem into the so-called Time Series Classification (TSC) problem [36], proactive failure detection systems could benefit from the recent advancement made in TSC area.

Until recently, Hierarchical VotE – Collective Of Transformation-based Ensembles (HIVE-COTE) [37] that combines predictions from 35 individual classifiers built on four different data representations, was recognized as the only way to achieve state-of-the-art performance in TSC problems, however due to its complexity it cannot be trained in reasonable time for large datasets. In these scenarios, CNNs emerged as great alternative because of their capability to learn time-invariant representations [38].

The ResNet architecture [36], with 3 residual blocks followed by a global average pooling layer [39] (that averages the time series across the time dimension), showed its clear dominance in wide spread of domains for both univariate and multivariate time-series [38]. The structures of ResNet model and residual block are shown in Fig 4.

The time-invariant representation of S.M.A.R.T attributes that ResNet (and generally CNNs) generate enable failure prediction systems to detect important patterns that are not time dependent - something really important as different hard drives can exhibit similar patterns but not at the same time interval before the failure.

4.3 The Simple CNN Model

The computational complexity of the prediction model is also an important factor to allow a system to be implemented Industry 4.0's low-powered computing devices. To further explore this aspect, we modified the simple CNN architecture [12] that combines sequential (S.M.A.R.T attributes) and static information (disk vendor), by removing the dense layer used to include vendor information. Information about the manufacturer could help the ML model to differentiate between the behavior of drives from different vendors. However, in order to be successful, each vendor must be represented by "sufficient" number of measurements in the dataset. As this prerequisite is not always fulfilled in Industry 4.0 we modified the ML model architecture. The structure of this model, named *Simple CNN* hereinafter, is shown in Fig 5. The main difference between Global MaxPooling 1D [40] and MaxPooling 1D Layer [34] in Simple CNN architecture, is that when downsample input representation first method finds the maximum value in the whole time axis - thus reducing the dimensionality of the layer output vector by one, whereas the latter considers maximum values in different

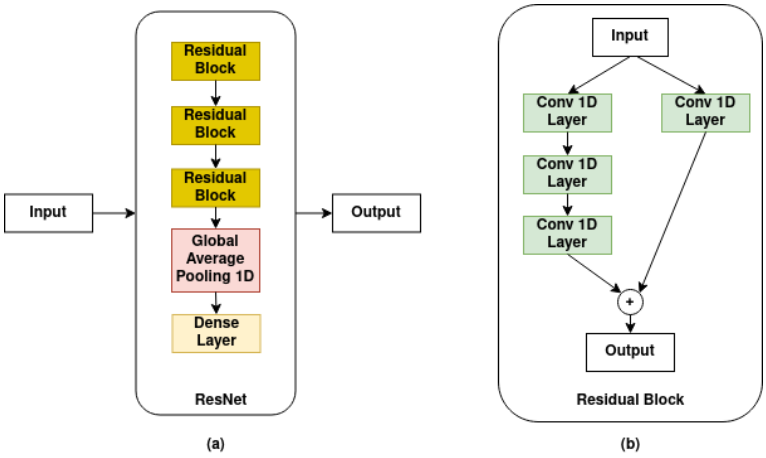

Fig. 3. a) Structure of ResNet model [38] b) Structure of residual block [1]

windows among time axis. This reduction in vector dimensionality is needed in order to prepare the inputs for the Dense layer.

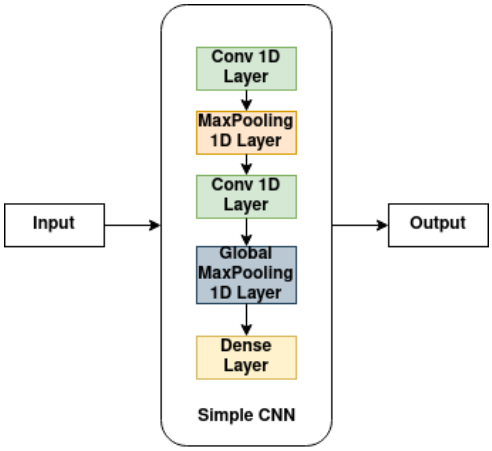

Fig. 4. Structure of Simple CNN model

5 Training Process

Adam, an adaptive learning rate optimization algorithm, with default initial learning rate parameter (lr=0.001) [41] was used in the process of training all prediction models described in Section 4.

Early stopping criteria, that will stop training if the optimization loss of the model on the validation dataset increases in predefined number of consecutive complete passes of training data (so called training epochs), is applied to avoid model overfitting. The training is done iteratively - so the training data is presented to neural network multiple times. We say that one training epoch is done when all the samples in a training dataset are presented to the neural network. The training dataset is divided into chunks of data (so called batches [42]) - when one batch is presented to the NN - we say that one iteration of training is done. After each epoch, model performance is evaluated on the whole validation dataset. If the loss increases or the value of some metrics (e.g., accuracy) declines for consecutive epochs, this means that further training will negatively affect the performance of the model on unseen datasets (although the performance on the training dataset could be improved). So in such a case we stop the training. Have we not apply this method, we increase the possibility of encountering the risk of achieving good performance for the training dataset, and poor generalization capability on unknown datasets.

Having in mind the challenges which proactive monitoring systems of storage devices are facing, special attention was paid to methods that deal with large amounts of data and with class imbalance.

5.1 Dealing with large amounts of data

As the size of the representative dataset is usually larger than the size of the RAM in a computing device (especially for low spec devices as those used in Industry 4.0), the training of ML models is typically done by splitting data into batches and iteratively process a single batch at a time. none of the previous works in the literature on proactive hard drive monitoring provided guidelines on how training batches should be constructed. Thus, in this work we examined two approaches where each batch is formed by:

1. data collected for a single drive
2. combination of data from failed and healthy drives

In the both cases, sampling without replacement is applied in order to select the drives which S.M.A.R.T measurements are used to construct the batches. The main difference is that in the first case, for each batch only one drive is selected, whereas in latter firstly one failed drive is chosen, followed by two healthy drives, and then their data is combined. This ratio of failed and healthy drives in latter case is used as it corresponds to the ratio initially applied to the downsample dataset.

5.2 Dealing with class imbalance

Once we split our dataset into batches, we then need to address the issue of class imbalance.

In the case where a batch is formed from data collected for a single drive, we apply both undersampling and cost-sensitive learning methods. For every healthy

disk in the training dataset, 64 data points are randomly selected, whereas for failed ones, all data points that correspond to soon-to-fail operating characteristics are included with addition of 34 randomly selected data points from times when they did function properly. The class imbalance factor is then calculated as the ratio between number of data samples where drives are marked as healthy and those marked as soon-to-fail, and in the process of learning impact of errors on healthy drives (negative) samples is reduced by this factor.

On the other hand, in the case when batch is formed from both healthy and failed drives data, we did not use undersampling to further reduce class imbalance in the training dataset. The reason for this decision was to support learning of differences between healthy and soon-to-fail operating characteristics in the same batch with as many examples. However in addition to standard cost-sensitive learning method, described in paragraph above, advanced method to fight with class imbalance in ML models with sequential inputs are also considered. Recent research showed that if cost/weights in training process is adjusted dynamically to class imbalance ratio in each batch, instead to global class imbalance ratio, ML models can achieve better classification performance [43], but until now this method is not examined in the context of the application in the disk failure prediction.

6 Experimental Results

To evaluate prediction capabilities and select the best neural network architecture for the proactive detection system for soon-to-fail drives, we conducted a series of experiments. As performance metrics - *accuracy*, *precision*, *recall* and *F1 score* was selected. Those are defined by the following equations:

$$accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{1}$$

$$precision = TP/(TP + FP) \tag{2}$$

$$recall = TP/(TP + FN) \tag{3}$$

$$F1 = 2 * (recall * precision)/(recall + precision) \tag{4}$$

where TP, TN, FP, and FN are number of true positives, true negatives, false positives and false negatives, respectively.

The prediction model weights that achieve the best F1 measure for validation dataset was saved as final output of the training for each model, and those are used for model performance comparison.

The experiments are divided into those that explore:

1. different ways to generate data in training batches, and
2. advanced class imbalance technique for time series.

6.1 Different ways to generate data in training batches

In the first series of experiments, the performances of LSTM, CNN LSTM, ResNet and Simple CNN neural network architectures are compared for cases

when each batch is formed from one disk data and when combine data from failed and healthy disks, respectively. The corresponding results are given in Table 2 and 3.

Table 2. Performance of models in case when batches are formed from one disk data.

Model	Precision	Recall	F1 measure	Accuracy
LSTM	0.45	1	0.62	0.55
CNN LSTM	0.45	1	0.62	0.55
ResNet	0.57	0.23	0.32	0.57
Simple CNN	0.56	0.8	0.66	0.6

Table 3. Performance of models in case when batches are formed from mixture of failed and non-failed drives.

Model	Precision	Recall	F1 measure	Accuracy
LSTM	0.57	0.88	0.69	0.61
CNN LSTM	0.57	0.88	0.69	0.61
ResNet	0.36	0.6	0.45	0.55
Simple CNN	0.32	0.66	0.43	0.52

According to Table 2., in the case when each batch is formed from one drive data, the best F1 measure can be achieved by using Simple CNN architecture. This model is capable of finding 80% of all failed drives, however this comes at price of 44% of false alarms. On the other side, the more complex architectures like LSTM and CNN LSTM are able to recognize all soon-to-fail drives in test dataset, but their precision is significantly reduced in comparison to Simple CNN architecture. There was no difference in the performances of those models so we can conclude that in our case representations learned by CNN was not more informative than S.M.A.R.T attributes as inputs to LSTM network. Lastly, the most complex architecture Resnet, performs poorly in the case when data from just one drive is included in each batch - even precision is slightly better than for Simple CNN architecture, it is capable to detect only 23% of failed drives in test dataset.

If we compare results from Table 2. and 3., we can easily observe that in the case when batches are formed as mixture of failed and non-failed drives the performance of more complex models are improving, whereas getting worse for Simple CNN model. The best performance under this conditions is achieved by LSTM and CNN LSTM model. Those are capable to detect 88% of drives which are in soon-to-fail conditions, while generating false alarms in 43% of cases. Although the F1-measure achieved by ResNet architecture was significantly increased, this model is still not one of the best ones. By using batches that contain information about both soon-to-fail and healthy disks it is possible

to significantly increase recall of ResNet model, but at the same time precision is deteriorated. On the other hand, as apparently this way of forming batches increases complexity of learning which values in the sequence of S.M.A.R.T attributes corresponds to soon-to-fail conditions, that is beneficial for more complex models, it results into performance degradation (both precision and recall) for simple models like Simple CNN.

6.2 Advanced class imbalance technique for time series

In the second series of experiments impact of introducing dynamic weights [43] into learning process to deal with class imbalance in the case when batch is formed as mixture of failed and non-failed drives is examined. The performances that different neural network architectures achieved under this conditions is given in Table 4.

Table 4. Performance of models in case when different class imbalance weights/costs are incorporated at batch level.

Model	Precision	Recall	F1 measure	Accuracy
LSTM	0.51	0.79	0.61	0.55
CNN LSTM	0.51	0.79	0.61	0.55
ResNet	0.34	0.54	0.37	0.51
Simple CNN	0.3	0.58	0.4	0.53

Unlike expected dynamic adjustment of weights of cost-sensitive learning to distribution in each batch of mixed failed and non-failed drives did not led to performance improvement, as it can be observed from Table 4. Even more, the performances for all neural network architectures was degraded. The possible reason for such behavior can be direct relation between global class imbalance in dataset and annual failure rates of drives, whereas this relation is not so clear in case when the weight/cost for positive and negative samples are adjusted in batch-by-batch basis. Thus global weights/cost beside helping models to deal with class imbalance could possibly incorporate information about global annual failure rates of drives used in training dataset.

6.3 Discussion

Bearing in mind, results of all experiments described above, LSTM model (shown in Fig. 2) should be trained in batch-by-batch manner with batches formed as mixture of healthy and failed drives in order to achieve best performance of system for detection of soon-to-fail drives. Although CNN LSTM model could achieve the same performance it introduces additional complexity that can be issue in some implementations of Industry 4.0. The model complexity can be really important in deployments where edge computing paradigm is used, and ML models are implemented locally on processing devices in Smart Factories.

For such cases, Simple CNN model trained with batches of data from one-by-one drive, can be an appropriate alternative.

7 Conclusion

As Industry 4.0 applications are heavily dependent on data acquisition and processing, storage devices in industrial infrastructure should fulfill strict requirements in terms of reliability, performance and monitoring. However, regardless to this fact application of proactive monitoring and detection systems, already considered by big infrastructure providers like Alibaba, are not yet thoroughly explored in Industry 4.0.

In this paper we demonstrated applicability of proactive detection systems for soon-to-fail drives in Industry 4.0 environments. The hard drive failure process is modeled as sequence of drive operational characteristics measured in the last 30 days by using ML models. Although specialized datasets for hard drives in Industry 4.0 still do not exist, large number and heterogeneity of drives in Backblaze dataset was adequate to emulate those environments in which similar characteristics are expected. The proposed detection system is designed in such a way to overcome challenges that arise from data characteristics, with the main focus on Big data and class imbalance implication. It is experimentally shown that LSTM neural network, trained in batch-by-batch manner, where each batch are formed from both information about healthy and failed drives, is capable to recognize 88% of drives that are in soon-to-fail conditions. However, this comes at price of 43% of false alarms, thus further research is needed to improve prediction precision. In order to achieve this future research will consider combining different ML models and ways of data representation (sequence of measurements vs only last measurement). The recent advances from Time Series Classification area in terms of ML models and methods to deal with class imbalance, that we examined in this paper, did not show valuable for application in domain of hard drive failure prediction.

References

1. Lu, Y.: Industry 4.0: A survey on technologies, applications and open research issues. Journal of industrial information integration **6**, 1–10 (2017)
2. Peraković, D., Periša, M., Sente, R.E.: Information and communication technologies within industry 4.0 concept. In: Design, Simulation, Manufacturing: The Innovation Exchange. pp. 127–134. Springer (2018)
3. Peraković, D., Periša, M., Zorić, P.: Challenges and issues of ict in industry 4.0. In: Design, simulation, manufacturing: The innovation exchange. pp. 259–269. Springer (2019)
4. KM, S.S., et al.: Self monitoring analysis and reporting technology (smart) copyback. In: International Conference on Information Processing. pp. 463–469. Springer (2011)
5. Backblaze dataset. https://www.backblaze.com/b2/hard-drive-test-data.html, last accessed 2021/07/13

6. Tensorflow framework. https://www.tensorflow.org/, last accessed 2021/08/01
7. Murray, J.F., Hughes, G.F., Kreutz-Delgado, K., Schuurmans, D.: Machine learning methods for predicting failures in hard drives: A multiple-instance application. Journal of Machine Learning Research **6**(5) (2005)
8. Hamerly, G., Elkan, C., et al.: Bayesian approaches to failure prediction for disk drives. In: ICML. vol. 1, pp. 202–209. Citeseer (2001)
9. Lu, S., Luo, B., Patel, T., Yao, Y., Tiwari, D., Shi, W.: Making disk failure predictions smarter! In: 18th {USENIX} Conference on File and Storage Technologies ({FAST} 20). pp. 151–167 (2020)
10. Alibaba pakdd2020 dataset. https://github.com/alibaba-edu/dcbrain/tree/master/diskdata, last accessed 2021/07/13
11. Rincón, C.A., Pâris, J.F., Vilalta, R., Cheng, A.M., Long, D.D.: Disk failure prediction in heterogeneous environments. In: 2017 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS). pp. 1–7. IEEE (2017)
12. Sun, X., Chakrabarty, K., Huang, R., Chen, Y., Zhao, B., Cao, H., Han, Y., Liang, X., Jiang, L.: System-level hardware failure prediction using deep learning. In: 2019 56th ACM/IEEE design automation conference (DAC). pp. 1–6. IEEE (2019)
13. Pakdd2020 alibaba ai ops competition. https://tianchi.aliyun.com/competition /entrance/231775/introduction?lang=en-us, last accessed 2021/07/13
14. He, C., Liu, Y., Huang, T., Xu, F., Liu, J., Han, S., Lee, P.P., Wang, P.: Summary of pakdd cup 2020: From organizers' perspective. In: AI Ops Competition. pp. 130–142. Springer (2020)
15. Faizin, R.N., Riasetiawan, M., Ashari, A.: A review of missing sensor data imputation methods. In: 2019 5th International Conference on Science and Technology (ICST). vol. 1, pp. 1–6. IEEE (2019)
16. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research **16**, 321–357 (2002)
17. Ran, X., Su, Z.: Anomaly detection of hard disk drives based on multi-scale feature. In: AI Ops Competition. pp. 40–50. Springer (2020)
18. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: A highly efficient gradient boosting decision tree. Advances in neural information processing systems **30**, 3146–3154 (2017)
19. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. pp. 785–794 (2016)
20. Wu, Q., Chen, W., Bao, W., Li, J., Pan, P., Peng, Q., Jiao, P.: Tree-based model with advanced data preprocessing for large scale hard disk failure prediction. In: AI Ops Competition. pp. 85–99. Springer (2020)
21. Zhang, J., Sun, Z., Lu, J.: First place solution of pakdd cup 2020. In: AI Ops Competition. pp. 30–39. Springer (2020)
22. Aussel, N., Jaulin, S., Gandon, G., Petetin, Y., Fazli, E., Chabridon, S.: Predictive models of hard drive failures based on operational data. In: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA). pp. 619–625. IEEE (2017)
23. Thai-Nghe, N., Gantner, Z., Schmidt-Thieme, L.: Cost-sensitive learning methods for imbalanced data. In: The 2010 International joint conference on neural networks (IJCNN). pp. 1–8. IEEE (2010)

24. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
25. Chalapathy, R., Chawla, S.: Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407 (2019)
26. Han, S., Lee, P.P., Shen, Z., He, C., Liu, Y., Huang, T.: Toward adaptive disk failure prediction via stream mining. In: 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS). pp. 628–638. IEEE (2020)
27. Top tips for iiot storage in industry 4.0. https://technative.io/top-tips-for-iiot-storage-in-industry-4-0/, last accessed 2021/08/01
28. Durga, V.S., Jeyaprakash, T.: An effective data normalization strategy for academic datasets using log values. In: 2019 International Conference on Communication and Electronics Systems (ICCES). pp. 610–612. IEEE (2019)
29. Han, S., Wu, J., Xu, E., He, C., Lee, P.P., Qiang, Y., Zheng, Q., Huang, T., Huang, Z., Li, R.: Robust data preprocessing for machine-learning-based disk failure prediction in cloud production environments. arXiv preprint arXiv:1912.09722 (2019)
30. Yuan, X., Li, L., Wang, Y.: Nonlinear dynamic soft sensor modeling with supervised long short-term memory network. IEEE transactions on industrial informatics **16**(5), 3168–3176 (2019)
31. Lstm layer. https://www.tensorflow.org/api_docs/python/tf/ keras/layers/LSTM, last accessed 2021/08/01
32. Dense layer. https://www.tensorflow.org/api_docs/python/tf/ keras/layers/Dense, last accessed 2021/08/01
33. Conv 1d layer. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv1D, last accessed 2021/08/01
34. Maxpooling 1d layer. https://www.tensorflow.org/api_docs/python/tf/keras/layers/MaxPool1D, last accessed 2021/08/01
35. Flatten layer. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Flatten, last accessed 2021/08/01
36. Wang, Z., Yan, W., Oates, T.: Time series classification from scratch with deep neural networks: A strong baseline. In: 2017 International joint conference on neural networks (IJCNN). pp. 1578–1585. IEEE (2017)
37. Lines, J., Taylor, S., Bagnall, A.: Hive-cote: The hierarchical vote collective of transformation-based ensembles for time series classification. In: 2016 IEEE 16th international conference on data mining (ICDM). pp. 1041–1046. IEEE (2016)
38. Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A.: Deep learning for time series classification: a review. Data mining and knowledge discovery **33**(4), 917–963 (2019)
39. Global average pooling 1d layer. https://www.tensorflow.org/api_docs/python/tf/keras/layers/GlobalAveragePooling1D, last accessed 2021/08/01
40. Global max pooling 1d layer. https://www.tensorflow.org/api_docs/python/tf/keras/layers/GlobalMaxPool1D, last accessed 2021/08/01
41. Adam optimizer. https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/Adam, last accessed 2021/08/01
42. Liu, B., Shen, W., Li, P., Zhu, X.: Accelerate mini-batch machine learning training with dynamic batch size fitting. In: 2019 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2019)
43. Geng, Y., Luo, X.: Cost-sensitive convolution based neural networks for imbalanced time-series classification. arXiv preprint arXiv:1801.04396 (2018)