A Data-driven Prognostic Architecture for Online Monitoring of Hard Disks Using Deep LSTM Networks

Sanchita Basak[1], Saptarshi Sengupta[2] and Abhishek Dubey[3]

***Abstract*— With the advent of pervasive cloud computing technologies, service reliability and availability are becoming major concerns, especially as we start to integrate cyber-physical systems with the cloud networks. For example, a number of smart and connected community systems such as emergency response systems utilize cloud networks to analyze real-time data streams and provide context-sensitive decision support. Improving overall system reliability requires us to study all the aspects of the end-to-end of this distributed system, including the backend data servers. To this end, in this paper, we describe a bi-layered prognostic architecture for predicting the Remaining Useful Life (RUL) of components of backend servers, especially those that are subjected to degradation. As an exemplar, we show that our architecture is especially good at predicting the remaining useful life of hard disks. A Deep Long-Short Term Memory (LSTM) Network is used as the backbone of this fast, data-driven decision framework and dynamically captures the pattern of the incoming data. In the article, we discuss the architecture of the neural network and describe the mechanisms to choose the various hyper-parameters. Further, we describe the challenges faced in extracting effective training sets from highly unorganized and class-imbalanced big data and establish methods for online predictions with extensive data pre-processing, feature extraction and validation through test sets with unknown remaining useful lives of the hard disks. Our algorithm performs especially well in predicting RUL near the critical zone of a device approaching failure. Also the proposed architecture is able to predict whether a disk is going to fail in next ten days with an average precision of 0.8435. In future, we will extend this architecture to learn and predict the RUL of the edge devices in the end-to-end distributed systems of smart communities, taking into consideration other context-sensitive external features such as weather.**

Keywords: RUL; Long Short Term Memory; Prognostics; Predictive Health Maintenance; RNN; Reliability

I. Introduction

Cyber-physical systems powered by cloud computing framework needs to support data driven real-time, complex, distributed and importantly enough, reliable decision providing capabilities. A cyber-physical system is an assimilation of computational resources and physical processes which when combined with cloud computing based data services provides enhanced reliability, resilience, scalability and organized resource utilization. As cloud-based services provides a shareable interacting virtual architecture, so they are of crucial importance these days, supported by data-centers with immense computational power distributed over various locations. To study the overall end-to-end system reliability the cloud-based software architecture with ill-defined degradation or failure model should be taken care of rather than well-documented degradation model of physical components [1]. The multi-layered cloud computing architecture has numerous hard disk drives at its base layers on which service infrastructure, APIs, software services and distributed programming environment are built. Consequently failure in any layer propagates through the upper layers affecting overall system reliability [2], with hard disks at the ground level being the most susceptible to failure [3]. Hence the deep reliance on information and data-driven technology which is now part of everyday life comes at the price of consistently meeting these reliability challenges faced by the data-storage systems in large-scale industrial servers. In order to provide high availability of cloud services and avoid service downtime and revenue loss, it is critically important to deepen our understanding of health statistics of the hard drives to predict beforehand when they can fail and what, if any, are their tell-tale signs.

The general prediction of failure rates of hard drives follows a 'Bathtub' curve capturing infant mortality, wearing out, random failures and degradation caused due to aging. The feature characteristics causing infant mortality is not well defined and thus not easily predictable and typically can be correlated with degraded manufacturing quality. The obvious concern is to predict the age-related failures that go beyond the infant mortality region and take measures accordingly.

In some of the related research the MTTF (Mean Time to Failure) as well as the hard disk replacement rates have been modeled by statistical distributions [4] whereas some others contend that the dynamics of TBF (Time Between Failures) cannot be completely captured by standard distributions[6]. Few existing works as in [5] reported the need of a greater feature space to carry out any predictive analysis in this domain. On the other hand, with the advent of Machine Learning (ML) and Deep Learning (DL) technologies, researchers are trying to make generic ML as well as Neural Network (NN) models, that learn on their own the distribution of occurrence of failure through study of hundreds of thousands of data instead of relying on the effectiveness of the standard distributions partially capturing the trend of the data.

In this work, we provide a Recurrent Neural Network based online prediction model with Deep Long Short Term Memory (LSTM) [7] network using the Backblaze hard drive dataset [8] to predict the Remaining Useful Life (RUL) of the

[1][3]Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN 37212, USA sanchita.basak@vanderbilt.edu abhishek.dubey@vanderbilt.edu

[2]Department of EECS, Vanderbilt University, Nashville, TN 37212, USA saptarshi.sengupta@vanderbilt.edu

drives under test. Most of the existing work in this domain use cross validation models that divide the dataset randomly in train and test sets such that both the sets are normalized and abide by the same distribution. Real world test data may have no time overlap with the training dataset. As for the training data the time of failure as well as the feature values corresponding to that are known, and as such data preprocessing and normalization techniques may use that information (which is quite often the norm in training). But if the test data is also normalized in a similar manner, then the test accuracy close to that of training and validation can be obtained. However it is not representative of any real world test cases, where we do not know the time when a disk is going to fail and the corresponding feature set. Therefore, the challenge is to figure out a way to best estimate the RUL without using any future knowledge from the test set. In essence, the trained model should be able to predict the RUL of test data without any time overlap with the training data.

Contributions: In this work, we propose a two layered prognostics architecture of a proactive online prediction system for estimating the RUL of disk drives just before any job allocation at a cloud service system. The first layer of the architecture is comprised of a) data collection from the datacenter clusters, b) storing it in elastic stack, c) querying to extract data for training, d) data visualization through Kibana (not shown in the paper), e)data curation and preprocessing to feed the training data into stacked LSTM network to capture the sequential feature characteristics to learn a pattern out of it. The second layer carries out online prediction through transfer learning using the learnt model parameters to predict an impending failure of a device under test. Additionally, system health alerts can be generated for the predicted RUL under a user-defined threshold to prepare for back up and replacement of the respective drive. Based on the predicted RUL of the disks to be involved in a cloud service, the service availability throughout the job could be predicted based on their degree of failure-proneness and the reliability of the job can be assessed beforehand depending on the confidence level, which can be deduced by computing average testing accuracy [9]. Based on the online decisions, the allocation of Virtual Machines (VMs) to healthier disks with greater RUL can be made. Furthermore, if a job is already running, the online RUL prediction system can recommend whether the VMs (Virtual Machine) should be shifted to other healthier disks [10]. This consolidated architecture is capable of producing fast data-driven predictive conclusions carried out in the second layer backed by the readily available model obtained as a result of data driven computations in the first layer.

Our approach is unique in the way it proposes a hybrid architecture that answers the following challenges:

- Extracting the training set from highly unorganized feature sets with major class imbalances and establish techniques for online prediction using deep Neural Networks.
  Preprocessing and data curation steps have been adopted to extract meaningful information from the dataset where some of the failure states have similar feature sets as those of active states, making it infeasible for ML networks to learn from these features directly [11]. From the highly imbalanced dataset with a fault occurrence ratio of almost 4 out of 35000 disks of the selected model per day, time series for the devices capturing temporal progression of the feature characteristics towards failure with labels indicating time to failure have been generated prioritizing faulty devices so as to have a training set with labels of varying time to failures with almost near-equal occurrences.
- Proposing a two layered hierarchical prognostics architecture of proactive online prediction system.
  The virtual machines can be allocated or shifted to healthier disks from the currently assigned disk having less reliability all in real-time prohibiting revenue loss due to service unavailability. The process is aided by online generation of labels to indicate whether the disk is safe for VMs to be allocated to or not based on predefined threshold of failure within certain amount of time depending on the output of the computationally inexpensive online prediction system before each allocation.
- The proposed architecture also mitigates the challenge of predicting RUL of a device without any future knowledge of test data manifesting the real-world scenario along with computation of precision, recall and F-measure for several consecutive days to indicate the consistency and robustness in decision making. The system is designed in a way to work for test data without any time overlap with training data.

Although, the inferences made are based on our work on the hard disk data, the proposed architecture can be applied for predictive health maintenance of other components in cyber-physical systems to predict overall system reliability. Thus the architecture can be generalized to any time series data for capturing impending failures, detecting anomalies as well as performing continuous health monitoring of satellite missions [12], among other related online prediction systems.

The rest of the paper is organized as follows: Section II discusses the related work, Section III outlines the problem formulation and Section IV details the proposed approach. Section V reports the results followed by an involved analysis of the outcomes while Section VI provides concluding remarks and future directions.

II. Related Work

Classical approaches on error prediction of disk drives are traditionally focused on modelling the failure patterns with statistical distributions. B. Schroeder et al.[4], provided a quantitative analysis of hard disk replacement rates and discussed the statistical properties of the distribution capturing time between replacements, considering the variability of failure behavior exhibited by disk drives of same model. The authors investigated the empirical cumulative distributions of disk replacements to verify its goodness of fit with standard

exponential, Weibull, gamma and lognormal distributions generally used in reliability theory with chi-squared tests. The authors opined that the modeling of failures using Poisson assumption is a weaker approach and that the time between failure could be best modeled by the Weibull distribution.

Pinheiro et al., [5] report disk drive failure characteristics by analyzing the dependence of annualized failure rates on age group, utilization ratio of the disks, temperature, activity levels and some of the SMART indices in an in-depth study addressing life expectancy and survival rates of the hard disk drives. Their findings suggest a weaker correlation of failure with parameters such as temperature and advocate the need of much bigger signal space to be captured in powerful predictive analysis. On the other hand, Wang et al. [6], concludes that the time between failure cannot be captured through any of the standard distributions discussed above based on over 290,000 hardware failure reports from dozens of data centers. They carried out statistical testing to discover failure trends along multiple lines such as time, product, space as well as component dimensions. They refute the notion that failures are uniformly random and discuss the variability of failure rates across the entire lifetime of each component and commented on the difficulty of capturing fault trend using standard distributions.

Some relevant studies addressing failures occurring in commercial, large-scale data centers include Vishwanathan et al. [13], who carried out analyses over 100,000 servers over a 14-month time window for hardware repair logs. Birk et al. [14] experimented with both physical and virtual machine crash tickets over 10,000 servers over a period of one year spread across 5 data centers. Component level failure analyses have been carried out in [15], [16] for memory devices, in [17] for diskettes and in [18], [19] for Solid-state Drives (SSD). At the Graphics Processing Unit (GPU) level, Tiwari et al. worked on understanding errors on large-scale high-performance computing systems and their implications for system design and operation [20]. Nie et al. followed up in [21] by conducting a large-scale study of soft errors on GPUs.

The recent works on device health forecasting has adopted an array of approaches: in one study by Eker et al. [22] RUL prediction was carried out by directly comparing sensor similarity instead of using any health estimates. Similar analogies may be made for deep convolutional neural network (CNN) based RUL prediction studies by Sateesh Babu et al. [23] and recurrent neural network (RNN) based ones by O Heimes [24]. Gugulothu et al. [25] used an RNN model to generate embeddings which capture the summary trend of multivariate time series data followed by factoring the notion that embeddings for healthy and degraded devices tend to be different, into their forecasting scheme. Recent work on device health monitoring as evidenced in [26], [27] reinforce the idea of using RNNs to capture intricate dependencies among sensor observations across time cycles of dynamic period range. In [28], the authors came up with disk replacement prediction algorithm with changepoint detection in time series Backblaze data and concluded some rules for directly identifying the state of a device: healthy or faulty. Aussel et al., [11] used the same dataset to perform hard drive failure prediction with SVM, RF and GBT and discussed their performances based on precision and recall. Prediction of remaining useful lives of lithium-ion battery using quantum particle swarm optimization has been discussed in [29] and a host of recent swarm intelligence algorithms [30] [31] can be effectively applied in prediction of RUL of various devices in conjuction with other ML approaches.

Inspite of there being significant existing work on the issue, there is a clear lack of an end-to-end system architecture of online device health monitoring that leverages the learning capabilities of Long Short Term Memory Networks in order to ensure cloud service availability. We bridge this gap by proposing a two-layered end-to-end prognostics architecture of proactive online prediction system with real-time decision making capabilities for allocation of VMs to hard disks, ensuring smooth operation of cloud based cyber physical system.

III. Problem Formulation

In this work, we seek to provide a recurrent neural network based RUL prediction model by using a Deep Long Short Term Memory (LSTM) [7] network. The Backblaze hard drive dataset [8] containing statistics and trends from January 2013 to June 2018 has been considered for this work. The dataset reports the snapshot of 30 different S.M.A.R.T. indices including both raw and normalized values for each operational hard drive model from various companies. Performance metrics for each of the active hard drives are captured through their SMART (Self-Monitoring, Analysis and Reporting Technology) indices reported once every twenty-four hours. We captured the data sequences from January 2017 through December 2017 containing information about 91,243 devices manufactured by various companies out of which we chose to work with the device model ST4000DM000 from Seagate due to the following reasons:

- Failure statistics from 2017 suggest Seagate devices failed the most and
- Out of all device models ST4000DM000 from Seagate contributed to most of the failures.

As a result, we can generate maximum number of training examples capturing trends leading to failure by choosing this model. Furthermore, different companies do not report the same SMART indices. Seagate model ST4000DM000 reports 24 SMART indices out of 30. The raw data is made to undergo requisite preprocessing following which the extraction of features, strongly correlated with the occurrence of a failure are carried out. The data is then used to generate training examples and a deep LSTM network is trained for regression and subsequently tested to predict the Remaining Useful Life of a device under test. Figure 1 describes the entire bi-layered architecture of the RUL prediction system with both pre-training and online prediction modules.

So out of the F features reported for each device, we select f of them as shown in table 1, through feature selection

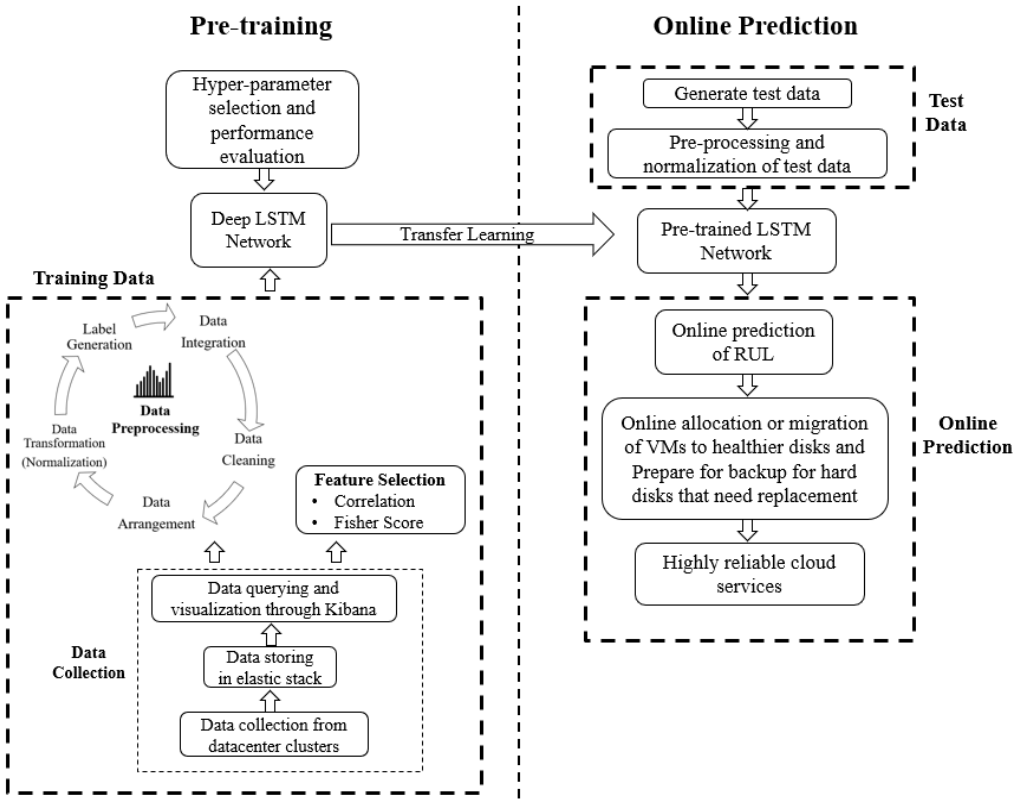

Fig. 1. Two-layered prognostic architecture of the online RUL prediction system

TABLE I
SUMMARY OF SMART FEATURES USED

ID	Attribute Name	Description
SMART 7	Seek error rate	Frequency of the errors during disk head positioning and rises with approaching failure.
SMART 9	Power-on-hours count	Estimated remaining lifetime, depending on the time a device was powered on. Raw value indicates the actual powered-on time, usually in hours.
SMART 240	Head flying hours or transfer error rate	Time spent during the positioning of the drive heads
SMART 241	Total LBAs written	Related to the use and hence indicating the aging process of hard drives
SMART 242	Total LBAs read	Related to the use and hence indicating the aging process of hard drives

methods discussed in Section IV, such that, $f \subseteq F$. Out of various devices of model ST4000DM000, only a few have failed in the time duration January to December 2017. With device instances indicated by i, we can formulate a multivariate time series data $A_i \subseteq \mathbb{R}^{j \times f}$ where each data matrix A_i for each device i is of dimension $(j \times f)$. All the time instances have been discretely sampled at an interval of one day. If the data collection starts at day T_0 and ends at day T_z where a device can fail in any day T_f in between, such that $T_0 \leq T_f \leq T_z$, the time index in days i.e., j varies from T_0 to T_f. For each individual device i the data matrix is organized starting from the feature set corresponding to failure day T_f. Then the time series is traversed back to append rows such that the j_{th} row of A_i corresponds to feature sets at the j_{th} day, and the process is continued upto the day T_0. For normalization purposes discussed in section IV, we need to have similar length for each A_i matrix, such that the time index j varies from T_f upto a fixed length of a sequence back, which in our case is upto 150 days prior to failure. In order to accomodate 150 days of data prior to failure we collected the failure instances of Seagate model ST4000DM000 from June to Dec 2017 which consists of almost 592 failing devices in that time period each with information of feature sets 150 days prior to failure. So now the training set can be represented as TRD $\{A_i \subseteq \mathbb{R}^{j \times f} \mid j = T_f - 150, ..., T_f\}$. So TRD comprises of all such A_i. The corresponding training labels attached to each row of data for each A_i is $T_f - j$ denoted by TRL.

For testing purposes, a test set has been formulated such that TSD $\{B_i \subseteq \mathbb{R}^{m \times f} \mid m = T_c - 150, ..., T_c\}$ for each test instance i. T_c is the current time when the device is active and when we want to test the device for its RUL, such

that $T_0 < T_c < T_f$. So the problem boils down to predicting the RUL of a device i, i.e., the remaining time it is active from the time instant T_c upto failure.

IV. Our Approach

A. Data Cleaning and Arrangement

The files containing data for each day contain the date, the manufacturer-assigned serial number of the drive, model number of the drive, the drive capacity in bytes and the status which indicates 1 to denote failure and 0 to denote that the device is active. The data for a specific model is collected at first from all the files indicating hard drive SMART statistics for each day for all hard drives. Starting from the last day of data collection for a specific model when it failed, the serial number corresponding to the failed model is noted and traversed back in chronological order through all the files to look for the SMART indices for that specific serial number of that model when it was active/working. The examples are saved in a matrix in a form described in setion III, starting with the features in 1st row corresponding to the failure day and the next rows containing features corresponding to previous days of failure each with increasing RUL from the previous ones. Such a collection of SMART indices for specific model in a chronological order is to be saved as input data matrices for training the LSTM network. Appropriate labels indicating remaining unit life have been generated and appended to the saved matrices. Thus, all the examples for training the network have been generated.

B. Feature Selection

As the dataset is highly unorganized we do not observe the same set of features causing failure over all devices of the same model. Also, all the 24 features are not at all correlated with failure. So, an effective feature selection is carried out by both correlation and Fisher score methods. At first, correlation coefficients are generated to study the dependence of failure on each individual feature to observe how the feature trends change as the device progresses towards failure. If U and V are two random variables denoting values of each feature and label of RUL corresponding to them respectively, then the correlation between them can be expressed as:

$$R_{UV} = \frac{C_{UV}}{\sqrt{\sigma_U * \sigma_V}} \tag{1}$$

Where C_{UV} denotes the co-variance of the two random variables and σ_U, σ_V denote standard deviations of the variables U and V. The features are sorted in the order of higher absolute values of correlation score to extract the features highly correlated to failure.

Next we choose another supervised filter feature selection method, Fisher score [32], which focuses on features having better distinguishing capability in terms of greater variance of values among different classes and more similar feature values within a particular class. An input set of features x_k with d data samples, $k = 1, 2, ...d$, along with their labels y_k comprising of L different classes, such that $y_k \in 1, 2, .., L$, and the number of data instances in each class is d_k, we calculate the mean of all data samples of n_{th} feature as μ_n, while μ_n^k and σ_n^k are the mean and variance of the n_{th} feature for class k. Hence the Fisher score of the n_{th} feature F_n can be described as:

$$F_n = \frac{\sum_{k=1}^{L} d_k(\mu_n^k - \mu_n)^2}{\sum_{k=1}^{L} d_k(\sigma_n^k)^2} \tag{2}$$

The numerator $\sum_{k=1}^{L} d_k(\mu_n^k - \mu_n)^2$ denotes the inter-class variance and the denominator $\sum_{k=1}^{L} d_k(\sigma_n^k)^2$ denotes the intra-class variance with respect to n_{th} feature. Then the features are sorted in the order of higher Fisher score as features with higher Fisher scores tend to exhibit better differentiating capacity among classes.

Through these two processes the best common five features have been selected out of twenty-four. This subset of features have produced significantly better training and validation accuracy averaged over several independent trials. Hence they are ultimately selected for training.

C. Normalization

As discussed earlier, the features corresponding to failure day varies widely, hence the LSTM network cannot be directly trained with these vastly varying features. But the trend that is commonly observed among all the features irrespective of their values, is that those significant five features have an increasing trend while approaching failure. So, min-max normalization i.e., normalization of features from 0 to 1 can be an option in this case to capture the increasing trend of features approaching failure.

As different devices fail on different days throughout the year, the saved matrices containing feature statistics have differing numbers of days of data starting from the failure day and backtracking till the starting day of data collection. Now, if they are normalized between 0 to 1, then varying number of days of data will be distributed between 0 to 1 introducing skewness or asymmetry in data distribution. So, the features expressed in dissimilar fractions will correspond to similar labels of remaining useful lives or vice-versa, making it difficult as well as inappropriate for the network to associate the feature sets to the labels given. So, the length of the data matrices should be of same size in order to establish the fact that after normalization, the similar set of features are associated with similar remaining useful lives across all devices. Hence, for this work each matrix is designed such that it contains data from failure day for a disk until 150 days prior to it and then these 150 days of data is normalized for each device.

A sequence length of 25 is chosen as the look back sequence of LSTM (discussed later in this section) with a trade-off between minimum required sequence length, time for training and testing and accuracy of prediction, and the data is organized in the form

(number of examples, sequence length, features)

Each (25, 5) matrix has the remaining useful life corresponding to the features of the latest day in a matrix such

that the network learns to predict remaining useful life at any timestamp given past 25 days of data inputs. Thus, training examples along with the labels are generated from which 5% of the data are used for validation purposes. The training data consists of (71072, 25, 5) examples which are fed to a stacked LSTM network, the detailed description of which is provided subsequently. Table 2 lists all the symbols used in this paper along with their description.

TABLE II
SUMMARY OF SYMBOLS USED

Symbols	Meaning
TRD	Training data
TRL	Training label
TRL_i	Training labels for A_i
TSD	Test data
f	Selected features
A_i	Training data matrix for each device i
T_0	Day when the data collection starts
T_f	Day when the device fails
T_z	Day when the data collection ends
T_c	The current time when the device is active and when we want to test the device for its RUL
B_i	Test data matrix for each device i
j	Time instances for training data varying from $T_f - 150, ..., T_f$
m	Time instances for test data varying from $T_c - 150, ..., T_c$
R_{UV}	Correlation of a feature with failure with U and V being two random variables denoting each feature values and training labels.
C_{UV}	Co-variance of two random variables U and V
F_n	Fisher score of feature n
μ_n^k	Mean of the n_{th} feature for class k
σ_n^k	Variance of the n_{th} feature for class k
d_k	The number of data instances in each class
Q	The entire training data matrix of shape (number of examples, sequence length, features) to be fed into LSTM
L	The entire training label having a shape of (number of examples,1) corresponding to each (sequence length, features) dimensional matrix of Q
H	Preprocessed test data in the form of (Test instances, sequence length, features) to be fed into pretrained LSTM model

D. Preparation of Test Data

For testing purposes, a time series data for an active device can be taken and an estimate when the device can fail can be given. The test data matrices are designed as discussed in section III.

For the training purposes, the data has been normalized from 0 to 1 including the features on day of failure with the feature corresponding to the failure day treated as 1, while in case of testing the actual failure characteristics is not known beforehand. But as the increasing trend of features approaching failure has been observed, if a time series data for an active device with unknown RUL is taken then the maximum attainable value of a feature from the historical data can be found. Considering that value to be maximum, the features in the given data series can be normalized and can be fed into the LSTM network to predict when it is going to fail.

In this process a lot depends on the values of features which will be taken as maximum and w.r.t. which the current feature values will be normalized. For finding the optimal value for setting as maximum in the normalization process we look two months back for any currently active device under test. In spite of fluctuations in the feature data within a shorter period of time, the overall moving average of feature values taking any data segment comprising of few months into account, tend to increase over months. So consideration of past two months of data in fetching historical max is appropriate in the sense that it is not too short to be affected by noisy fluctuations of small sequence as well as not too large so as to balance the computational overhead and diminish redundancy.

The distribution of data for a duration of any two consecutive months are mostly similar for a particular feature. Figures 2 through 6 show the box plots for each feature to show a typical distribution of that feature for any two consecutive months. On an average the box plots describe that feature 1 has most of its data concentrated within lower 10% of the entire data range, with outliers occupying the rest of the dataspace. Features 2 and 3 have data upto the upper quartile within almost 80% of the entire data spectrum. Feature 4 also has many outliers with mid 50% of data squeezed between 30% of the data spectrum and the rest contributing to the outlier values. Feature 5 has 75% of its data within 50% of its data spectrum indicating a somewhat more sparse distribution in the 4th quartile.

So, if we consider the maximum of each feature data and normalize the test data with respect to that, then in most of the cases, the normalized feature values will result in lesser fractions than it would have been, if normalized through its actual failing features. Because maximum value of a feature may be just an outlier in the distribution. This results into much greater Remaining Useful Life (RUL) than actual ones. This is referred to as Prediction Strategy 1.

To resolve this issue the data is sorted at first and normalized considering the values at 3rd quartile (75_{th} percentile) of the entire data spectrum as the maximum feature value. The quartiles are chosen to work with so as to keep the generality of normalization over data from any part of the year, as in spite of having similar distribution, they are not identically distributed and hence not possible to find a specific percentile that suits all. By choosing 75_{th} percentile we are keeping close to the historical max, yet not eliminating vast amount of data from consideration, and also eliminating the possible outliers. This kind of normalization resulted into significantly better approximations of the RUL as shown in Prediction Strategy 2.

We can also consider the projection of the feature values at the upper quartile on the entire data spectrum to find the fraction of the entire data range within which most of the values are concentrated and take that as maximum and normalize the test features through that, which is essentially equivalent to strategy 2. Another approach which was considered, was to adaptively define the quartiles for each individual features with respect to which the corresponding

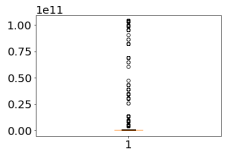

Fig. 2. Feature 1: SMART 7 historical data distribution

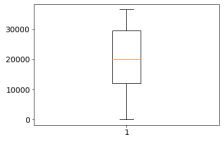

Fig. 3. Feature 2: SMART 9 historical data distribution

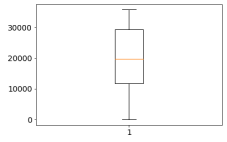

Fig. 4. Feature 3: SMART 240 historical data distribution

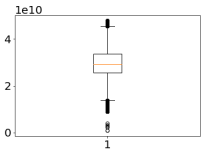

Fig. 5. Feature 4: SMART 241 historical data distribution

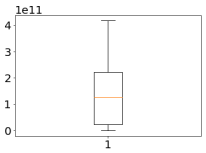

Fig. 6. Feature 5: SMART 242 historical data distribution

features of test data would be normalized. As feature 1 and 5 have most of the outliers and data sparsity respectively, the sorted feature 1 and 5 data were chosen to be normalized with respect to 1st (25th percentile) and 2nd quartile (50th percentile) of the entire data spectrum respectively with other features using 3rd quartile of the range as maximum. But the method was not able to produce more accurate predictions primarily because of elimination vast amount of data from consideration, resulting into much lesser prediction than actual RUL for devices having shorter actual RUL and showed insignificant improvement for devices having longer actual RUL as compared to what was predicted through strategy 2. So, this method cannot be applied in a generalized way to predict varying RUL across multiple devices.

E. Long Short Term Memory (LSTM)

LSTM is a widely used variant of recurrent neural network proposed by Hochreiter and Schmidhuber [7]. RNNs are capable of capturing dynamic temporal behavior in time series data by the use of shared parameters while traversing through time. At any point of time RNN takes the current input, captures hidden state information for previous time steps upto a given sequence length and generates output according to the task given, i.e., classification, regression or prediction of data for the next timestamp.

The exploding and vanishing gradient problems encountered by RNN has been solved in LSTM. A self-feedback LSTM unit associated with input, output and forget gate control the motion of information through the gating mechanisms. As the name suggests, the input and output gate of each memory cell in LSTM directs the inputs and outputs flowing in and out from the cell respectively, whereas the forget gate decides upon which information needs not to be memorized anymore. The values of the gates range within 0 to 1 based on the output of the sigmoid activation function. The input feature at a timestamp combining current input, previous hidden state information and shared parameters with $tanh$ activation function is multiplied with the input gate coefficient and updates the value of the memory cell combining the forget gate coefficient controlled previous timestamp's value of the cell. Any hidden output state is thus a result of output gate-controlled $tanh$ activated memory cell value, which is then used for producing outputs.

1) Hyper-parameter Selection:

a) Number of Units in each layer: In this paper we have used a stacked architecture of LSTM comprising of two layers. The first layer is an LSTM layer with 100 units

TABLE III
SUMMARY OF LSTM MODEL 1 USED

Layer (type)	Output Shape	Parameters
LSTM layer 1	(None,25,100)	42400
Dropout layer 1	(None,25,100)	0
LSTM layer 2	(None,50)	30200
Dropout layer 2	(None,50)	0
Dense	(None,1)	51

TABLE IV
SUMMARY OF LSTM MODEL 2 USED

Layer (type)	Output Shape	Parameters
LSTM layer 1	(None,25,100)	42400
Dropout layer 1	(None,25,100)	0
LSTM layer 2	(None,100)	80400
Dropout layer 2	(None,100)	0
Dense	(None,1)	101

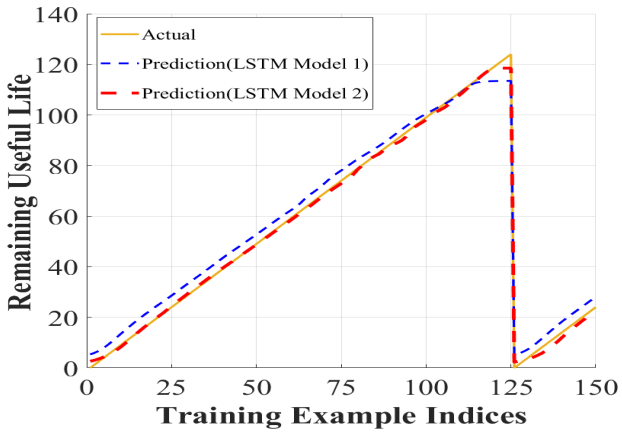

Fig. 7. RUL prediction with LSTM Model 1 and 2

followed by the second LSTM layer with 50 units which is referred to as LSTM Model 1. Dropout is also applied after each LSTM layer to control overfitting. Final layer is a dense output layer with single unit with linear activation function. Thus, a LSTM network has been designed for prediction purposes over 71072 examples with 5% of that reserved for validation purposes. The network reports mean squared training error of 51.2935 over 67518 samples and mean squared validation error of 10.8463 on 3554 validation examples.

Along with this LSTM Model 1, another deep LSTM

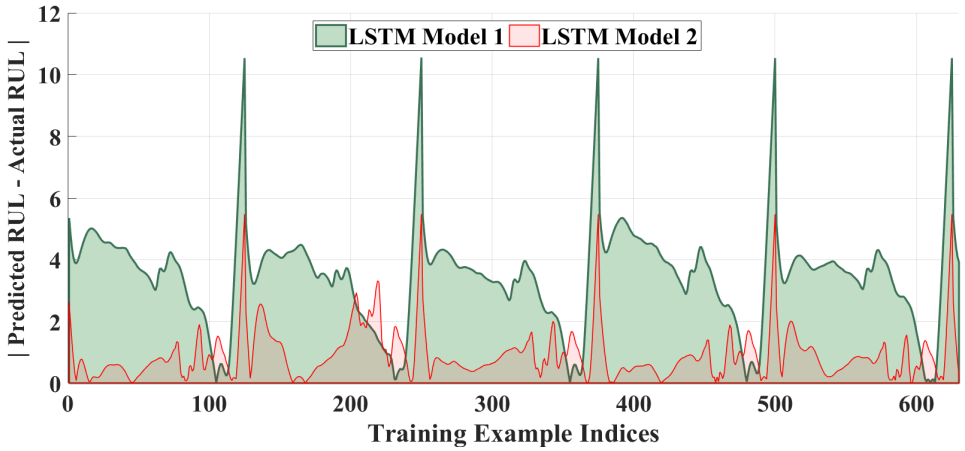

Fig. 8. Variation of Absolute Error Over Training Examples

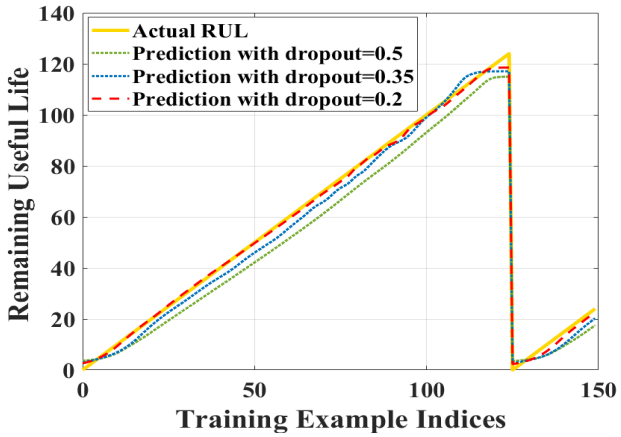

Fig. 9. Effect of variation of dropout ratio on estimation of RUL

network with 100 units in both the layers have been built with significant improvement in both training and validation accuracy. LSTM Model 2 reports mean squared training error of 26.8489 over 67518 samples and mean squared validation error of 4.2417 on 3554 validation examples. Table 3 and 4 provides the summary of models 1 and 2 respectively. Figure 7 shows the variation of absolute error between actual and predicted values over training examples between LSTM Model 1 and 2. Figure 8 also indicates the supremacy of LSTM Model 2 over Model 1 as the predicted RUL follows the actual ones more closely in case of Model 2. Hence, LSTM Model 2 is finally used for RUL prediction of disk drives.

b) Selection of Dropout Ratio: Dropout is an important regularization parameter to control over-fitting. Very high values of dropout ratio may shut down most of the units of hidden layers resulting into non-optimal decision boundaries. On the other hand, if the dropout ratio is very low for all the layers, then the neural network behaves as an unregularized one leading to overfitting of the training data and greater error in validation or test set. So, a moderate dropout ratio gives optimal or near optimal decision boundary. The effect of variation of dropout ratio on the accuracy of estimation of Remaining Useful Life has been shown in Figure 9. According to the results we chose dropout ratio of 0.2 for both the layers.

To prove the fit of LSTM to this problem we also trained a Naive Bayes classifier which is a probabilistic classifier based on Baye's Theorem. The network has good training and validation accuracy because of the normalized structure of training data and distinct labels associated with distinct feature values. But at the time of testing for devices it fails completely as it is assigning similar label i.e., similar RUL to whatever input is given. As test data distribution is different than that of training data and as there is no dependence on the previous time sequences for a given input, so that it can learn from the temporal dynamics, it is naively assigning a value to whatever data is being fed as it cannot fit it perfectly to any of the probabilistic decisions learned while training. This is due to the fact that the combination of features for a given test input do not necessarily match with the combinations with which it was trained. Therein, comes the utility of transfer learning used in LSTM, where the model is being able to predict the labels of test data with similar but different distribution from training data.

The online prediction module used in our approach is computationally inexpensive as we are just feeding the test data into pre-trained LSTM model and obtaining the remaining useful life instantly. Algorithm 1 describes the entire approach for prediction of RUL of a device.

V. Results and Discussion

A. Variation of Predicted RUL over Various Devices

For the testing purpose, the data for any active device is taken at a randomly chosen day when the device is working and is considered as the current date T_c. For validation of the prediction, the actual failure day, which is certain number of days after the current date is noted, but not used

in preparation of test data in any form and the subset of data from current date leading to failure day is completely ignored. Now past 150 days of data from the current date, is normalized as discussed in prediction strategy 2. Then a look back sequence length of 25 similar to that used in training is chopped off starting from the current date and fed to the pre-trained LSTM model which is able to produce online decision about the RUL of the device. The actual and predicted RUL of the devices are shown in Fig 10 where different disk drives having various remaining useful lives have been considered to show the performance of the algorithm over a diverse range.

Algorithm 1 Algorithm for Estimation of RUL

Input: Training Data : TRD $\{A_i \in \mathbb{R}^{j \times f}\}$ for each i; Training Labels: TRL $(T_f - j)_i$ for each i
Test Data : TSD $\{B_i \in \mathbb{R}^{m \times f}\}$ for each i
Output: Predict RUL for each i from time T_c

Preprocessing:
length(A_i) for each i should be same for normalization. TRD comprises of all such A_i.

Training:
for each A_i in TRD **do**

  for $p = 0$ to $(length(A_i) - sequencelength)$ **do**
    $q = A_i(p) : A_i(p + sequencelength)$
    append q to Q
    $l = TRL_i(p)$
    append l to L
  end for
end for
return Q
return L
Feed Q and L to LSTM network as training data and labels respectively
Save trained model

Preprocessing for Testing:
Perform the preprocessing steps on test data
Extract test data in the form H=(Test instances,sequence length,features)

Online Prediction:
Feed H in saved model
Output the RUL

With the prediction Strategy 2, the test result in Fig. 10 shows better accuracy of prediction for shorter actual RUL and lesser accuracy for longer actual RUL. When actual RUL is short, then its features have already reached nearer to the maximum values of the features causing failure. So, the variation in the feature space between the current feature values and that of the 75th percentile of the past data is limited as the two values described above are close to each other making way for lesser error in prediction. On contrary, when the actual RUL is larger, then its feature values are much smaller than those causing failure, leading to higher variation in feature space between the current feature values and that of the 75th percentile of the past data, bringing about more variability in possible position of the actual normalized features causing failure between this range, making more room for error in prediction of the RUL. Overall, prediction strategy 2 produces good estimation of RUL for the devices with impending failures which is more important and provide a fair approximation of devices that are going to fail later.

B. Variation of Predicted RUL over a single hard disk over progression of time

The revenue loss caused due to allocating VMs to faulty disks or not migrating them to healthier ones on not being able to identify disk faults apriori or migrating VMs form disks which are not going to fail within the allocated service time but misidentified as to be failure prone by the prediction algorithm can be modeled as:

$$RevenueLoss = loss_{nfp} \times nfp + loss_{nfn} \times nfn \quad (3)$$

Where nfp and nfn are the number of false positives and false negatives respectively and $loss_{nfp}$ and $loss_{nfn}$ are losses incurred due to false positives and false negatives respectively. Depending on the application, more importance can be given to precision or recall depending on the costs associated with $loss_{nfp}$ and $loss_{nfn}$.

As we are more interested in predicting imminent failures, the prediction accuracy for devices which are going to fail sooner is more critical than those that having greater RUL. Based on the online prediction results of disk drives having imminent failure the decision on the allocation of jobs in the cloud architecture is to be taken. So the prediction accuracy is much sensitive in this region as smaller number of false positives or false negatives can incur greater revenue loss. It is of critical importance to determine how the uncertainty in prediction of RUL for any device reduces as it approaches its end of life and remains to be analyzed. Figure 11 shows a graph of prediction of RUL for a single device at different time instances and indicates greater accuracy of prediction near the time region of actual failure.

C. Variation of Precision, Recall and F1 score over time

The Precision and Recall of prediction of RUL for numerous devices have been shown in Figure 12 based on the fixed threshold of whether a device is going to fail in next ten days. The process is carried on for seven consecutive days to get a time series variation of these measures over a fixed threshold. The plot shows an average Precision of 0.84, Recall of 0.72 and F1 score of 0.77. The flat nature of the curve indicates the consistency and robustness in decision making using the model over several consecutive days.

In most of the existing research [33] [11] cross validation models are used that divide the dataset randomly in train and test sets such that both the sets are normalized abiding

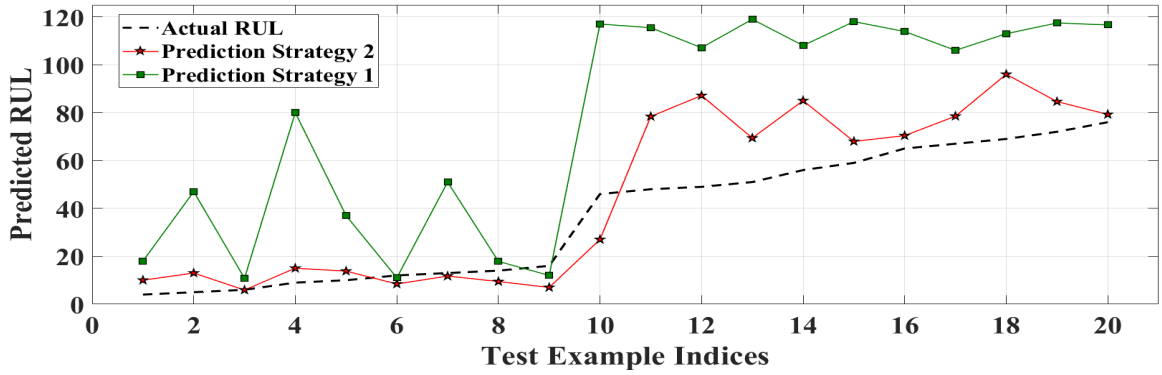

Fig. 10. The actual and predicted Remaining Useful Lives of various devices under test

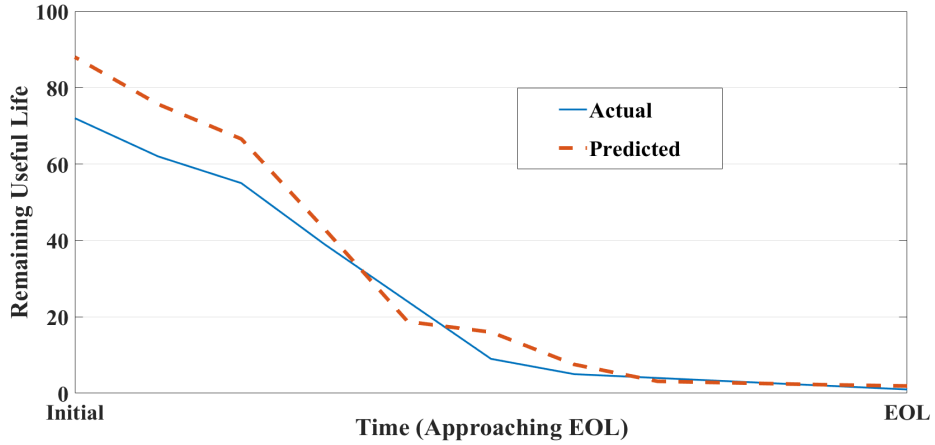

Fig. 11. Reduction in Uncertainty in Prediction As a Device Approaches EOL

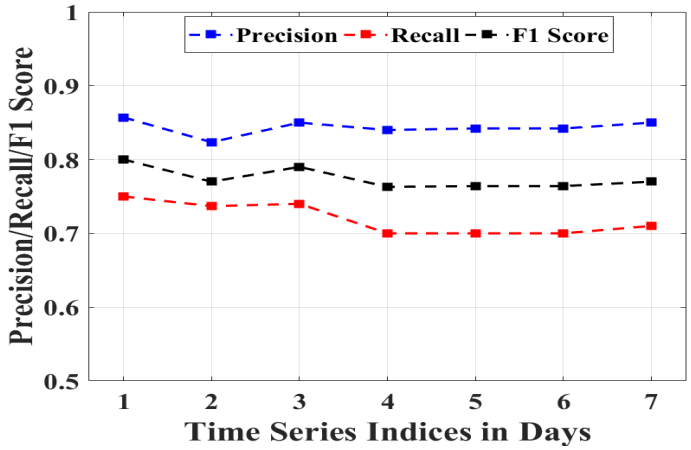

Fig. 12. Precision and Recall of Prediction of RUL over Seven Consecutive Days

by the same distribution. In real world the test data may have no time overlap with the training dataset and also we cannot use any future information in the testing process. If we would have drawn the test data distribution in a similar fashion as that of the training one, using future information, then we could have had much better prediction of the RUL as shown in Figure 13. But as this does not indicate the actual efficiency of the prediction model, this cannot be used in a real life scenario. In [11] the authors claimed that the best performance on Backblaze dataset was shown

by Random Forest (RF) using all features. The Precision and Recall values based on the threshold of device failure within 10 days were recorded as almost 0.93 and 0.6 using cross-validation, whereas we obtained an average precision of 0.84 and recall of 0.72 using the decision threshold of device failure within 10 days without using cross-validation and any future information in testing process. Hence our proposed architecture is able to mitigate the challenge of predicting RUL of a device without any future knowledge of test data with acceptable decision outputs manifesting real-world scenarios.

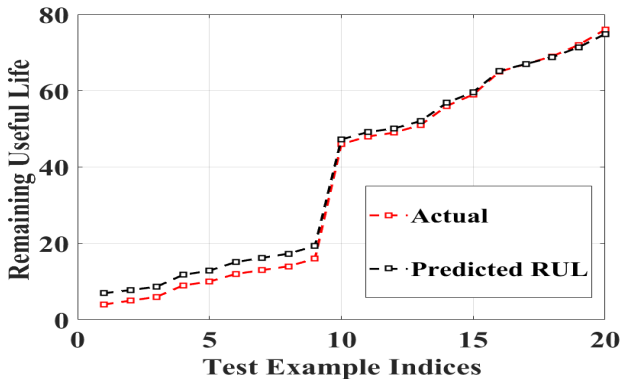

Fig. 13. Evaluation Results Using Future Information While Testing

The prediction mechanism discussed in this paper is sufficiently fast to provide a decision about which of the disks can be safely involved in the cloud service to ensure high service availability. This helps to take online decisions to allocate the Virtual Machines on healthier disks depending on their failure proneness. The reliability of an already running job can also be predicted through online prediction of RUL of all of their respective components using the same strategy discussed here. Also, live migration of Virtual Machines from failure-prone to healthier disks based on the online prediction is possible without any interruption in client service.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we propose a two-layered architecture comprising of pre-training and online prediction module using deep LSTM network. As error in disk drives is one of the driving factors leading to service/resource inaccessibility, an online Remaining Useful Life prediction model has been put up. The proposed framework based on SMART indices collected from Backblaze dataset provides effective approximations of the disk health which can subsequently be used in online allocation of jobs in a cloud service system. In future, we expect to use this generalized framework in various related applications with data specific modifications focused on online relaying of decisions of critical importance.

REFERENCES

[1] S. Nannapaneni, S. Mahadevan, S. Pradhan, and A. Dubey, "Towards reliability-based decision making in cyber-physical systems", In 2016 IEEE International Conference on Smart Computing (SMARTCOMP), May 2016, pp. 16

[2] R. Jhawar and V. Piuri, "Fault tolerance and resilience in cloud computing environments", Computer and Information Security Handbook, pp. 128, 2014.

[3] M. Botezatu, I. Giurgiu, J. Bogojeska, D.Wiesmann, "Predicting Disk Replacement towards Reliable Data Centers", Conference of Knowledge discovery and data mining KKD2016, August 13 - 17, 2016, San Francisco, CA, USA.

[4] B. Schroeder and G. A. Gibson. "Disk failures in the real world: what does an MTTF of 1,000,000 hours mean to you?", In Proceedings of the 5th USENIX Conference on File and Storage Technologies, February 2007.

[5] E. Pinheiro, W. D. Weber, and L. A. Barroso, "Failure trends in a large disk drive population", In Proceedings of 5th USENIX Conference on File and Storage Technologies (FAST 2007), San Jose, CA, February 2007.

[6] G. Wang, L. Zhang and W. Xu, "What Can We Learn from Four Years of Data Center Hardware Failures?", 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Denver, CO, 2017, pp. 25-36.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory", Neural Comput. 9, 17351780.

[8] Hard drive smart stats. https://www.backblaze.com/blog/hard-drive-smart-stats/.

[9] A. Verma, L. Pedrosa, M. Korupolu, D. Oppenheimer, E. Tune, and J. Wilkes, "Large-scale cluster management at Google with Borg", In European Conference on Computer Systems (EuroSys), 2015.

[10] Y. Xu et al., "Improving Service Availability of Cloud Systems by Predicting Disk Error", Annual Technical Conference USENIX, 2018.

[11] N. Aussel, S. Jaulin, G. Gandon, Y. Petetin, E. Fazli and S. Chabridon, "Predictive Models of Hard Drive Failures Based on Operational Data", 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, 2017, pp. 619-625.

[12] F. Sun, A. Dubey, C. Kulkarni, N. Mahadevan, A. G. Luna, "A Data Driven Health Monitoring Approach to Extending Small Sats Mission", In Conference Proceedings, Annual Conference of The Prognostics And Health Management Society 2018

[13] K. V. Vishwanath and N. Nagappan, "Characterizing cloud computing hardware reliability", In ACM Symposium on Cloud Computing, 2010.

[14] R. Birke, I. Giurgiu, L. Y. Chen, D. Wiesmann, and T. Engbersen, "Failure analysis of virtual and physical machines: patterns, causes and characteristics", In International Conference on Dependable Systems and Networks (DSN14). IEEE, 2014.

[15] J. Meza, Q. Wu, S. Kumar, and O. Mutlu, "Revisiting memory errors in large-scale production data centers: Analysis and modeling of new trends from the field", In International Conference on Dependable Systems and Networks (DSN15). IEEE, 2015.

[16] L. Bautista-Gomez, F. Zyulkyarov, O. Unsal, and S. Mcintosh-Smith, "Unprotected computing: a large-scale study of DRAM raw error rate on a supercomputer", In International Conference for High PERFORMANCE Computing, Networking, Storage and Analysis, 2016.

[17] J. Yang and F. B. Sun, "A comprehensive review of hard-disk drive reliability", In Reliability and Maintainability Symposium, 1999. Proceedings

[18] J. Meza, Q. Wu, S. Kumar, and O. Mutlu, "A large-scale study of flash memory failures in the field", ACM Sigmetrics Performance Evaluation Review, vol. 43, no. 1, pp. 177190, 2015.

[19] I. Narayanan, D. Wang, M. Jeon, B. Sharma, L. Caulfield, A. Sivasubramaniam, B. Cutler, J. Liu, B. Khessib, and K. Vaid, "SSD failures in datacenters: What, when and why?", ACM Sigmetrics Performance Evaluation Review, vol. 44, no. 1, pp. 407408, 2016.

[20] D. Tiwari et al., "Understanding GPU errors on large-scale HPC systems and the implications for system design and operation", In HPCA 2015. IEEE, 2015.

[21] B. Nie, D. Tiwari, S. Gupta, E. Smirni, and J. H. Rogers, "A large-scale study of soft-errors on GPUs in the field", in HPCA 2016. IEEE, 2016.

[22] O.F. Eker, F. Camci and I. K. Jennions, "A similarity-based prognostics approach for remaining useful life prediction", The 2nd European Conference of the Prognostics and Health Management (PHM) Society, Nantes, France, 8-10 July 2014, vol. 5, no. 11, 2014.

[23] G. S. Babu, P. Zhao, and X. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life", In International Conference on Database Systems for Advanced Applications. Springer, 214228, 2016.

[24] F. O Heimes, “Recurrent neural networks for remaining useful life estimation”, In Prognostics and Health Management, 2008. PHM 2008. International Conference on. IEEE, 16. 2008.

[25] N. Gugulothu, V. TV, P. Malhotra, L. Vig, P. Agarwal, G. Shroff, “Predicting remaining useful life using time series embeddings based on recurrent neural networks”, International Journal of Prognostics and Health Management (ijPHM) 9(1): 004.

[26] P. Filonov, A. Lavrentyev and A. Vorontsov, “Multivariate Industrial Time Series with Cyber-Attack Simulation: Fault Detection Using an LSTM-based Predictive Data Model”, NIPS Time Series Workshop 2016, arXiv preprint arXiv:1612.06676 (2016).

[27] P. Malhotra, et al., “LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection”, In Anomaly Detection Workshop at 33rd ICML, 2016, arXiv:1607.00148 (2016).

[28] M. Botezatu, I. Giurgiu, J. Bogojeska, D.Wiesmann, “Predicting Disk Replacement towards Reliable Data Centers”, Conference of Knowledge discovery and data mining KKD2016, August 13 - 17, 2016, San Francisco, CA, USA.

[29] J. Yu et al., “Remaining useful life prediction for lithium-ion batteries using a quantum particle swarm optimization-based particle filter”, In Quality Engineering, 2017, Vol 29, pp.536-546.

[30] S. Sengupta, et al., “Particle Swarm Optimization: A Survey of Historical and Recent Developments with Hybridization Perspectives”, Machine Learning and Knowledge Extraction. 2018; 1(1):157-191.

[31] S. Sengupta, et al., “QDDS: A Novel Quantum Swarm Algorithm Inspired by a Double Dirac Delta Potential”, Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, Nov 2018, Bengaluru, India. (Accepted) arXiv:1807.02870 [cs.MA]

[32] R. Sheikhpour, M. A. Sarram, S. Gharaghani and M. A. Z. Chahooki, “A survey on semi-supervised feature selection methods”, Pattern Recognit., vol. 64, pp. 141158, 2017

[33] F. Mahdisoltani, I. Stefanovici and B. Schroeder, “Proactive error prediction to improve storage system reliability”, In 2017 USENIX Annual Technical Conference (USENIX ATC 17) (Santa Clara, CA, 2017), USENIX Association, pp. 391402.