Finding soon-to-fail disks in a haystack

Moises Goldszmidt
Microsoft Research

Abstract

This paper presents a detector of soon-to-fail disks based on a combination of statistical models. During operation the detector takes as input a performance signal from each disk and sends and alarm when there is enough evidence (according to the models) that the disk is not healthy. The parameters of these models are automatically trained using signals from healthy and failed disks. In an evaluation on a population of 1190 production disks from a popular customer-facing internet service, the detector was able to predict 15 out of the 17 failed disks (88.2% detection) with 30 false alarms (2.56% false positive rate).

1 Introduction

The ability to provide advance warning of hard disk failures is extremely useful in internet and cloud services. For cloud services this ability enables completely uninterrupted service as preventive reallocation may be executed prior to imminent disk failures. For internet services such as email, content (news, search, movies), and ecommerce, advance warning may trigger immediate replication or replacement which will increase availability and reliability statistics, and may even reduce replications and prevent data loss. In addition to preventive actions, one can imagine more invasive ones involving perhaps intrusive diagnostics and repairs. IT departments in regular businesses and individual users would obtain similar benefits from having a detector producing these alarms.

In this paper we describe, characterize, and evaluate a software based detector of soon-to-fail disks. This detector which we call *D-FAD* (for *Disk Failure Advance Detector*), takes as input a performance signal from each disk and produces an alarm according to a combination of statistical models. The parameters in these models are automatically trained from a population of healthy and failed disks, using machine learning techniques.

Predicting the future is notoriously difficult. There is a trade-off between the ability to accurately predict failure (indicated by the detection rate), and the number of false alarms (indicated by false positive rate). This trade-off is reflected in the tension between the cost of ignoring an alarm for a failing disk and the cost of unnecessary action in reaction to a false one. Ignoring an alarm for an impending failure will result in a degradation of the quality, availability, and reliability of the service. Acting on a false alarm will result on unnecessary costs related to, for example, migrating users in a cloud service. The decision about how to act on this trade-off is entirely dependent on economic and business considerations of the service. Thus, we report on the detection and false positive rates of *D-FAD* providing the necessary information to act on this trade-off.

When applied to a population of 1190 production disks from a popular 24x7 customer-facing internet service *D-FAD* predicts 15 out of 17 failed disks (88.2% detection rate), with 30 false alarms (2.56% false positive rate). In the case of the preventive action being moving users in a cloud service, this translates to 30 additional "moves" replicas for guaranteeing uninterrupted service in spite of 15 disks failures.

The parameters of the statistical models for *D-FAD* were fitted using 200 disks, containing a mixture of 17 failed disks and 183 healthy disks. In addition, we used 700 disks to estimate the false positive rate. The disks used for training and those used for testing (evaluating the performance of *D-FAD*) came from the same population of disks. We divided this total population in two sets, maintaining the same proportion of failed and healthy disks in each set. Signals from the disks in the testing set were only used during the evaluation of *D-FAD*. Also, for the purposes of this paper, as long as *D-FAD* sends an alarm between 18 days and 4 hours of the disk failing we will regard the impending failure as detected. By training *D-FAD* on the specific workload that disks experience

in production, the models are not subject to the inherent uncertainty in the failure rates specified by the manufacturer which can present a lot of variability (see [5]), and which were estimated from proprietary stress workloads. Another way to look at the benefits of *D-FAD* is that it reduces the uncertainty of identifying failing disks in the whole population of disks (using the manufacturer failure rate), to a smaller population composed of the of detected failures (15 in our case) plus the false alarms (30 in our case).

In summary, this paper proposes and evaluates *D-FAD* a software based detector of soon-to-fail disks, with the following benefits:

- The input to the detector is a single performance signal, *average maximum latency*; consequently it is efficient at scale.
- It is based on models fitted directly from production workloads; thus, the detector will customize to the particular usage and stress in production.
- The results on real data from 1190 production disks of a 24x7 customer-facing internet service are a 88.2% detection rate and a 2.56% false positive rate.

These benefits are encouraging for further research into building robust detectors based on performance signals from the disks.

The rest of the paper is organized as follows: Section 2 briefly compares this work against the most relevant published literature. Section 3 introduces a set of preliminary definitions. Section 4 describe the models and the methods for training. Section 5 discusses the results of the evaluation and we conclude in Section 6.

2 Related Work

This section is by no means comprehensive. It touches on the published work that is closer to ours and that help to put our work in the right perspective. The seminal paper by Pinheiro, Weber, and Barroso [5] contains an in depth study on a much larger scale than ours. We use the same definition of failure and like them we study disks in operation under stress from real workloads. Our results also confirm their finding that "...we find that failure prediction models based on SMART parameters alone are likely to be severely limited in their prediction accuracy..." Similarly as further reported in that paper we found that a small percentage of failed disks show no SMART error signals (at least amongst the ones we had access to). The important difference between the current paper and [5] is that we focus on prediction based on performance (and not on error signals) with positive results. In [2] and [3] the authors look at machine learning

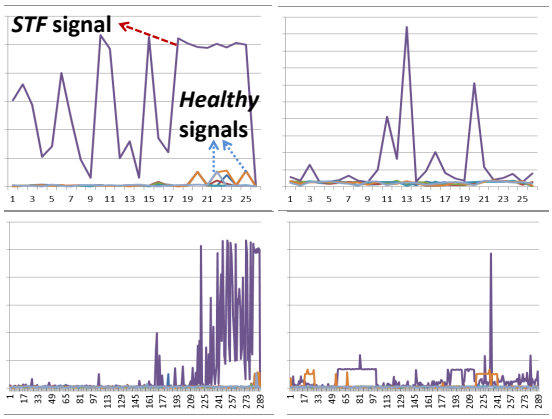

Figure 1: Examples of the *AML* time series for healthy and *STF* disks. Each graph contains 8 time series from neighboring disks. Time series for healthy disks are at the bottom of the *Y* axis. The difference with the values for *STF* disks is sometimes over 7 standard deviations. The top graphs are taken 8 hours before failure. The bottom two are taken 4 days before failure. The *X* axis is in intervals of 20 minutes. Units in the *Y* axis are obscured by request from the operations team at the site.

methods for predicting disk failures using also SMART signals from the disk. Again, this contrasts with us using performance signals. Even though the population of disks in our study is of similar size, our study differs in two important dimensions: (1) the disks in our study are stressed by a real world workload under production conditions and (2) our statistical models are parametric (theirs are non-parametric). This is important as in our models the number of parameters do not increase with the data. Finally, the study in [7] provides understanding on the statistical properties of disk failures but is not focused on detection or prediction.

3 Problem Statement

The problem we are solving is the detection of *soon-to-fail (STF)* disks. We define *disk failure* as in in [5]: *a drive is considered to have failed if it was replaced as part of a repairs procedure*. There are several considerations that go into this definition such as the actual time of failure may not be as accurate as it depends on when the disk was actually exchanged etc. Still, we agree with the authors of [5] that this is the ultimate ground truth for evaluating any predictor/detector. Our proposed solution is a software based detector, *D-FAD*, that uses statistical models to find abnormal behavior. The input to *D-FAD* is a single signal from every disk, a time series containing in each sample the *Average Maximum Latency (AML)* over each sample period, and the output is an alarm if the models in *D-FAD* decide that the signal presents abnormal behavior.

The data collection was done via a software interface provided by the disk vendor. In this interface, all signals are exposed every 20 minutes. The *AML* is computed as follows: the maximum latency of each disk is recorded every minute, then they are averaged over the 20 minutes of the reporting period, and finally exposed by the interface. These are non-overlapping averages. Examples of *AML* for both healthy and *STF* disks are depicted in Figure 1. A list of all the signals exposed by the interface is presented in Table 3. We couldn't find any predictor of *STF* disks other than the *AML* (see Section 5).

Besides being restricted by the parameters set by the vendor, we were constrained by several other factors outside of our control. We only had data for a month for about 2380 disks. We reserved 1190 disks for *training* and 1190 for *testing* the models. All the parameters of the models described in Section 4, model selection, and initial estimation of the various rates, were fitted and computed using the training data. The testing data was only "seen" once during the evaluation of *D-FAD*. The detection and false positive rates reported come from the evaluation (although we remark that they coincide with the ones computed during training). We actual detected disk failures (ground truth) through changes in the serial number of the disk in the logs. The particular disks we study come from a cluster of 144 servers. Each server has 34 drives in an array and drives are 750GB SATA drives. We had data for 70 of those servers. The data was collected during December of 2007 from production disks of a popular 24x7 customer-facing internet service.

With respect to the lead time to failure, that is how soon will the disk fail, we consider that an imminent failure is predicted if the alarm is sent between 18 days and 4 hours before the disk is changed. Note that the definition of false positives is very stringent. If the detector emits an alarm and the disk has not failed within 18 days, (or by the end of the month of data available) then this alarm will be considered as a false positive (even if it is changed within the period established above).

4 The *D-FAD* Models

The schematics for *D-FAD* are shown in Figure 2. The raw *AML* goes through two filters in parallel. The filter on the left compares the (log) odds of the probability that the data is produced by a *STF* disk vs. a healthy disk. It is based on Hidden Markov Models (HMMs) [6] and is described in Subsection 4.1. The other filter counts the number of peaks in *AML* in a 24 hour period (see Subsection 4.2). The output from these two units is taken by a final unit that uses a logistic regression model [8] to decide whether the disk will fail.

These are well understood models, and techniques for fitting the parameters from data and performing inference on new data are well known. The specific combination and application to this particular problem is new.

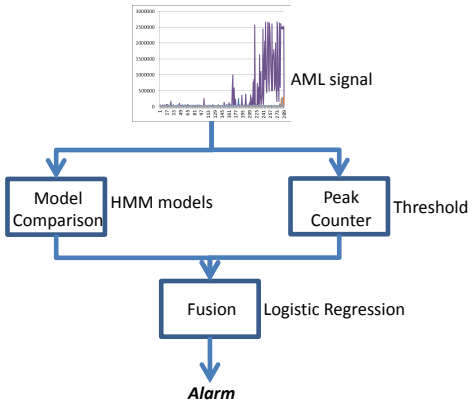

Figure 2: Schematic diagram for *D-FAD*. The *AML* signal goes through a model comparison filter (using HMMs) and a peak counter. The output of these two filters is fused by a logistic regression model.

4.1 The Hidden Markov Model Unit

From inspection of the difference between the *AML* signals of *STF* and healthy disks (see Figure 1), it is clear that at least part of the detector must consist of comparing the statistics of healthy and *STF* disks. We initially tried comparing running averages and using change point detection algorithm with very little success. Both detection and the false positive rate were abysmal. Our next step was to try Hidden Markov Models (HMMs). These models capture several "modes" of operation for the disks, and also take into consideration the time dependence inherent in the signal.

HMMs are statistical state space models [6] used in applications ranging from speech processing to time series prediction. These models assume that the signal is sampled at regular intervals (our case) and therefore time is discrete. An HMM consists of two sets of variables. The first set is the *hidden* part of the model representing a disk *state*. In our case the different states *emit* either low, medium, or high values of the *AML* signal. We will use S_i to denote these variables where the index i represents a particular instance in time in the time series. The second set of variables, denoted by Y_i, is the observable part which is continuous and is used to model the actual values of the *AML* signal. We will use upper case letters for the variables, and lowercase letters for their values. Given a sequence of values $y_1, y_2, \ldots, y_n$ the model assumes that the disk is in one state at each time $s_1, s_2, \ldots, s_n$ and the value y observed at time i is a (probabilistic) function of the state s of the disk. In order to have a complete specification of the model, we need the

probability functions relating the hidden states to the observables, and the transition between the states. Thus we have: (1) $P(S_{i+1}|S_i)$, the probability that the disk is in a particular state $S_{i+1} = s'$ given that it was in state $S_i = s$ in the previous time instance; (2) $P(Y_i|S_i)$ a set of conditional distributions modeling the probability of the values observed given the state of the disk; and (3) a probability for the initial state namely $P(S_1)$. The parameterization of these probability functions in this paper is as follows: $P(Y_i|S_i)$ is considered to be a Gaussian distribution with mean μ_s and variance σ_s. $P(S_i|S_{i-1})$ is considered to be multinomial (as S is a discrete random variable). The final parameter is an integer denoting the number of states $|S|$. These parameters are automatically fitted from samples of the *AML* signal. In this paper we used maximum likelihood estimators using well known algorithms described in [6] (rather than Bayesian techniques [8]).

The problem of fitting the number of states $|S|$ is what is known a *model selection* problem. For this paper we relied on the Bayesian Information Criteria (BIC) [8] which scores the a model based on the goodness of fit to the training data, and the number of parameters (complexity) of the model. Given that we expect to have no more than a handful of states, the problem of model selection proceeds as a linear search: we start with $|S| = 2$ and incrementing by 1 until BIC decreases. In our case $|S| = 3$. Once we fit these parameters the model is completely specified and we can compute the probability of any sequence of observations $y_1, y_2, \ldots, y_n$, using $P(y_1, \ldots, y_n) = \sum_{S_1,\ldots,S_n} P(y_1|S_1)\ldots P(y_n|S_n)P(S_n|S_{n-1})\ldots P(S_1)$. We are now ready to explain the inner workings of the model comparison box in Figure 2. We first train (fit parameters offline) an HMM model using data coming from healthy disks. Let's denote that probability model by P_H. We do the same using data from an *STF* disk and obtain an HMM model for *STF* disks. Let P_{STF} denote that model. Given data D coming from a disk in operation, we compute $P_H(D)$, $P_{STF}(D)$ (using the equation above), and then the ratio $\log(P_{STF}(D)/P_H(D))$ (the log-odds between the models). This ratio, which we denote by *LO* is the output of the model comparison box. If this ratio is bigger than zero then it is more likely that D comes from a *STF* disk. This is one of the two pieces of evidence that the logistic regression unit considers in making a final decision regarding an alarm.

4.2 The Peak Counter Unit

In our initial experiments with the training data, the output *LO* (i.e. the comparison between HMM's) was not enough to decrease the false positive rate while maintaining the detection rate. Thus we introduced an additional filter to provide another piece of evidence. We found that *AML* signals from *STF* disks have peaks that are over 7 standard deviations away from *AML* signals coming from healthy disks (see Figure 1). Running simple statistics we found that a threshold of 2 seconds was over four standard deviations from the signal of healthy disks. This unit returns the number of times that the *AML* signal peaks over 2 seconds in a period of 24 hours. We will use *PC* to denote this count.

4.3 The Logistic Regression Unit

Now we have two signals, the *LO* and the *PC*, and we need to make a decision on whether the disk is *STF* or healthy. We use a logistic regression model for this task. A logistic regression model is the simplest model with the least assumptions about the probabilistic models of these two signals which still helps in making a decision in a principled way. Let $P(A) = P(STF|LO, PC)$ denote the probability that the disk is an *STF* disk. Then the logistic regression computes $\log(\frac{P(A)}{1-P(A)}) = \beta_0 + \beta_1 \times LO + \beta_2 \times PC$. Fitting the parameters, β_0, β_1 and β_2 from data is a standard statistical problem [1]. The final output of *D-FAD* comes from this equation: the more positive $log(\frac{P(A)}{1-P(A)})$ is, the more evidence according to the models for sending an alarm. The threshold T for determining when to send an alarm will be fitted by balancing the trade-off between detection and false positives.

4.4 Parameter Fitting

To fit the parameters of the HMM's, we used 24 hours of data from 60 disks. We selected 12 of (17) failed disks, and 48 healthy disks from the same arrays as the failed disks. This selection from the same array guarantees that we control for common factors (i.e. workload) that may affect the whole array. The period of 24 hours was selected from the point of disk change for the failed disks to avoid daily cyclic confounder factors in the data (the disks do not fail at the same time). Thus P_{STF} was fitted from data coming from 24 hours prior to the failure of 12 disks and P_H was fitted from data coming from 60 healthy disks (in the same period). Given that we will be substituting the parameters of maximum likelihood (and not integrating over them as in the Bayesian methodology) we need to take care that the models are comparable in the number of parameters. This is determined entirely by the number of states $|S|$ which must be equal in both models. In our case the BIC score decreased for $|S| > 3$ in the model for *STF* disks. Thus we set $|S| = 3$. We then used a set of 200 disks (which includes the 60 disks above) to fit the parameters (β_i) of the logistic regression model. Finally, we used an additional 700 disks to continue testing the models and for setting the the threshold T for the decision on an alarm. We increased T until we

Life Ref Time	Reset Ref Time	Perf Collection Interval
Life Sectors Read	Reset Sectors Read	Perf Read Requests
Life Hard Reads	Reset Hard Reads	Perf Write Requests
Life Retry Reads	Reset Retry Reads	Perf Max Queue Depth
Life ECC Reads	Reset ECC Reads	Perf Average Latency
Life Sectors Written	Reset Sectors Written	Perf Maximum Latency
Life Hard Writes	Reset Hard Writes	Perf Maximum Wait Time
Life Retry Writes	Reset Retry Writes	Error Log Total Errors
Life Used Reallocs	Reset Used Reallocs	SMART Reallocated Sectors
Life Timeouts	Reset Timeouts	SMART Pending Reallocs
Life Pred Fail	Reset Pred Fail	SMART ECC Errors

Figure 3: List of signals available to us from each disk.

decreased the rate of false positives without impacting the detection rate.

We performed all these operations only on the training data. The test data was not touched or inspected, so there is no risk of overfitting. We were satisfied with a false positive rate of 2.54% which is similar to the one obtained on the test data later.

5 Results

As specified above, we used half of the population of 2380 disks for training and the other half for testing. The 1190 disks in the testing data were unseen by the algorithms until the evaluation. The evaluation proceeded as follows: *D-FAD* went over the month of data for each disk, taking 24 hours of *AML* data at a time with a sliding window of 4 hours. Thus, every 4 hours, *D-FAD* makes a decision on the disk of whether to classify it as *STF* or healthy. If during the month a disk is classified once as *STF* and it is changed within 4 hours to 18 days from the alarm, then it counts as a predicted failure. We found that 8 disks were predicted with over 10 days of lead time and the rest between 5 days and 8 hours. Otherwise it is counted as a false positive. The results were 88.2% detection (15 out of the 17 changed disks during the month) with 2.56% false positives (30 disks). It is reasuring that these numbers are almost identical to the ones in the training set.

It is interesting to note that we couldn't find in the data anything particularly salient on the two disks that were not detected. Also, we couldn't find any correlation between the *AML* and any of the other signals that were collected from the disks (see Table 3). The workloads (both read and write) were not affected, nor were any other error related indicators. We used several techniques to search for those correlations. First we fitted a logistic regression standard regressions to *AML* as the predicted signal and all the other signals as regressors, using state of the art feature finding techniques such as $L1$ regularization with no success [1, 4]. This lack of correlation may be a result of the level at which the signals were collected, the architecture of the service, or just that more sophisticated analyses are required. Another hypothesis is there are many ways in which the disk may fail and we need a larger population of failed disks to find reliable correlations between the *AML* signal and the different (error) signals behind the different types of failures. Further research is needed but we are comforted by the fact that [5] also reports on the lack of correlation for prediction with SMART error signals.

6 Conclusions

We have described a detector capable of predicting disks that are soon to fail, and we have evaluated the detector on real data from disks used in a $24x7$ customer-facing internet service. The results, 88.2% detection rate and 2.56 false positive rate, are encouraging and provide a significant reduction in the initial state of uncertainty about disks failure.

The decision to deploy *D-FAD* is a trade-off between the cost of false positives and the loss of service quality for not performing preventive actions. Nevertheless the reported results and the advantages of the methodology provide evidence that this is a fruitful avenue of research to pursue.

Acknowledgments

Many thanks to F. McSherry, D. Woodard, N. Reddy, and Q. Ke for comments on previous versions of this paper.

References

[1] HASTIE, T., TIBSHIRANI, R., AND FRIEDMAN, J. *The Elements of Statistical Learning*. Springer, 2009.

[2] HUGHES, G., MURRAY, J., KREUTZ-DELGADO, K., AND ELKAN, C. Hard drive failure prediction using non-parametric statistical methods. *IEEE Transactions on Reliability* (2002).

[3] MURRAY, J., HUGHES, G., AND KREUTZ-DELGADO, K. Machine learning methods for predicting failures in hard drives. *Journal of Machine Learning Research* (2005).

[4] PARK, M. Y., AND HASTIE, T. L1 regularization path algorithm for generalized linear models. *Royal Statistical Society* (2007).

[5] PINHEIRO, E., WEBER, W.-D., AND BARROSO, L. Failure trends in a large disk drive population. In *Proceedings of the 5th USENIX Conference on File and Storage Technologies* (2007).

[6] RABINER, L. B., AND HUANG, H. L. An introduction to hidden Markov models. In *IEEE ASSP Magazine*. 1986.

[7] SCHROEDER, B., AND GIBSON, G. Disk failures in the real world. In *Proceedings of the 5th USENIX Conference on File and Storage Technologies* (2007).

[8] WASSERMAN, L. *All of statistics*. Springer, 2004.