Failure Prediction in Hardware Systems

Doug Turnbull
Department of Computer Science
University of California, San Diego
La Jolla, CA 92093
dturnbul@cs.ucsd.edu

Neil Alldrin
Department of Computer Science
University of California, San Diego
La Jolla, CA 92093
nalldrin@cs.ucsd.edu

Abstract

We analyze hardware sensor data to predict failures in a high-end computer server. Features are extracted using *sensor windows* and *potential failure windows*. We then train radial basis function networks on these features and achieve a 0.87 true positive rate and 0.10 false positive rate for predicting failures using a data set where failures and non-failures are equally likely. This shows that sensor data can be used to predict failures in hardware systems. We conclude by discussing the effects of costs, benefits, and prediction accuracy in the context of infrequent hardware failures.

1 Introduction

A common problem for high-end server systems is that customers demand systems that never fail, but hardware components are inherently prone to failure. This dichotomy has led to server solutions that rely on replication, with hot-swappable system boards and hundreds of processors per system. While these systems rarely "crash", individual hardware components can still fail causing software running on them to fail as well. These failures have high costs for customers who may experience loss of service. They are also expensive for hardware vendors who have to send specialized technicians to repair the problem and have to replace highly specialized components. If a failure can be *predicted*, preventive action can be taken to mitigate the pending failure.

Fortunately, many servers are built with sensors that monitor their hardware and software state. However, it is unclear whether sensor information is useful for failure prediction. Our goal is to show that sensor data *is*, in fact, useful for this purpose. This is achieved by developing a framework that applies machine learning techniques to sensor data in order to predict hardware failures.

Related work has been done for predicting failures in other hardware systems. Gordon Hughes and Joseph Murray analyze failure in hard disk drives [5] [6]. Their general framework is to detect anomalies, or variations from "normal" behavior, using a rank-sum null-hypothesis test. Greg Hamerly and Charles Elkan also examine failures in disk drives [3], but use different statistical tests based on naive Bayesian classifiers. Mike Chen, et. al. analyze runtime paths with statistical inference to detect failures in distributed systems [1]; this is also highly related to their work on Pinpoint [2].

One common problem encountered in much of this research is that failures are infrequent.

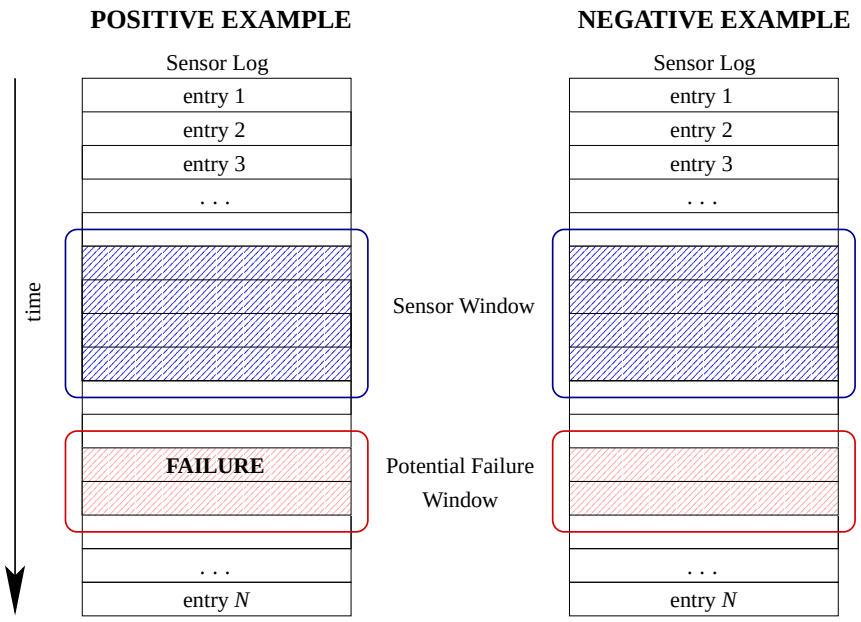

Figure 1: Windowing Technique for Feature Extraction. A positive feature vector is constructed using the sensor data found in the sensor window when a failure occur in the potential failure window. A negative feature vector is constructed from sensor data when no failure is found in the potential failure window.

This can lead to large numbers of mispredictions and complicates collection of failure examples.

2 Failure Prediction

The specific high-end server architecture analyzed for this project contains 18 hot-swappable system boards, each with 4 processors. The system boards are monitored by hardware sensors that record various board temperatures and voltages. Our specific goal is to show that the information captured by these sensors can be used to predict system board failures. Our approach to failure prediction is broken into two stages: feature extraction and classification.

2.1 Feature Extraction

Each system board has an associated sensor log, which records sensor measurements at a rate of about one entry per minute[1]. Failure events are also recorded in this log. From these logs, we create *feature vectors* of sensor information. A feature vector is constructed from the sensor entries in a *sensor window* (see figure 1). Associated with each feature vector is a *potential failure window*. Positive feature vectors are feature vectors where a failure occurs in the potential failure window. Negative feature vectors are feature vectors where the potential failure window does not contain a failure.

There are two kinds of feature vectors: *raw feature vectors* and *summary feature vectors*. The raw feature vector is composed of the values of the sensor log entries in a sensor win-

[1]The sensor sample rate varies between 50 and 60 samples per hour.

dow. Summary feature vectors contain higher level features such as the average, standard deviation, range, and rate of change for each sensor over a sensor window.

2.2 Classification and Subset Selection

Once positive and negative feature vectors are extracted, failure prediction can be framed as a two-class classification problem. A number of machine learning techniques, such as Support Vector Machines and Neural Networks, can be used. We use Radial Basis Function (RBF) networks, a special form of a neural network, because they are computationally efficient during training and classification[7]. Using a subset of the features can further improve classification accuracy. We use *Forward Stepwise Selection*[4] with RBF networks to pick good subsets of features.

2.3 Methodology

Our data set contains sensor and failure information taken from a single server over a five month period. Each board in the system has 18 hardware sensors each of which is sampled at rate of slightly more than one per minute. A raw feature vector contains $18 \times n$ features where n is the number of entries contained in the corresponding sensor window. A summary feature vector contains 72 features, where we include the mean, standard deviation, range, and slope[2] for each of the 18 sensors.

During the five month period, 65 system board failures occurred, all of which are included in our positive example set. We randomly choose 65 negative examples for our negative set of normal operating conditions. We then perform 10-fold cross-validation. Each fold involves choosing a training set of 108 examples and a test set of 12 examples. In both sets, the number of positive and negative examples are equal so that the prior probability of each is 0.5. (Note that we use 120 of the 130 examples so that each fold will have exactly 6 positive and 6 negatives examples.) After training with the training set, we test the performance of the RBF network by finding the percentage of test set examples that are correctly classified.

The goal is to construct an RBF network producing a high *true positive rate* (tpr) and a low *false positive rate* (fpr). The true positive rate is a measure of our ability to identify boards that will fail. The false positive rate is a measure of the boards that we incorrectly predict to fail. More formally, the rates are given by,

$$\begin{aligned} \text{tpr} &= (\text{boards correctly predicted to fail})/(\text{all boards that fail}) \\ \text{fpr} &= (\text{boards incorrectly predicted to fail})/(\text{all boards that do not fail}). \end{aligned}$$

All reported tpr and fpr values represent the average of five runs of ten-fold cross-validation.

3 Error Detection

Table 1 evaluates the performance of summary and raw feature vectors for three window lengths. Summary feature vectors show better false positive rates and similar true positive rates for all three window lengths. In addition, summary feature vectors have lower dimensionality which suggests they should be preferred over raw feature vectors. We do not see any significant trends across window sizes.

We are able to improve the true positive and false positive rates of our classifiers by using a subset of the features. Figure 2 shows rates for both raw and summary feature vectors ex-

[2]The slope is approximated by the last entry minus the first entry in the sensor window.

Features Used	Window Size	tpr	fpr
Raw Features	∼5 Minutes	0.83	0.17
Raw Features	∼10 Minutes	0.87	0.26
Raw Features	∼30 Minutes	0.82	0.24
Summary Features	5 Minutes	0.82	0.15
Summary Features	10 Minutes	0.86	0.22
Summary Features	30 Minutes	0.81	0.16

Table 1: Prediction Performance. Shown is a comparison of raw feature vectors and summary feature vectors for three sensor window sizes. The raw feature vectors contain 4, 8, and 24 sensor log entries for the 5, 10, and 30 minute windows respectively.

tracted from a five minute sensor window. The subsets are selected using forward stepwise selection.

For both raw and summary feature vectors, the tpr increases and the fpr decreases steadily as subset size grows from one to fifteen features. For summary features, both rates peak when the subset includes two-thirds of the features. With 48 features, tpr is 0.87 and fpr is 0.10 which is an improvement from using all 72 features as shown in table 1. There is a drop in performance when we include the last ten percent of the the features suggesting that some of the features actually hurt performance. This phenomenon is not exhibited by the raw feature vectors where the perfomance levels out after we include roughly half of the featues.

The graph also suggests that summary feature vectors yield better performance than raw feature vectors, especially if we are interested in reducing false positives. Here we see a false positive rate that is half that of the raw feature vectors for almost all subset sizes.

4 Discussion and Future Work

What defines a "good" failure prediction system? Three factors are involved: the benefit of correctly predicting a failure, the cost of mispredicting a failure, and the prediction performance. The value of a prediction system can be summarized as

$$\text{value} = (\text{benefit of predicted failure}) \times tpr - (\text{cost of mispredicted failure}) \times fpr.$$

Not predicting a failure, regardless of whether there actually is a failure, incurs no penalty or benefit; it is the same as if no predictor existed at all.

Although our results show that sensor data is useful for predicting failures, our training set assumes that failures and non-failures occur with equal probability. In reality failures are very rare. Since the likelihood of a failure is so small, even a modest false positive rate yields many more false positives than correctly predicted failures. If the predictor is adjusted to reflect this small *prior* probability, then even a seemingly high prediction rate will be low in the adjusted predictor. This is one of the biggest hurdles for failure prediction systems because the cost of many mispredicted failures is typically high compared to the benefit of a few correctly predicted failures. An obvious extension to this project is to explore ways of overcoming this problem.

One way is to is to decrease the false positive rate while increasing the true positive rate. This might be possible by applying other learning techniques such as support vector machines, anomaly detection, and boosting. Classification performance might also be improved by gathering larger data sets containing more failures.

Another way to increase the value of a prediction system is to find ways to decrease the

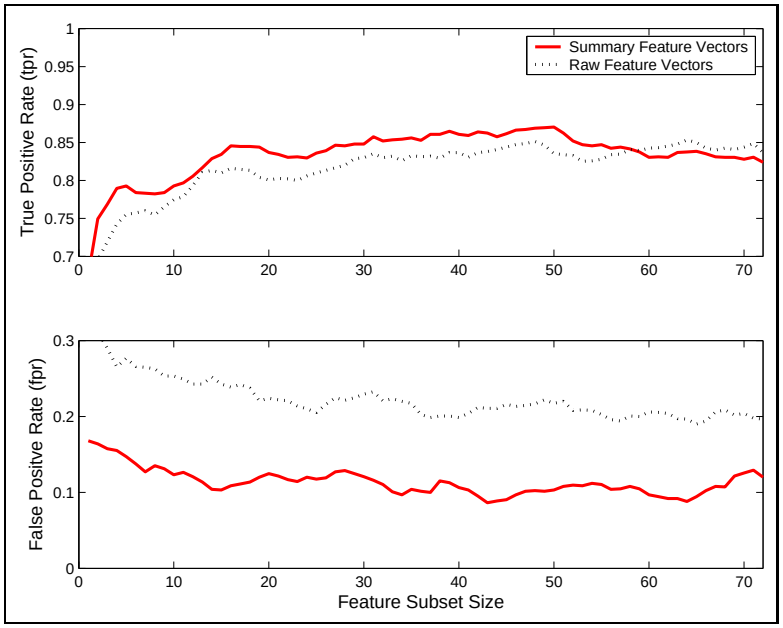

Figure 2: Feature Subset Selection. The top graph shows tpr and the bottom graph shows fpr for both raw and summary feature vectors. A five minute sensor window is used. Note that the two vertical axes have different ranges.

cost of false positives and/or increase the benefit of true positives. This can be done, for example, by using predicted failures as hints to the operating system, which could then automatically move critical jobs to other parts of the system. This increases the reliability of the system for essentially no cost. Some other ways the operating system could take advantage of failure predictions include stopping the system, performing diagnostic tests, lowering processor load, turning off suspect parts of the system, backing up information, and/or migrating processes to other systems. Obviously, the more information the classifier can provide, the more the operating system can do to reduce the cost of a potential failure.

Another direction for our work is to see how well our prediction methods generalize to different types of sensor information. Specifically, we hope to examine logs of the software state, which can be captured at a much finer grain and provide a richer set of information than the hardware variables we currently use.

5 Conclusions

Our failure prediction system achieves a true positive rate of 0.87 and a false positive rate of 0.10 when failures and non-failures are equally likely. This suggests that there is significant information present in hardware sensor data logs for predicting failures. Our windowing abstraction is useful for encapsulating a period of time after which failures may or may not occur. We also find that RBF network classifiers work well both in terms of computational performance and classification accuracy. Classification accuracy is further improved by using feature subset selection.

While our findings show potential, even a false positive rate of 0.10 will be problematic when considering that our data set consists of 65 errors over the course of 4 months. However, if the benefit of correctly identifying a few failures outweighs the cost of misclassi-

fying many failures, given this low failure probability, then our prediction system will be useful.

References

[1] CHEN, M., KICIMAN, E., ACCARDI, A., FOX, A., AND BREWER, E. Using runtime paths for macroanalysis. In *Proceedings of the 9th Workshop on Hot Topic in Operating Systems HotOS 2003* (May 2003).

[2] CHEN, M. Y., KICIMAN, E., FRATKIN, E., FOX, A., AND BREWER, E. Pinpoint: Problem determination in large, dynamic internet services. In *Proceedings of the 2002 International Conference on Dependable Systems and Networks* (2002), IEEE Computer Society, pp. 595–604.

[3] HAMERLY, G., AND ELKAN, C. Bayesian approaches to failure prediction for disk drives. In *Proceedings of the Eighteenth International Conference on Machine Learning* (2001), Morgan Kaufmann Publishers Inc., pp. 202–209.

[4] HASTIE, T., TIBSHIRANI, R., AND FRIEDMAN, J. *The Elements of Statistical Learning*. Springer, 2001. HAS t 01:1 1.Ex.

[5] HUGHES, G., MURRAY, J., KREUTZ-DELGADO, K., AND ELKAN, C. Improved disk-drive failure warnings. In *IEEE Transactions on Reliability, vol. 51, no. 3, September 2002* (September 2002), pp. 350–357.

[6] MURRAY, J., HUGHES, G., AND KREUTZ-DELGADO, K. Hard drive failure prediction using non-parametric statistical methods. In *Proc. ICANN/ICONIP 2003* (June 2003).

[7] TURNBULL, D., AND ELKAN, C. Fast music genre identifi cation using rbf networks. To be submitted., 2003.