Machine Learning for Rocket Propulsion Health Monitoring

Article · October 2005

DOI: 10.4271/2005-01-3370

1 author:

Mark Schwabacher
Google Inc.
45 PUBLICATIONS **1,364** CITATIONS

SEE PROFILE

2005-01-3370

Machine Learning for Rocket Propulsion Health Monitoring

Mark Schwabacher
NASA Ames Research Center

ABSTRACT

This paper describes the initial results of applying two machine-learning-based unsupervised anomaly detection algorithms, Orca and GritBot, to data from two rocket propulsion testbeds. The first testbed uses historical data from the Space Shuttle Main Engine. The second testbed uses data from an experimental rocket engine test stand located at NASA Stennis Space Center. The paper describes four candidate anomalies detected by the two algorithms.

INTRODUCTION

The ability to detect anomalies in sensor data from a complex engineered system such as a spacecraft is important for at least three reasons. First, detecting anomalies in near-real-time during flight can be helpful in making crucial decisions such as the decision of whether to abort the launch of a spacecraft prior to reaching the intended altitude. Second, for a reusable spacecraft such as the Space Shuttle, detecting anomalies in recorded sensor data after a flight can help to determine what maintenance is or is not needed before the next flight. Third, the detection of recurring anomalies in historical data covering a series of flights can produce engineering knowledge that can lead to design improvements.

The current approach to detecting anomalies in spacecraft sensor data is to use large numbers of human experts. Flight controllers watch the data in near-real time during each flight. Engineers study the data after each flight. These experts are aided by limit checks that signal when a particular variable goes outside of a predetermined range. The current approach is very labor intensive. Also, humans may not be able to recognize faults that involve the relationships among large numbers of variables. Further, some potential faults could happen too quickly for humans to detect them and react before they become catastrophic. On future missions to Mars, there will be a speed-of-light delay of up to 20 minutes in between when the data is sensed and when flight controllers on Earth first see it, furthering the need for automated approaches to anomaly detection.

One approach to automating anomaly detection is the model-based approach. This approach encodes human knowledge into a model, which is then used to automatically detect faults. Examples of systems that use the model-based approach include Livingstone [1, 2, 3], Titan [4], TEAMS-RT [5], RODON [6], SHINE [7], and MEXEC [8]. Building the models is very labor intensive; it therefore may not be feasible to model every part of a highly complex system such as a spacecraft. It also may not be possible to model all possible failure modes. We therefore consider supplementing the model-based approach with the data-driven approach.

The data-driven approach seeks to build a model for detecting anomalies directly from the data, rather than building it based on human expertise. In this paper, we explore a particular data-driven approach, which is based on anomaly detection algorithms from the machine learning community.

ANOMALY DETECTION

Anomaly detection algorithms, also known as outlier detection algorithms, seek to find portions of a data set that are somehow different from the rest of the data set. A *supervised anomaly detection algorithm* requires training data consisting of a set of examples of anomalies, and a set of examples of non-anomalous (or *nominal*) data. From the data, the algorithm learns a model that distinguishes between the nominal and the anomalous data points. Supervised anomaly detection algorithms typically require tens or hundreds of labeled examples of anomalies, plus a similar number of labeled examples of nominal data points, in order to obtain adequate performance. *Unsupervised anomaly detection algorithms* are trained using only nominal data. They learn a model of the nominal data, and signal an anomaly when new data fails to match the model. They typically require tens or hundreds of nominal data points in order to obtain adequate performance.

For both of the testbeds that we used, the number of examples of anomalies available to us was fairly small. We therefore decided to use unsupervised anomaly detection algorithms, since they do not require labeled examples of anomalies. The next two subsections describe the two unsupervised anomaly detection algorithms that we used: Orca and GritBot.

ORCA

Orca, which was developed by Bay and Schwabacher, uses a nearest-neighbor approach to anomaly detection. It defines an anomaly to be a point whose nearest neighbors in feature space are far away from it. It uses a novel pruning rule to obtain near-linear-time performance, allowing it to scale to very large datasets [9].

GRITBOT

GritBot is a commercial product from RuleQuest Research [10]. Rather than just looking for points that are anomalous with respect to the entire dataset, GritBot searches for subsets of the dataset in which an anomaly is apparent. For each anomalous point, it reports a description of the relevant subset of the dataset, based on values of discrete variables or ranges of continuous variables, in which the target variable usually has a particular value (if it is discrete) or range of values (if it is continuous). The point is considered to be an anomaly because the target variable at that point is significantly different from the value of the target variable at the vast majority of the other points in the subset.

TESTBEDS

We used two testbeds to test the anomaly detection algorithms: The Space Shuttle Main Engine and a rocket engine test stand at NASA Stennis Space Center.

SPACE SHUTTLE MAIN ENGINE

The Space Shuttle propulsion system consists of two solid rocket boosters, and three Space Shuttle Main Engines (SSMEs) [11]. The SSMEs are located on the Orbiter. During the initial ascent, all five engines are fired together. Approximately two minutes after launch, the solid rocket boosters run out of fuel and are jettisoned, and the Shuttle continues its ascent using only the three SSMEs. The SSMEs are liquid-fueled rocket engines employing cryogenic liquid hydrogen and liquid oxygen as propellants. These propellants are stored in the external tank, which is jettisoned when the Shuttle reaches its intended orbit, about nine minutes after launch.

The SSMEs are reusable. After each flight, they are removed from the Orbiter, inspected, serviced as necessary, and then installed onto an Orbiter (which is not necessarily the same orbiter). Each engine is periodically acceptance tested by firing it on the ground using test stands located at NASA Stennis Space Center.

Each SSME has approximately 90 sensors measuring quantities such as temperature, pressure, fuel flow rate, rotational velocity, and vibration. Many of the sensors are redundant for reliability reasons. For example, four identical pressure sensors may be placed right next to each other with the expectation that they will all provide approximately the same value. When one of the sensors provides a substantially different value from the other three, it can be inferred that the sensor has failed. When the SSMEs are used on the Shuttle, additional relevant information is available from sensors located in the Shuttle's fuel feed system. When the SSMEs are fired on a test stand, the test stand acts as a fuel feed system, and provides additional relevant information from sensors located on the test stand. A small number of additional sensors are placed directly on the SSME during testing. The initial results reported in this paper use only the data from the sensors that are located on the SSME and available both in flight and on the test stand.

ROCKET ENGINE TEST STANDS

NASA Stennis Space Center (SSC) in Mississippi operates several rocket engine test stands. Each test stand provides a structure strong enough to hold a rocket engine in place as it is fired, and a fuel feed system to provide fuel to the engine. Test stands A-1 and A-2 are large test stands used to test the SSMEs. Test stand E-1 is a much smaller test stand used to test experimental rocket engines [12]. Test stand E-1 is being used as part of the Integrated Systems Health Management (ISHM) Testbeds and Prototypes Project as a testbed for a variety of ISHM technologies. It has numerous sensors on its fuel feed system, which are analogous to the sensors on a spacecraft's fuel feed system. In this paper, we present the initial results of applying anomaly detection algorithms to these sensor data.

RESULTS

So far all of the anomalies that we have detected using the two algorithms have been ones that the domain experts either already knew about, or considered to be insignificant. However, some of the anomalies that have been detected demonstrate the potential of the algorithms to detect significant anomalies in the future. In this section, we review several examples of such detected anomalies.

In each case described in this section, we ran an anomaly detection algorithm on data from one rocket engine test. For the SSME, each test had 129 continuous variables, 18 discrete variables, and about 13,000 time steps. For Test Stand E-1, each test had 73 continuous variables, 87 discrete variables, and about 39,000 time steps. GritBot took about 5 minutes to analyze the data from an SSME test, and about 7 minutes to analyze the data from an E-1 test, on a 1.5 GHz Pentium M laptop. Orca took about 3 minutes to analyze the data from an SSME test, and about 12 minutes to analyze the data from an E-1 test, on a 500 MHz Sun Blade 100 workstation. Each SSME test lasted about 8 minutes, so both algorithms were able to process the data faster than real time. Each E-1 test lasted about 2.5 minutes, so both algorithms processed that data somewhat slower than real time. The data from

the E-1 tests had a higher sampling rate than the data from the SSME tests, which prevented the algorithms from being able to process the data in real time. Note that the processors we used were slow by today's standards; we believe that both algorithms could process the E-1 data in real time using faster processors.

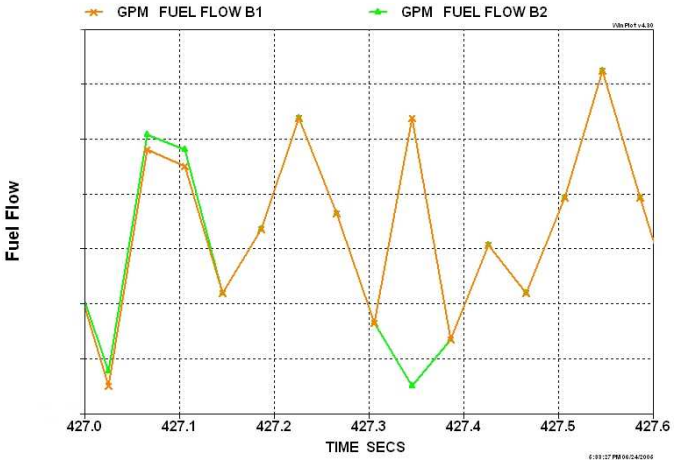

Figure 1: SSME fuel flow anomaly

The first anomaly that we consider is shown in Figure 1. (Note that in all of the figures in this paper, we have removed the numbers from the Y axis in order to protect the confidentiality of the data.) The SSME has redundant fuel flow sensors. Figure 1 shows the values of two of those sensors during a short period of time (0.6 seconds). The values of the two sensors are usually almost exactly the same. However, at approximately 427.35 seconds, they differed substantially. GritBot considered this difference to be an anomaly. GritBot also detected 140 other similar anomalies involving differences between redundant fuel flow sensors; the one in Figure 1 was the strongest anomaly. The domain experts could not explain the difference between the two sensor values, but they noted that this type of small difference occurs fairly regularly, that the small errors occur in both directions, and that the values are averaged before being used to make a decision. They therefore felt that the sensor anomaly was not a reason for concern.

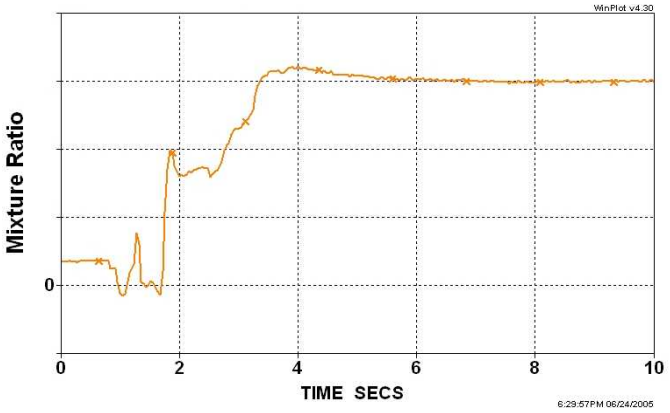

Figure 2: SSME mixture ratio anomaly

The second anomaly we consider is shown in Figure 2. Mixture ratio is the oxidizer to fuel ratio of the SSME's main combustion chamber. By definition, it can not be negative. GritBot observed that this variable was almost never negative in the training data, and then detected several points at which it was negative in the test data, all within the first two seconds of a test. The domain experts said that negative reported mixture ratio at the beginning of a test is a known and well-understood artifact of the way in which mixture ratio is calculated. The SSME does not have an oxidizer flow meter, so the ratio can not be calculated directly. Instead it is calculated indirectly. The indirect calculation produces reasonably accurate values, except during startup.

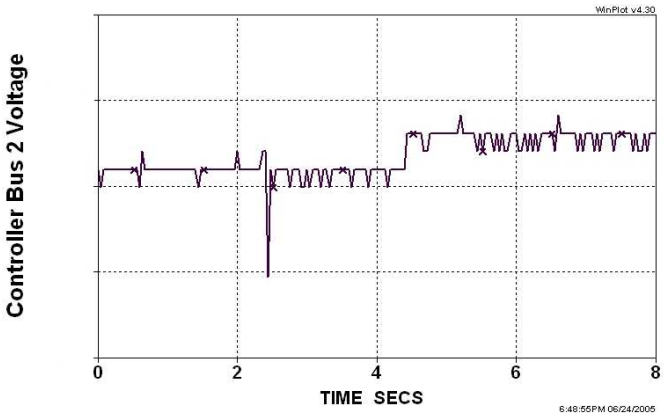

Figure 3: SSME controller bus voltage anomaly

The third anomaly that we consider is shown in Figure 3. The voltage on controller bus 2 usually stays within a fairly narrow range. Orca detected one point in time, approximately 2.5 seconds after the beginning of the test, at which the voltage dropped to an unusually low level. The domain experts said that the main igniters fire at that point in time, which causes the drop in voltage.

They already knew about the drop in voltage, and did not consider it to be an anomaly.

The fourth anomaly that we consider is from Rocket Engine Test Stand E-1. Some of the variables for E-1 are redline enablers – these are discrete control inputs that enable or disable a redline, which is a threshold on a sensor reading beyond which an automatic shutdown is triggered. There are some pairs of redline enablers that tend to be turned on or off at the same time. GritBot noticed that one particular pair of redline enablers had the same on or off value at every timestep in a test except for one timestep: one redline enabler was turned on one timestep later than the other one. GritBot considered this one point where the two redline enablers had opposite Boolean values to be an anomaly. Orca found some similar discrepancies between redline enablers. Our domain experts felt that these very small delays in between enabling one redline and another redline were not significant and should not be considered anomalies.

RELATED WORK

Park, *et al*. applied the BEAM (Beacon-based Exception Analysis for Multi-Missions) system to anomaly detection in SSME data [13]. BEAM has nine components that use nine different approaches to anomaly detection. The work reported in [13] only used one of the nine components: the Dynamical Invariant Anomaly Detector (DIAD). DIAD is an unsupervised anomaly detection algorithm, like GritBot and Orca. DIAD differs from GritBot and Orca in that DIAD only considers one variable at a time, while GritBot and Orca both consider all of the variables together and look for anomalies in the relationships among the variables, in addition to anomalies in individual variables. Park, *et al*. trained DIAD using data from 16 nominal tests, and tested it using data from seven tests that contained known failures. It detected all of the major failures in these seven tests, although it missed some minor failures and had some false alarms.

Iverson's Inductive Monitoring System (IMS) [14] is another unsupervised learning system for fault detection. It uses a clustering algorithm to cluster the nominal training data into clusters representing different modes of the system. When new data fails to fit into any of the clusters, it signals an anomaly, using the distance from the nearest cluster as a measure of the strength of the anomaly. After the STS-107 Space Shuttle Columbia disaster, Iverson applied IMS to some relevant data. He trained it using data from five previous Space Shuttle flights, and then tested it using STS-107 data. It detected an anomaly in data from temperature sensors on the Shuttle's left wing shortly after the foam impact, suggesting in retrospect that with the aid of IMS, flight controllers might have been able to detect the damage to the wing much sooner than they did.

Many of the existing approaches to data-driven systems health monitoring have used artificial neural networks to model the system. Artificial neural networks are a type of model based loosely on the neural structure of the brain, in which the weights of the connections among neurons are automatically adjusted to maximize the fit of the model to the data [15]. Guo and Musgrave [16] applied neural networks to sensor validation for the SSME. He and Shi [17] found that support vector machines produced better accuracy than artificial neural networks when applied to a pump diagnosis problem. One disadvantage of neural network approaches is that most humans are unable to understand or interpret the neural network models. Models based on decision trees, decision rules, or nearest neighbors are generally easier to understand, and therefore more likely to be accepted by human experts.

CONCLUSION

This paper presents four candidate anomalies that were discovered by applying two unsupervised anomaly detection algorithms to two rocket propulsion testbeds. Although none of these candidate anomalies were judged by our domain experts to be significant, we believe that they demonstrate that the algorithms have the potential to detect the kind of unusual phenomena in the data that could correspond to significant anomalies. We have also demonstrated that these algorithms have the potential to process real rocket propulsion sensor data in nearly real time.

An important point to make is that although some anomalies were detected by both Orca and GritBot, other anomalies were only detected by one algorithm or the other algorithm. The reason for the difference is that the two algorithms use two different definitions of an anomaly. Other anomaly detection algorithms besides these two algorithms use other definitions of an anomaly. Because of these differences, it can be useful to run multiple anomaly detection algorithms on a data set.

FUTURE WORK

This paper presents early results; there are many directions for future research. The preliminary results presented in this paper consist of four examples of candidate anomalies that were detected by two unsupervised anomaly detection algorithms. Although interesting, none of these candidate anomalies were judged by the domain experts to be significant. In the future, we would like to test the ability of these algorithms to detect known anomalies in the data, by applying them to several data sets that are known to contain anomalies. Unfortunately, we will probably never have enough data to calculate false positive and false negative error rates with any statistical significance, but testing the algorithms on a larger amount of data than we have used so far should shed additional light on the algorithms' performance.

The results presented in this paper apply the algorithms to data from one propulsion test at a time, flagging

points in time within the test that differ from the rest of the test as anomalies. Another approach is to use a set of past tests that are thought to be nominal as training data, and then test the algorithm on a new test. We have done some preliminary trials in which we trained each algorithm on one past test and then tested it on one newer test. The anomalies detected were essentially the same ones that were detected when running the algorithm on only one test. Further testing of the algorithms across multiple engine tests is needed.

Because of the small number of examples of anomalies available to us, we decided to first try unsupervised anomaly detection algorithms. We believe that the small number of examples of known anomalies will present a challenge for supervised anomaly detection algorithms. A direction for future research is to try to find ways to make supervised learning algorithms perform adequately given such a small number of examples of anomalies. One approach would be to use each time step from an anomaly that spans multiple time steps as an example of an anomaly.

In the results presented in this paper, the anomaly detection algorithms did not make use of any expert knowledge. We plan to explore ways of using background knowledge within the automated anomaly detection context. One possible source of knowledge is the determinations made by the domain experts that the candidate anomalies described in this paper are not significant. One way to make use of this knowledge would be to use semi-supervised learning algorithms, and providing them with examples that are labeled as nominal for each candidate anomaly that the experts judged to be nominal. The algorithm would then avoid incorrectly signaling an anomaly in the future when similar patterns appear in the data.

The data sets used in this research are all time series. The algorithms described in this paper, however, do not explicitly make use of time. Instead, time is treated just like any other variable. We plan to explore algorithms that explicitly model time. One method in particular that we plan to explore is windowing. In the windowing approach, for each time step the algorithm considers the sensor values at that time step and at a fixed number of previous time steps in determining whether or not the point is an anomaly.

We also plan to test anomaly detection algorithms using other systems health management testbeds besides the two rocket-propulsion testbeds described here. In particular, we plan to test the algorithms using data from the International Space Station and from the Apollo missions to the Moon.

ACKNOWLEDGMENTS

This work would not have been possible without the collaboration of our domain experts. For providing SSME data and expertise, we thank Matt Davidson, Al Daumann, and John Stephens of Boeing Rocketdyne. For providing rocket engine test stand data and expertise, we thank Curtis Olive, Randy Holland, Mark Hughes, and Fernando Figueroa of NASA Stennis Space Center.

We thank Matt Davidson, Anna Pryor, and Adrian Agogino for reviewing a draft of this paper.

The research described in this paper was funded by the NASA Exploration Systems Mission Directorate (ESMD) under the Technology Maturation Program as part of the ISHM Testbeds and Prototypes Project, and by ESMD's Advanced Space Technology Program as part of the Collaborative Decision Systems Project.

REFERENCES

1. Williams, Brian C. and P. Pandurang Nayak, "A Model-based Approach to Reactive Self-Configuring Systems," In *Proceedings of the National Conference on Artificial Intelligence,* 1996.
2. Kurien, James and P. Pandurang Nayak, "Back to the Future for Consistency-based Trajectory Tracking," *Proceedings of AAAI-2000 and DX-2000.* 2000.
3. Narasimhan, Sriram, Richard Dearden, and Emmanuel Benazera, "Combining Particle Filters and Consistency-based Approaches for Monitoring and Diagnosis of Stochastic Hybrid Systems," 15th International Workshop on Principles of Diagnosis (DX04), Carcassonne, France, June 2004.
4. Brian C.Williams, Mitch Ingham, Seung Chung, Paul Elliott, and Michael Hofbaur. Model-based Programming of Fault-Aware Systems. *AI Magazine*, Fall 2003.
5. TEAMS-RT Web page. http://www.teamqsi.com/RT.html
6. RODON Web page. http://www.sorman.se/products/rodon/overview.asp
7. R. Colgren, R. Abbott, P. Schaefer, H. Park, R. Mackey, M. James, M. Zak, F. Fisher, S. Chien, T. Johnson, S. Bush, "Technologies for Reliable Autonomous Control (TRAC) of UAVs," 19th Digital Avionics Systems Conference, October 2000.
8. Barrett, Anthony. Model Compilation for Real-Time Planning and Diagnosis with Feedback. *Proceedings on the International Joint Conference on Artificial Intelligence*, 2005. To appear.
9. Bay, S. D. and M. Schwabacher, "Mining Distance-Based Outliers in Near Linear Time with Randomization and a Simple Pruning Rule," *Proceedings of The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 2003.
10. Gritbot. RuleQuest Research Web site. http://www.rulequest.com. 2005.
11. Main Propulsion System. NASA Kennedy Space Center Web page. http://science.ksc.nasa.gov/shuttle/technology/sts-newsref/sts-mps.html. 2005.

12. E-1 Test Facility. NASA Stennis Space Center Web page. https://rockettest.ssc.nasa.gov/ssc_ptd/ssc_e1_test_stand.asp. 2005.
13. Park, H., Mackey, R., James, M., Zak, M., Kynard, M., Sebghati, J., and Greene, W., Analysis of Space Shuttle Main Engine Data Using Beacon-based Exception Analysis for Multi-Missions. Aerospace Conference Proceedings, IEEE, Vol 6, pp 6-2835 - 6-2844, March 9-16, 2002.
14. David L. Iverson. Inductive System Health Monitoring. *Proceedings of the International Conference on Artificial Intelligence*, IC-AI '04, Volume 2 & *Proceedings of the International Conference on Machine Learning; Models, Technologies & Applications*, MLMTA '04, June 21-24, 2004, Las Vegas, Nevada, USA.
15. Bishop, C. M. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
16. Guo, T.-H. & Musgrave, J. Neural network based sensor validation for reusable rocket engines. *Proceedings of the American Control Conference*, 1995, Volume 2, pp. 1367-1372.
17. He, F., & Shi, W. WPT-SVMs Based Approach for Fault Detection of Valves in Reciprocating Pumps. *Proceedings of the American Control Conference*, 2002.

CONTACT

Mark Schwabacher earned his Ph.D. in computer science in 1996 from Rutgers University. His thesis work applied artificial intelligence to engineering design. He has worked at NASA Ames Research Center for seven years. He served as the Software Lead of the NASA X-37 Integrated Vehicle Health Management Experiment, and is currently researching data mining for Integrated Systems Health Management in collaboration with NASA Johnson Space Center, NASA Stennis Space Center, Boeing Rocketdyne, and Northrop Grumman. He has also applied anomaly detection algorithms to Earth science and to aviation security.

Mark Schwabacher
NASA Ames Research Center
MS 269-3
Moffett Field, CA 94035
650-604-4274
mark.a.schwabacher@nasa.gov
http://ti.arc.nasa.gov/people/schwabac/