Case-Based Reasoning for System Anomaly Detection and Management

Sowmya Ramachandran
Stottler Henke Associates, Inc.
1650 S. Amphlett Blvd.
San Mateo, CA 94402
650-931-2700
sowmya@stottlerhenke.com

Christian Belardi
Stottler Henke Associates, Inc.
1650 S. Amphlett Blvd.
San Mateo, CA 94402
650-931-2700
cbelardi@stottlerhenke.com

***Abstract*—Integrated System Health Management (ISHM) technologies are mission-critical for deep space exploration. Space habitats are complex systems, made up of several subsystems such as Life Support, Communications, Data Acquisition, Thermal Control, Environmental Monitoring, and Electrical Power Systems. Deep space missions will face increased latency in communications with Earth. As such, it is critical that deep space aircrafts and habitats be capable of detecting anomalies early enough to mitigate the effects of communication delays. It would also be beneficial if the space crew had the tools to not just detect anomalies but also to help them resolve them without waiting for assistance from ground control. Traditionally, model-based reasoning techniques have been effective in monitoring the health of such operations, but the rising demand for rapid fault detection and response in deep-space habitats calls for autonomous monitoring software that is agile, scalable, and can respond to previously unseen events. Data-driven approaches using machine learning techniques can provide the ability to detect novel anomalies and to evolve over time. However, they are not as effective at generating explanations or tracing root causes as rule-based modeling systems.**

In this paper, we describe a case-based reasoning (CBR) approach to developing ISHM tool that combines the data-driven approach with a method for diagnosing and explaining them. CBR is an artificial intelligence (AI) technique that aims to solve problems by analogy. The system, Anomaly Detection via Topological feature Maps (ADTM), uses self-organizing maps (SOMs) as an unsupervised learning approach for modeling each individual case. ADTM's case-base models represent knowledge about different operational modes, including faults, as individual cases. The design of appropriate indexing mechanisms is crucial to the effectiveness of a case-based model. We describe two different approaches to case-base indexing and retrieval and compare their performance on a data set from a simulation of a CubeSat.

TABLE OF CONTENTS

978-1-7281-7436-5/21/$31.00 ©2021 IEEE

1. INTRODUCTION

Manned deep space travel is becoming increasingly imminent, with multiple organizations working to accelerate the timeline towards this future. NASA is planning a crewed landing on Mars within the next ten years. The Artemis mission, a step along the path to manned missions to Mars, is already underway and has the goal of humans landing on the Moon by 2024 [1]. SpaceX also has outlined an ambitious plan for crewed landing on Mars [2].

Whether the destination is Mars or the Moon, these planned missions provide significant challenges that are currently being addressed. Integrated System Health Management is one such area. Space travel places tremendous risks on human life and solutions are being developed to help manage and mitigate them. Space habitats are made up of a complex web of systems—including but not limited to Life Support, Communications, Data Acquisition, Thermal Control, Environmental Monitoring, and Electrical Power Systems. Astronauts' lives depend on the health of these subsystems. There is a critical need to develop tools to help future astronauts monitor and maintain the health of their habitats, and to handle known and unknown anomalies with limited and access to ground staff.

Model-based reasoning (MBR) is often used to detect faults and diagnose their causes by encoding the schematic information into a model of the subsystem being monitored, which includes the components (including sensors), their normal behavior and known abnormal modes of behavior, and the connections between components. During normal operations, the model is used to simulate the current subsystem behavior and compare the simulated sensor output values to the actual sensor outputs. Significant deviations are used to detect faults. Then the model is used to reason about the components that are faulty and most likely to lead to the currently deviating sensor values. The set of possible faults (possibly including sensor faults) which explain the sensor values is the MBR diagnosis engine's output. The process of using the model to diagnose failures is considered somewhat analogous to the reasoning an engineer uses when using a schematic to try to diagnose the fault. The process can be made more efficient by various heuristics used by spacecraft engineers to quickly diagnose problems and include knowledge of which components are most likely to fail (and

how, e.g., mechanical relays tend to fail open while solid state relays tend to fail closed) and/or be the most likely explanation for certain types of sensor values. Because models encode the effects of contextual factors, they can be applied reliably across contexts. MBR thus requires knowledge engineering efforts to encode these interacting effects. Furthermore, reasoning with complex models can be computationally expensive and this can render them unsuitable for use in ongoing, real-time monitoring of system health.

Data-driven approaches using machine learning techniques overcome the problem of laborious, manual model development and computational inefficiency. Unsupervised approaches like clustering can automatically models patterns of nominal behavior from unlabeled data and monitor real-time telemetry for deviations from these behaviors [3]. Unsupervised learning has been shown to be very effective at detecting anomalies that have never been seen before. However, these approaches cannot effectively trace the root causes for an anomaly for a variety of reasons: 1. Not having the information (i.e. labels) needed to discriminate between faulty and nominal behavior, and 2. Not having any insights into un-instrumented system components. The latter is an important point; not every component of a system will be instrumented to provide telemetry and thus the models will not have visibility into significant portions of the system.

We describe and demonstrate an approach that combines a data-driven behavior modeling with case-based reasoning (CBR) to address these concerns. CBR is a branch of Artificial Intelligence that is concerned with reasoning by analogy with past experiences. It provides a way to leverage data-driven approaches and augment them with additional information to guide troubleshooting and repair/mitigation.

2. Related Work

The focus of this work was on unsupervised anomaly detection for discrete sequences of subsystem data using SOM-based models trained on nominal subsystem behavior. Similar approaches to anomaly detection have been applied in existing research. Principal Component Analysis has been a widely used algorithm for anomaly detection across a wide breadth of applications, including diagnosing offshore wind turbines [4], cyber networks [5], and space telemetry and telecommunications [6, 7]. Furthermore, Gaddam *et. al.* [8] used a supervised approach to anomaly detection by combining *k*-means clustering with ID3 decision tree classification. The classification decisions across the clusters and decision trees were combined for a final decision on class membership. Naik et. al. [9] use supervised a machine learning approach to not just detect but also predict anomalies in spacecraft behavior using telemetry data. The main challenge for such an approach is access to labeled fault data, which can be limited in the space domain.

Iverson *et al.* [10] use *k*-means and density-based clustering techniques for system monitoring in its IMS and ODVEC software systems. Similarly, Gao, Yang, and Xing used a K-Nearest-Neighbor (kNN) approach for anomaly detection of an in-orbit satellite using telemetry data [11]. SOMs have been used for fault detection and diagnosis in several industries. Datta, Mabroidis and Hosek combine SOMs with Quality Thresholding (QT) to refine the resolution of clusters learned by SOMs within the semi-conductor industry [12]. Similarly, Tian, Azarian, and Pecht train a SOM on nominal cooling fan bearing data but use a kNN approach in place of the more traditional Minimum Quantization Error (MQE) to assign test data anomaly scores based on their distance to centroids learned by the kNN model [13]. Cottrell and Gaubert apply anomaly scores to aircraft engine test data using the MQE approach that we have used in this paper and leverage the visualization capabilities of SOMs to visualize the transition states of engines from run-to-failure datasets [14].

ADTM contributes to this existing bed of clustering research by combining a Self-Organizing Map with a Case-Base Reasoning (CBR) approach that assists end-users in responding to detected anomalies. It does so by retrieving records of similar past behavior with pertinent information about the anomaly and, when relevant, past troubleshooting activities.

Such assistance mirrors the role of Mission Control during a failure onboard a spacecraft. In such a situation, teams of scientists and domain experts on the ground help astronauts to quickly respond to a failure to mitigate further risk. They do so by drawing upon years of experience with the systems onboard the spacecraft as well as familiarity with past anomalies, either from test scenarios or real-time failures. ADTM's CBR module aims to mirror such remediation assistance in the context of deep-space exploration, where crew dependency on Mission Control is no longer an option due to significant communication delays.

3. The ADTM approach

ADTM utilizes Self-Organizing Map (SOM) Neural Networks to model system behavior from sensor data in an unsupervised fashion. Also known as a Kohonen map, a SOM is a two-layer artificial neural network (ANN) that uses unsupervised learning to produce a low-dimensional representation of the training samples [15]. The goal of training a SOM is to transform incoming inputs to a 1- or 2-dimensional map in a topologically ordered fashion such that points that are close together in the higher-dimensional input space are close together in the lower-dimensional output space as well. This mapping allows ADTM to detect patterns of normal or anomalous behavior in a system, as different types of behavior map to different output units. At the end of training, each input vector is associated with one or more output neurons, which therein become 'implicit cluster centers' akin to 'centroids' in the k-means algorithm.

A SOM maps *N*-dimensional weight vectors to a (*k x k*) lattice of output neurons O_i that are fully connected to the input layer (see Figure 1). Each neuron O_i is associated with a *N*-dimensional weight vector w_i and represented by a two-dimensional coordinate of its position in the grid, e.g., O_i = (x, y).

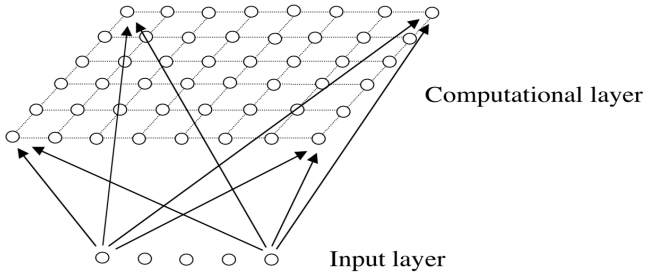

Figure 1: SOM architecture

Like the clustering technique *k*-means, the value of *k* is a parameter that is tuned during model validation. Unlike *k*-means, however, the clusters learned during SOM training are topologically ordered through a competitive learning rule. That is, each input vector $x_i \in X$ is compared with the *N*-dimensional weight vector $\{ w_{1j}, w_{2j}, \ldots, w_{Nj} \}$ associated with each output node O_j. The closest O_j is chosen as the winner, or 'best matching unit' (BMU), where 'close' is defined by some distance function (e.g., Euclidean distance). Each BMU is associated with a neighborhood of related neurons whose weight vectors are also updated, though to a lesser extent proportional to their distance to the BMU in the 2D output lattice. A common choice of 'neighborhood function' h(j,k) that computes the relation between two neurons O_j and O_k is:

$$h(\mathrm{j,k}) = e^{-\frac{D}{2\delta(\mathrm{t})^2}} \quad (1)$$

where $k \neq j$ and D is the lateral distance between the neurons O_j and O_k in the output grid, and δ(t) is the time-dependent exponential decay.

Together, the update rule for the BMU is:

$$\Delta w_{ji} = \alpha(\mathrm{t})\, h(\mathrm{i,j})(x_i - w_{ji}) \quad (2)$$

given

$$\| w_j - x_i \| \leq \| w_k - x_i \| \text{ for all } \mathrm{j} \neq \mathrm{k} \quad (3)$$

where x_i is the input vector and α(t) is the learning rate.

We can think of this learning rule as pulling the weight vector w_j associated with the BMU towards the input vector x_i. All neurons in the same 'neighborhood' are also dragged along, but to a lesser extent.

Once trained on nominal data, the SOM maps new target data to the most similar weight vector of an output neurons using the same process used during the training cycles. The winning neuron is again referred to as the Best Matching Unit (BMU). The difference between the BMU's weight vector and the target point is the Minimum Quantization Error (MQE). We can interpret the set of weight vectors associated with each neuron as a condensed representation of the space of states seen in the training data. Thus, the MQE reflects the SOM's ability to categorize new input data into one of these known states. A low MQE implies that the target sample has characteristics very similar to a sample seen during training. The lower the MQE, the greater the similarity between the target observations and the SOM of comparison. ADTM uses MQE thresholds to determine to detect anomalous system behavior. Points in the telemetry that exceed this threshold are flagged as anomalous. The threshold is tuned using a subset of the data reserved for tuning parameters. The details of this approach are described in [16].

Unsupervised learning approaches have the advantage that they do not need labeled examples, thus eliminating the need to manually label data. Furthermore, these learning approaches can successfully detect previously unknown or unanticipated anomalies. Models learned from a representative sample of nominal behavior can effectively detect deviations or off-nominal behavior [16]. This is beneficial for modeling complex systems like deep-space exploration habitats where unanticipated anomalies can occur and lead to serious life-threatening consequences. However, identifying an anomaly is just the first step; it needs to be followed up by localization and tracing of the anomaly's causes in order to handle the situation [17].

Models developed using unsupervised learning can be utilized for localizing the features most associated with observed anomalies (as discussed in [16]). That still leaves open the problem of tracing the causes of faulty behavior and finding a fix. Our solution generalizes the approach by modeling other known operating regimes, including fault regimes, using the same SOM-based approach above. It is based on the observation that SOM-based modeling does not necessarily have to be limited to the nominal operating regime. It can be used to model any operating regime using data solely representing that regime. Thus, when data from a system operating under a known fault is available, we can build a SOM for that condition and identify when a system is exhibiting that fault using the same SOM-based distance criteria as before. This allows us to detect previously known faults by comparing them to their respective SOMs, as well as novel ones when none of the models for known conditions exactly match the current behavior. For novel anomalies, we can find examples of known faults that are similar, thus providing astronauts with a tool to troubleshoot by comparison to analogous conditions.

This insight has led us to the design of ADTM to use a case-based reasoning (CBR) approach, where a system is modeled as a case-base of SOMs for different known operating regimes. Like supervised learning, this requires labeled examples of multiple conditions. However, there are some key differences:

1. Like supervised learning, the data for one operating regime can serve as negative labeled examples for another, assuming there are no overlaps between the faults in the data provided. However, unlike supervised learning, we do not need all possible operating conditions.

2. This also means that you can still have robust models with limited data. If all you have is data for nominal operations, you can still build a robust anomaly detection and localization model.

3. It also supports incremental modeling, i.e. as new anomalies are discovered, they can be modeled and added incrementally to the case-base without having to retrain the other SOMs.

An added, significant benefit is that cases can include amplifying annotations to help users trace root causes by examining un-instrumented components. The annotations also help preserve institutional memory from previous episodes of similar behavior by including information on tests conducted and steps taken to resolve the problem.

Case-based modeling in ADTM

ADTM uses cases to represent different operational regimes for a system. A particular emphasis is on modeling known faults as cases. The core attribute of a case is a SOM trained on the data representing the corresponding operating condition. The SOM also serves as the index for the case. A case will include other pertinent information such as a description of the operating condition, troubleshooting steps and recommended actions. Each case also contains a label to say if it represents a nominal or a faulty condition.

CBR is an artificial intelligence (AI) technique that aims to solve problems by analogy. With this approach, automated systems solve new problems by retrieving solutions to previous similar problems and altering them appropriately to meet current needs. The field of CBR focuses on developing intelligent and efficient techniques for defining similarity metrics, retrieving cases based on these metrics, and modifying similar solutions to fit the target problem. The underlying inspiration is the analogical reasoning mechanism that humans often use to solve novel problems. While a problem itself may be novel, analogical reasoning helps situate it in the context of similar prior experiences and to discover a new solution by adapting prior solutions [18].

CBR consists of the following basic four steps: 1. Retrieving a set of closest matching cases using some similarity metric, reusing the information in the retrieved cases to solve the problem, revising the prior solution if necessary to solve the current problem, and retaining the parts of the current experience for future problem solving. Defining the mechanisms for indexing and retrieving similar past experiences from the case-base, including developing an appropriate similarity metric, is crucial aspect of this modeling approach.

In an ADTM model, a case represents an operating mode, nominal or faulty, of the system. The retrieve operation will find the most similar operating condition by using the associated SOM as the index. Once found, the system reuses the matching cases by presenting to the user the information associated with the case, including information pertinent to diagnosing and fixing the problems. Sometimes a matched case will be similar enough to the target problem that its solution can be reused without modifications. In other cases, the information from the reference case will help the users develop a revised solution on their own. The revised solution, along with a new SOM trained on relevant data, becomes a new case for the model. This is the retain phase of CBR. While human effort is still required to understand and solve truly novel situations, access to similar prior reference situations provided by the CBR approach will assist users in applying their analogical problem-solving skills to find a solution more effectively. This new knowledge then becomes a part of system memory and can be reused in the future using the retrieve mechanism. Thus, CBR can support active learning.

Adding a new case involves training a new SOM for the sensor data covering the duration of the episode and adding supporting details about diagnosis and mitigation. In case the new episode is marked as a variation of an existing case, the system will merge the two by retraining the associated SOM with the new data and updating the supporting information. Thus, case-based reasoning enables an easy extension of the model based on new observations. The initial case base for a system will consist of a small set of cases representing nominal operations as well as known anomalies. This will grow over time as experiences build up. The ADTM system is designed to include a user-facing tool to facilitate creating and updating the system model.

CBR retrieval using SOMs

Our initial implementation used SOMs as the similarity index, using the following algorithm:

1. Calculate the MQE for all the data points in the target telemetry stream against the index SOMs for each of the cases in the case base.

2. Classify each data point as Anomaly/Not Anomaly by comparing its MQE relative to the index SOM against a tuned threshold for the case [16].

3. The similarity of the data stream to a SOM is the number of points in the data set classified as NOT ANOMALOUS with regard to that SOM.

Section 4 reports the results of using CBR using this indexing scheme to identify faults from new telemetry.

The ultimate purpose of ADTM modeling is to analyze real-time sensor telemetry generated within a spacecraft or habitat to detect anomalies as soon as they occur and alert users. ADTM's case-base model allows for rapid detection of anomalies. However, a case-base is likely to have a large number of cases. Comparing incoming telemetry data in real-time to the entire case-base for every time segment of interest could be prohibitively expensive computationally. On the

other hand, it is important to detect signs of anomalies at the earliest to give the crew the most amount of time to respond.

We have developed and tested a technique for real-time monitoring of real-time data that balances these two competing criteria. The algorithm works as follows:

1. *Batch_Size = k (Batch_Size* is the number of data points from the incoming telemetry that will be analyzed as a batch. *k* >=1)

2. *Current_Model*=Nominal (If there is more than one nominal operating mode represented in the case-base, any one of these may be selected)

3. For every new batch of data of size *k*

3.1 Measure model drift relative to *Current_Model*
Model_Drift = Number of points flagged as anomalies using MQEs derived from *Current_Model*

3.2 If *Model_Drift>Threshold*Batch_Size (Threshold is a configurable parameter)*
New_Model = Result of performing CBR retrieval to get a new model to match the current batch of data

3.3 *Current_Model=New_Model*

The algorithm selects a model that is most similar to the most recent batch of data that was analyzed. This model is retained for subsequent batches, unless a significant drift is detected. Drift is measured by the number of anomalous points in the current data batch relative to the current model. If this drift is higher than a threshold, the ADTM analyzer is triggered to perform a full search with the case-base to find a case that is a closer match. This is retained as the current model going forward. This process is repeated as long as there is new telemetry coming in.

This method addresses two issues: the computational cost of searching the case-base and controlling for noisy behavior. As stated earlier, comparing a data point to a SOM model is an expensive operation. For example, for a SOM with 100 nodes, comparing a datapoint against a SOM requires 100 vector operations. Doing a full case search for a datapoint multiplies this number by the total number of cases. Maintaining a current model and comparing incoming data points reduces number of full case searches, assuming that the system behavior data is not changing too rapidly. However, this advantage may be lost if the data being analyzed is noisy. The *Batch_Size* and *Threshold* parameters control for noisy behavior and allow ADTM to perform a full case-retrieval only when there is sufficient evidence for a change in operating regime. However, a large *Batch_Size* increases the computational cost of comparisons to *Current_Model* (which is itself a SOM comparison) and can also lead to over-smoothing, where genuine behavior transitions are missed. In our experiments, we have found a Batch_Size of 10 and a Threshold of 0.1 substantially reduces the number of full case retrievals while maintaining high accuracy of case identification. However, the choice of these parameters will depend on the system being modeled and must be tuned for each system.

Using Signature Vectors for faster case retrievals

To reduce the data analysis time, we have investigated other approaches to efficient matching and retrieval of cases. One is to use a signature vector as an alternative Case-Base Reasoning (CBR) indexing function. Our preliminary experiments (Section 4) show comparable performance to the SOM indexing function we have been using. Performing CBR look ups with a signature vector reduces computational and memory complexity of the indexing operation. Using a SOM as a case-index requires *N* vector operations for to compare a single data point to a single case, where N is the number of nodes in the corresponding SOM. Using a signature vector, on the other hand, requires only a single vector operation, leading a reduction in computational cost by a factor of *N*. Even for small SOM sizes such as a network with 10 nodes, using a signature vector can lead to a ten-fold increase in computational efficiency for case retrievals. We are experimenting with a few variations of the technique, but the basic principle is the same. We train a SOM to model all nominal behaviors. Then to classify new data, we calculate the feature deviation of the new data from the SOM's *k* BMUs. This calculation gives us the contributions of each feature to the MQE, essentially indicating which features are most different from and most similar to known nominal behavior represented in the SOM. We construct signature vectors by concatenating feature deviation with the average weight of the *k* BMUs. The motivation behind this construction is that because the feature deviation is created with respect to the weights, the deviations themselves might be ambiguous without the context from which they were created. More intuitively, the weights of the SOM capture different nominal behaviors, it is important to know what nominal modes/conditions we are deviating from to understand what is going wrong.

In the approach described previously, ADTM's case-base is constructed by training a SOM for each operating condition using associated training data. For the signature vector approach, a SOM is created the nominal operating condition (one condition is chosen randomly if there are multiple nominal operating conditions represented in the data). The case for each operating condition is then constructed by deriving a case *signature vector* using the training data for that condition. A case signature vector is simply the average of the signature vectors for all data associated with that case. We also keep track of the maximum distance a signature vector associated with the case is from the case signature vector. Thus, under this approach, the case-base is a collection of cases, one for each distinct operating condition in the training data, where each case is indexed by its case

signature vector that represents the most similar reference behavior in the SOM and the cases deviation from it. All the other aspects of a case, i.e. additional annotations, are maintained as before.

To retrieve a case for a new data point, the algorithm computes the signature vector relative to the reference SOM and compares it to the case-signature vectors for each case. The vector distance is the similarity metric used to retrieve the closest matches. Note that if the distance to a case signature vector is greater than the maximum distance vector for the case, the case is excluded from the list of matches. Thus, it is possible to return no matching catches, indicating that the new data represents a novel condition. To label a batch of data, the algorithm labels each data point as described and assigns the most commonly occurring label to the entire batch.

4. ADTM PERFORMANCE STUDIES

We have tested ADTM's approach on data from a range of systems, both simulated and real. We compare the performance of the CBR approach using the two indexing mechanisms described above. Note that these experiments did not utilize the real-time analysis algorithm described earlier and instead treated the tested data for each condition as a single batch.

Data Description

We validated the performance of ADTM's case-based analysis on data from a simulation of a CubeSat (named "LabSat"), originally designed after the Ionospheric-Thermospheric Scanning Photometer for Ion-Neutral Studies (IT SPINS) project to study the nocturnal ionosphere. We used this dataset because it had the greatest number of operating conditions and fault conditions represented. The LabSat was subdivided into three circuit boards. Board 1 was designated for power generation and storage (solar array simulators and batteries), while Boards 2 and 3 had redundant regulators and loads consuming power from Board 1. The system is connected to software that enabled us to insert a variety of hardware faults. We collected nominal data from a 15-minute run of the LabSat across all three boards, with telemetry collected once per second. We also generated data for a total of *5* anomalous conditions across the three boards. Note that a single fault condition generated data for all three boards. However, the fault often introduced anomalies in the measurements of only a single board, while remaining nominal for the rest of the boards. Therefore, our data set consisted of a total of 18 different operating conditions, of which 8 represent faulty conditions. Furthermore, of the 8 anomalous cases, only 6 could be considered similar to each other in that one could be considered a progression in severity of another. The other 2 anomalous cases were of novel and not similar to any of the other anomalies.

Experimental Procedure

Each data set was split into a training set and a test set (using 2/3-1/3 split for training and testing respectively). The case-bases were created as described in the previous sections using the training data sets. We created a case-base for each board. This resulted in a case base with six cases for each board. Once the case base was built, the test set for each of the six conditions was compared to the case base to find the closest match using the similarity metrics described above.

The preprocessing step consisted of standardizing the data using decimal scaling. The nominal training set was scaled to lie in the range -1.0 to +1.0. Data for all other conditions were scaled relative the nominal training data. There were no instances of missing values in the entire data set.

We considered two cases:

1. When the test data is an instance of some condition that has been seen before. In this situation, it is desirable CBR include the prior case in its list of retrieved cases.

2. When the test data represents an instance of some condition never seen before. In this situation, it is desirable that the CBR include a prior case that is similar to the new data. In both situations, it is desirable that CBR excludes dissimilar cases from its list of matches.

The effectiveness of the CBR retrieval can be measured by whether the list of similar cases retrieved contains an exact match from past experience when it exists, whether it contains other known similar cases, and whether it excludes cases that are known to be dissimilar.

Results and Discussion

We performed case-base retrieval using two indexing mechanisms: 1. SOM-based indexing, 2. Signature-based indexing.

For the first of the two cases, namely where the new observed data is an instance of an operating regime encountered and modeled before, ADTM's performance is summarized in Table 1. Recall that in this case, every new set of observations should produce an exact match with the case-base.

This indicates that ADTM's CBR approach using SOMs for indexing was successful at retrieving a case that matches the current behavior exactly. There was only one condition for which it was not able to retrieve an exact match. Even in that case, it did retrieve a case that was similar is cause and effect but differed only in severity. Furthermore, in our experiments, in addition to retrieving an exact match, ADTM also retrieved other instances of prior situations that can be considered similar. This provides additional information to users as they try to resolve observed anomalies. Finally, the

approach was successful in excluding dissimilar cases, leading to higher accuracy of information provided.

Table 1: Performance of ADTM when prior similar cases exist in the case-base

Measure	SOM-based Indexing	Signature-Based Indexing
Number of times an exact match was retrieved	17/18 (94.4%)	16/18 (88.8%)
Number of times a similar match was retrieved	18/18 (100%)	18/18 (100%)
Number of times an incorrect match was retrieved	0/18 (0%)	0/18 (0%)

Using signature-based indexing produced comparable results. While it did not retrieve the exact match for all cases, it did find similar matches for all. However, the benefits of this approach in terms of significantly reduced computational costs of case retrieval overcome the marginal loss of performance.

Situation 2: These are the results for the case where the new data represents a condition that has NOT been seen before (Table 2). Here we focus on the 8 fault conditions.

Table 2: Performance of ADTM when no prior similar cases exist in the case-base

Measure	SOM-based Indexing	Signature-Based Indexing
Number of times a similar match was retrieved, when such a match existed	5/6 (83.3%)	5/6 (83.3%)
Number of times an incorrect match was retrieved	1/8 (12.5%)	2/8 (25%)

Note that 2 of the 8 faults in the dataset represent novels conditions not similar to any other. Using SOMs for indexing, CBR did not return any matches for these faults. For 5 of the remaining 6 faults, CBR successfully retrieved prior similar cases. The signature-based indexing approach produced more incorrect matches than SOM-based indexing. The causes for this will be investigated further and the statistical significance of the differences in performance between these approaches will be studied through with more experiments.

Our preliminary experiments demonstrate that the CBR approach can successfully identify and label novel anomalies that have either been previously encountered or are somewhat similar to a past anomaly. For faults that are not similar to anything observed in the past, the SOM-based approach correctly identifies the occurrence of the anomaly and identifies it as a novel condition. The signature-based retrieval mechanism does sometimes fail to recognize a novel anomaly. It labeled one out of the two novel faults as nominal. The reasons for this will be investigated further the approach will be further refined.

6. Conclusions And Future Work

Preliminary experiments demonstrate the effectiveness of using case-based reasoning with SOM-based modeling approaches to detect anomalies and to find prior similar occurrences. Our experiments demonstrate that ADTM is effective accurately retrieving prior cases that have some similarity but are not exactly identical to the current fault. However, some novel anomalies may not match with any previously seen cases. In this case, ADTM can still flag the anomaly and alert users. Furthermore, ADTM can add the new anomaly to its case base, along with user-provided annotations to help future such incidents. We also demonstrated an innovative approach to case base indexing that significantly reduces the computational cost of case retrievals. While successful in most cases, the signature-vector base approach demonstrated a need for further investigation and refinement.

Because we were able to inject known hardware faults into the LabSat, the data served as a testbed for validating the accuracy of ADTM's anomaly detection and localization techniques. Going forward, it will be important to validate ADTM on real system generated data. We will also experiment with larger samples of data from different operating conditions to test the robustness and generalizability of ADTM.

We are currently investigating a hierarchical modeling architecture to model and analyze systems at different levels (i.e. system, subsystem, component). This type of modeling would help identify and explain faults that span multiple subsystems. The overall objective is to use these techniques combined with other predictive modeling approaches to provide NASA with a comprehensive system health maintenance tool.

Additionally, we plan to integrate with the approach of using assessments of human physiology as sensors to detect habitat anomalies [23].

Acknowledgements

This work was performed under a contract awarded and administered by National Aeronautics and Space Administration Agency (NASA). We would like to thank Dr. Rodney Martin and Dr. Craig Moore of NASA for their ongoing feedback on this effort.

REFERENCES

[1] NASA. "NASA: Artemis." Accessed October 13, 2020. https://www.nasa.gov/specials/artemis/index.html.

[2] SpaceX. "SpaceX." Accessed October 13, 2020. http://www.spacex.com.

[3] Hastie, T., Friedman, J., & Tisbshirani, R. (2017). The Elements of statistical learning: Data mining, inference, and prediction. New York: Springer.

[4] Bennouna O, Heraud N, Leonowicz Z (2012) Condition monitoring & fault diagnosis system for Offshore Wind Turbines. 2012 11th International Conference on Environment and Electrical Engineering. doi: 10.1109/eeeic.2012.6221389

[5] Pascoal C, de Oliveira M, Valadas R et al. (2012) Robust feature selection and robust PCA for internet traffic anomaly detection. 2012 Proceedings IEEE INFOCOM. doi: 10.1109/infcom.2012.6195548

[6] Nassar B, Hussein W, Mokhtar M (2019) Space Telemetry Anomaly Detection Based on Statistical PCA Algorithm. In: Zenodo. http://doi.org/10.5281/zenodo.1109667.

[7] Mukai, R., Towfic, Z., Danos, M., Shihabi, M., & Bell, D. (2020). MSL Telecom Automated Anomaly Detection. *2020 IEEE Aerospace Conference*, 1–6. https://doi.org/10.1109/AERO47225.2020.9172573

[8] Gaddam S, Phoha V, Balagani K (2007) K-Means+ID3: A Novel Method for Supervised Anomaly Detection by Cascading K-Means Clustering and ID3 Decision Tree Learning Methods. IEEE Transactions on Knowledge and Data Engineering 19:345-354. doi: 10.1109/tkde.2007.44

[9] Naik, K., Holmgren, A., & Kenworthy, J. (2020). Using Machine Learning to Automatically Detect Anomalous Spacecraft Behavior from Telemetry Data. *2020 IEEE Aerospace Conference*, 1–14. https://doi.org/10.1109/AERO47225.2020.9172726

[10] Iverson D, Martin R, Schwabacher M et al. (2009) General Purpose Data-Driven System Monitoring for Space Operations. AIAA Infotech@Aerospace Conference. doi: 10.2514/6.2009-1909

[11] Gao Y, Yang T, Xu M, Xing N (2012) An Unsupervised Anomaly Detection Approach for Spacecraft Based on Normal Behavior Clustering. 2012 Fifth International Conference on Intelligent Computation Technology and Automation. doi: 10.1109/icicta.2012.126

[12] Datta A, Mavroidis C, Hosek M (2007) A Role of Unsupervised Clustering for Intelligent Fault Diagnosis. Volume 9: Mechanical Systems and Control, Parts A, B, and C. doi: 10.1115/imece2007-43492

[13] Tian J, Azarian M, Pecht M (2014) Anomaly Detection Using Self-Organizing Maps-Based K-Nearest Neighbor Algorithm. European Conference of the Prognostics and Health Management Society 5

[14] Cottrell M, Gaubert P, Eloy C et al. (2009) Fault Prediction in Aircraft Engines Using Self-Organizing Maps. Advances in Self-Organizing Maps 37-44. doi: 10.1007/978-3-642-02397-2_5

[15] Kohonen T (1982) Self-organized formation of topologically correct feature maps. Biological Cybernetics 43:59-69. doi: 10.1007/bf00337288

[16] Ramachandran, S., M. Rosengarten & C. Belardi (2020) Semi-Supervised Machine Learning for Spacecraft Anomaly Detection & Diagnosis. Proceedings of IEEE Aerospace Conference 2020. Big Sky, MT, March 8-13, 2020.

[17] Farrar, C. R., & Worden, K. (2012). *Structural Health Monitoring: A Machine Learning Perspective* (1st Edition). Wiley.

[18] Carbonell, J. (1985). *Derivational analogy: A theory of reconstructive problem solving and expertise acquisition.* (No. CMU-CS-85-115). Carnegie-Mellon Univ Pittsburgh PA Dept of Computer Science.

[19] Fytilis, N., & Rizzo, D. M. (2013). Coupling self-organizing maps with a Naïve Bayesian classifier: Stream classification studies using multiple assessment data: A Combined Naïve Bayesian-SOM Classification Network. *Water Resources Research*, *49*(11), 7747-7762.

[20] Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. Machine Learning 46:389-422. doi: 10.1023/a:1012487302797

[21] Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. Machine Learning 63:3-42. doi:10.1007/s10994-006-6226-1

[22] Breiman L (2001) Machine Learning 45:5-32. doi: 10.1023/a:1010933404324

[23] Brown, S., Prasad, A., & Fink, W. (2020). Inventory of Vital Sign Changes as Indicators of Environmental Changes aboard Space Habitats. IEEE Aerospace Conference Proceedings.

BIOGRAPHY

Dr. Sowmya Ramachandran *is a research scientist at Stottler Henke Associates where her research focuses on the application of artificial intelligence (AI) and machine learning to improve human performance in a broad range of domains such as intelligence analysis and tactical decision-making. This includes research on intelligent performance support systems and advanced training technologies. She led the development of an after-action review tool for large team training exercises that used machine learning techniques to automatically group chat stream messages into coherent topics. She is currently the PI on efforts to automate performance assessment in simulation-based training systems by generalizing from expert demonstrations of solutions using machine learning approaches. Dr. Ramachandran has led the development of Intelligent Tutoring Systems for a diverse set of domains, from training paramedics to training Information Systems Technicians in the U.S. Navy. She combines a rigorous background in artificial intelligence with deep knowledge of the science of human performance and training, both for individuals and teams. Her doctoral dissertation developed a novel approach to learning Bayesian Networks using a neural network training approach. The objective was to leverage the strengths of neural networks while endowing them with a semantic interpretation that is comprehensible to human experts. Dr. Ramachandran holds a Ph.D. in Computer Science from The University of Texas at Austin. She earned a Bachelor of Technology degree from the Indian Institute of Technology, Chennai, India.*

Christian Belardi *earned a Bachelor of Science in Computer Science from Cornell University's College of Engineering. At Stottler Henke he has contributed to a variety of applied reach efforts in artificial intelligence and machine learning. In addition to ongoing work on self-organizing maps and case-base reasoning, he works on self-supervised representation learning with neural networks as the lead engineer on a DARPA funded research effort. The primary purpose of this project is to develop machine learning and data fusion techniques to assess the pathogenic potential of novel bacteria. In previous work, he has developed intelligent software for a variety of NASA and DoD needs.*