Multiple Kernel Learning for Heterogeneous Anomaly Detection: Algorithm and Aviation Safety Case Study*

Santanu Das
UARC, UCSC
NASA Ames Research Center
Moffett Field, CA 94035
Santanu.Das-1@nasa.gov

Bryan L. Matthews
SGT Inc.
NASA Ames Research Center
Moffett Field, CA 94035
Bryan.L.Matthews@nasa.gov

Ashok N. Srivastava
NASA Ames Research Center
Moffett Field, CA 94035
Ashok.Srivastava@nasa.gov

Nikunj C. Oza
NASA Ames Research Center
Moffett Field, CA 94035
Nikunj.C.Oza@nasa.gov

ABSTRACT

The world-wide aviation system is one of the most complex dynamical systems ever developed and is generating data at an extremely rapid rate. Most modern commercial aircraft record several hundred flight parameters including information from the guidance, navigation, and control systems, the avionics and propulsion systems, and the pilot inputs into the aircraft. These parameters may be continuous measurements or binary or categorical measurements recorded in one second intervals for the duration of the flight. Currently, most approaches to aviation safety are reactive, meaning that they are designed to react to an aviation safety incident or accident. In this paper, we discuss a novel approach based on the theory of multiple kernel learning to detect potential safety anomalies in very large data bases of discrete and continuous data from world-wide operations of commercial fleets. We pose a general anomaly detection problem which includes both discrete and continuous data streams, where we assume that the discrete streams have a causal influence on the continuous streams. We also assume that atypical sequence of events in the discrete streams can lead to off-nominal system performance. We discuss the application domain, novel algorithms, and also discuss results on real-world data sets. Our algorithm uncovers operationally significant events in high dimensional data streams in the aviation industry which are not detectable using state of the art methods.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Reliability, availability, serviceability; H.2.4 [**Systems**]: Data Translation; H.2.5 [**Heterogeneous Databases**]: Discrete databases; H.4.2 [**Types of Systems**]: Decision Support

General Terms

Systems health management, measurement, human factors, data analysis

Keywords

aeronautics, anomaly detection, prediction, prognostics

*A full version of this paper is available as at `https://dashlink.arc.nasa.gov/topic/multiple-kernel-learning-based-heterogeneous-algorithm/`

1. INTRODUCTION

On January 31, 2000, a McDonnell Douglas MD-83 was enroute from Puerto Vallarta, Mexico to Seattle Washington when it experienced a catastrophic failure resulting in the death of 89 passengers and flight personnel as it dived from about 18000 feet into the Pacific Ocean. Analysis of data from the Flight Data Recorder (FDR) and the wreckage indicated that the probable cause of the accident was "a loss of airplane pitch control resulting from the in-flight failure of the horizontal stabilizer trim system jackscrew assembly's acme nut threads. The thread failure was caused by excessive wear resulting from Alaska Airlines' insufficient lubrication of the jackscrew assembly." [3] The precursors to this accident and other accidents due to mechanical issues and human factors are often evident in large data sets from Flight Data Recorders as well as textual reports written by the flight crew and other persons involved in flight operations. Indeed, an informal study at NASA of FDR data from similar aircraft showed that a multivariate query on certain parameters in the FDR could uncover aircraft with similar mechanical problems.

Boeing recently completed a comprehensive statistical survey of commercial aircraft accidents worldwide [18] which shows a dramatic drop in the accident rate, the fatal accident rate, and also the hull-loss accident rate as shown in Figure 1. These advances have been due to a significant investment in near-term technologies to improve aircraft safety. However, assuming that air traffic continues to grow at a modest rate of only 3% per year, over the next two decades that will lead to nearly doubling the air traffic within the US. The increased density of operations, com-

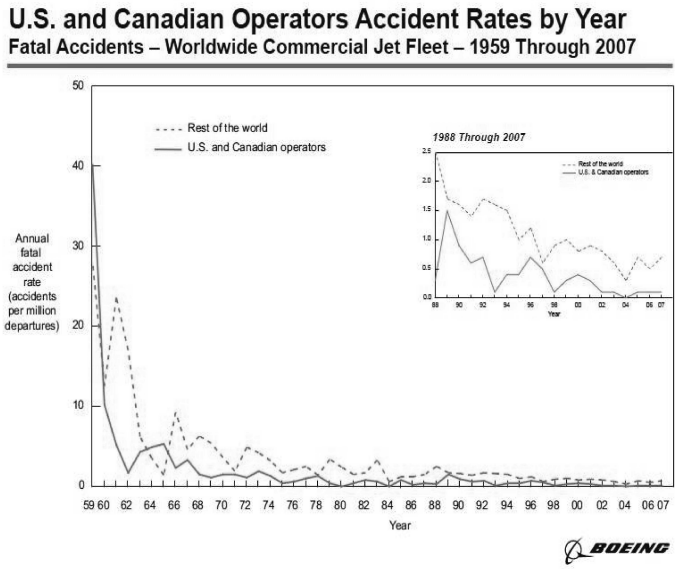

Figure 1: This figure from Boeing's Statistical Summary of Commercial Jet Accidents in 2007 shows that the fatality rate of accidents in the United States has dropped significantly over the past five decades. However, as air traffic continues to grow, it is essential to develop techniques to reveal precursors to safety incidents and accidents from large flight data bases.

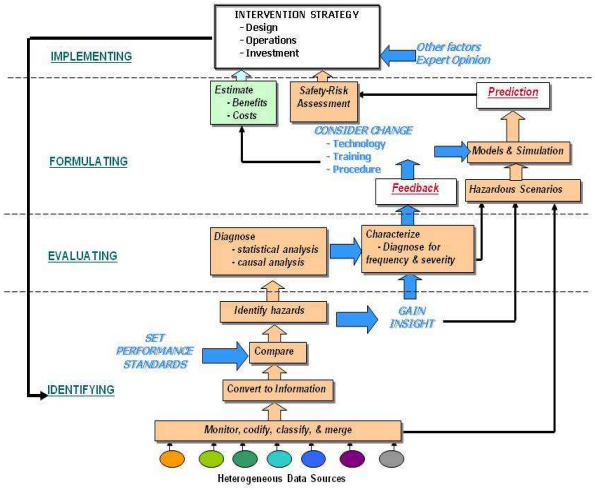

Figure 2: This figure shows our overall approach to addressing safety issues in the aviation system. The automatic identification and causal analysis of hazards from continuous and discrete data streams is the subject of this paper [**?**].

bined with increased demands on service and availability of aircraft can lead to a significant number of aviation accidents even with the current extremely low accident rate.

In 2007, NASA established an archive that contains data from flight data recorders from most of the major carriers in the US. The archive, known as the Distributed National FOQA (Flight Operations Quality Assurance) Archive (DNFA) contains over two million flights today and covers over 10 major carriers. Typical FOQA parameters consist of both continuous and discrete (categorical) data from the avionics, propulsion system, control surfaces, landing gear, the cockpit switch positions, and other critical systems. These data sets can have up to 500 parameters and are sampled at 1 Hz. For a moderate sized fleet that operates 1000 flights per day, these FOQA data sets become very large. Due to proprietary and legal issues, these data are not shared between airlines or directly with the government. Thus, in the DNFA architecture, each carrier has its own data repository, and a primary requirement of the DNFA is that the data from multiple carriers not be centralized. Data in the DNFA is anonymized so that flight numbers and marks identifying pilots and crew are removed. In this paper, we discuss the problem of detecting anomalies in FOQA data that may be indicative of safety issues due to mechanical or human factors issues. We present the results of our anomaly detection algorithms on real FOQA data from a regional carrier. This work is part of a comprehensive plan within the NASA Integrated ViIVHM project to analyze numerical data from DNFA and text reports from DNAA to assess health of large commercial fleets of aircraft.

2. BACKGROUND

This paper addresses the problem of detecting anomalies in high-dimensional, multivariate data streams containing both discrete and continuous channels. We assume that we are given data from a data generating process that can be functionally described by the following equations:

$$\mathbf{h}_t = \Gamma(\mathbf{h}^*_{t-1}) \quad (1)$$

$$\mathbf{x}_t = \Psi(\mathbf{x}^*_{t-1}, \mathbf{h}^*_t, u_t) \quad (2)$$

$$y_t = \Omega(\mathbf{x}_t) \quad (3)$$

We assume that the function Γ determining the evolution of the hidden system state $\mathbf{h}_t$ is unknown. We also assume that the function Ψ, which governs the evolution of the continuous state vector is also unknown. We assume that the vector $\mathbf{x}$ is an N dimensional state vector, and $\mathbf{x}^*_{t-1}$ is its history for the last D time steps: $\mathbf{x}^*_{t-1} = [\mathbf{x}_{t-D}, \mathbf{x}_{t-D+1}, ..., \mathbf{x}_{t-1}]$. The quantity u_t is the observed system input, and y_t is the observed system output which can contain discrete, categorical, and continuous features. We assume that the entire data that is available, covering both inputs and outputs is given by the set $(\mathcal{U}, \mathcal{Y})$.

In real-world flight operations, the pilot inputs $\mathcal{U}$ are determined by standard-operating procedure: for a given airport, aircraft, weather conditions, instructions from air traffic control, and other contextual elements of the flight, the flight procedures are well determined. However, while we can assume that all pilots are attempting to follow the standard operating procedures, some may deviate from these procedures which could lead to a different input sequence $\mathcal{U}'$, resulting in a different set of observed flight characteristics $\mathcal{Y}'$. The evolution of $\mathcal{Y}'$ over time will necessarily be different than that of the nominal case. This does not imply that the pair $\mathcal{U}', \mathcal{Y}'$ is an unsafe flight– it is simply not typically observed. Our goal in this paper is to develop algorithms that can detect if the discrete pilot inputs $\mathcal{U}$, combined with the observation vector Y are nominal or off nominal and to

diagnose the reason why such a potential anomalous event was so designated.

2.1 Current State-Of-The-Art Algorithms on Anomaly Detection

Data-driven anomaly detection is an active area of research (see [6] for a detailed survey). For example, there are many anomaly detection methods that identify anomalies in the vector space. However SequenceMiner [4] is the only algorithm that can analyze discrete sequences. Preliminary experiments with SequenceMiner on data from commercial aviation indicate that it is able to find examples of mode confusion, in which a pilot loses track of auto pilot mode changes and therefore effects anomalous switching behavior in order to determine the auto pilot mode. Because cockpit switch behavior is represented mostly by sequences of discrete switches, SequenceMiner is a natural algorithm for finding anomalies in such sequences. Soem details on SequenceMiner will be provided in Section **??** .Other anomaly detection methods, such as Orca [2] and the Inductive Monitoring System (IMS) [11], find anomalies in multivariate continuous data. Both Orca and IMS are distance-based anomaly detection methods, in that they use a metric related to distance, such as the average Euclidean distance to its k-nearest neighbors, to assess the anomalousness of a point. Clearly, the greater the value of this metric, the more anomalous a data point is. In principle, one could use Orca and IMS with heterogeneous data—data containing both discrete and continuous variables. However, in IMS the discrete variables' contributions to the Euclidean distance may not reflect their true importance, and finding the appropriate way to incorporate distances in the discrete and continuous spaces into a common metric is a problem with no clear solution. Additionally, Orca and IMS do not learn the sequential dependencies between points in the training data. Here we would like to identify an anomaly detection method that incorporates both continuous and discrete sequences and is able to identify anomalies within each separately but also across the two types of data.

Kernel methods [16] have been used for many different types of data with various types of features such as graphs [12] and multiple feature types in computer vision such as color [7], shape, texture [19], and graphs based on image segmentations [9]. Kernel functions map pairs of objects to the similarity between those objects, with a value of 1 indicating maximum similarity and 0 indicating no similarity. Therefore, subject to Mercer's conditions [5], one can devise a kernel function measuring similarity among objects of any type and incorporate this into kernel methods for classification, regression, or anomaly detection. This flexibility to incorporate kernel functions of different types motivates the question of whether multiple such kernel functions can be incorporated simultaneously. This question brought about the field of Multiple Kernel Learning (MKL) [1, 13]. MKL replaces individual kernel functions with combinations of kernel functions, thereby allowing the kernel method to use multiple kernels simultaneously. MKL was initially used with multiple copies of the same kernel function with different hyperparameter settings. However, MKL has been found most effective in cases where the different kernel functions use different attributes, such as in computer vision where SVMs that use a convex combination of kernels using multiple feature sets such as color, shape, and texture, have been found to outperform SVMs that only use one of the kernels [9].

MKL appears to be a promising way to satisfy our requirement of incorporating both discrete and continuous sequences in anomaly detection. We use the kernel anomaly detection method known as one-class Support Vector Machines (SVMs) [17, 15]. We incorporate a kernel over discrete sequences which is based on the normalized Longest Common Subsequence (nLCS) measure used in SequenceMiner and a kernel over continuous sequences that makes use of the Symbolic Aggregate approXimation (SAX) [14] representation. We demonstrate that this Multiple Kernel Anomaly Detection (MKAD) algorithm outperforms Orca and SequenceMiner at finding operationally significant anomalies in aviation safety data. To our knowledge, MKAD has never been used with sequences or within one-class SVM prior to our very recent poster which shows preliminary results of an earlier version of our algorithm on synthetic data only [8].

3. ALGORITHM

We first give a brief but general description of multiple kernel learning based detection technique. Then we describe the two kernels that we use within our model.

3.1 Multiple Kernel Learning based Detection

As mentioned earlier one of the major advantage of kernel based methods compared to other techniques is their ability to combine information from multiple data sources. The way this improved knowledge about the problem can be incorporated in the core optimization problem is very simple yet meaningful. The resultant kernel K can be a convex combination of all kernels computed over multiple features i.e. $K(\vec{x}_i, \vec{x}_j) = \sum_{p=1}^{n} \eta_i \hat{k}_p(\vec{x}_i, \vec{x}_j)$, with $\eta_i \geq 0$ and $\sum_{i=1}^{n} \eta_i = 1$. Here $\hat{k}_p(\vec{x}_i, \vec{x}_j)$ represents the p^{th} kernel computed, and η to weight individual kernels. Here we take advantage of the multiple kernel learning approach to incorporate more knowledge in the decision process so that we can achieve an improvement in detecting anomalies in complex heterogeneous systems that involves various data sources and data structures. Since we are interested in anomaly detection we have used the classical One-class SVMs [15] as our core algorithm. One-class SVM is a semi-supervised learning method that finds a set of outliers using a decision boundary. Below we will provide a brief description of some of the properties of the mapping function and talk a bit on the optimization problem of One-class SVM.

One-class SVMs constructs an optimal hyperplane in the high dimensional feature space by maximizing the margin between the origin and the hyperplane. This is done by solving an optimization problem [15]. The dual form of the optimization can be written as,

$$\begin{aligned} &\text{minimize} \quad Q = \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j k\left(x_i, x_j\right) \\ &\text{subject to} \quad 0 \leq \alpha_i \leq \frac{1}{\ell\nu}, \sum_i \alpha_i = 1, \rho \geq 0, \quad \nu \in [0,1] \end{aligned} \tag{4}$$

where α_i is Lagrange multiplier, ν is a user specified parameter that defines the upper bound on the training error, and also the lower bound on the fraction of training points that are support vectors, ρ is a bias term and k is the kernel matrix. Once this problem is solved at least $\nu\ell$ training points

with non-zero Lagrangian multipliers ($\vec{\alpha}$) are obtained and these points $\{x_i : i \in [\ell], \alpha_i > 0\}$ are called support vectors. The selected points can be marginal $\mathcal{I}_m = \{i : 0 < \alpha_i < 1\}$ and non-marginal $\mathcal{I}_{nm} = \{i : \alpha_i = 1\}$ support vectors. Once $\vec{\alpha}$ is obtained, SVMs compute the following decision function.

$$f(\vec{x}_j) = sign(\sum_{i \in \mathcal{I}_m} \alpha_i k(\vec{x}_i, \vec{x}_j) + \sum_{i \in \mathcal{I}_{nm}} k(\vec{x}_i, \vec{x}_j) - \rho) \tag{5}$$

If the decision function predicts a negative label for a given test point x_j, then it is classified as an outlier. Test examples with positive labels are classified normal.

3.2 Building Kernel

In this research our kernel takes the form of,

$$k(\vec{x}_i, \vec{x}_j) = \eta K_d(\vec{x}_i, \vec{x}_j) + (1 - \eta) K_c(\vec{x}_i, \vec{x}_j) \tag{6}$$

where K_d is a kernel over discrete sequences, K_c is a kernel over discretizedd continuous time series, and η is used to weight the two kernels (in this paper, we always use $\eta = 0.5$). We have[1]

$$K_d(\vec{x}_i, \vec{x}_j) = \frac{|LCS(\vec{x}_i, \vec{x}_j)|}{\sqrt{l_{\vec{x}_i} l_{\vec{x}_j}}}, \tag{7}$$

where $l_{\vec{x}}$ is the number of symbols in sequence $\vec{x}$. Given two sequences X and Z, Z is a subsequence of X if removing some symbols from X produces Z. Z is a common subsequence of sequences $\vec{x}_i$ and $\vec{x}_j$ if Z is a subsequence of both $\vec{x}_i$ and $\vec{x}_j$. The longest such subsequence of $\vec{x}_i$ and $\vec{x}_j$ is called the longest common subsequence (LCS) and is denoted by $LCS(\vec{x}_i, \vec{x}_j)$ and $|LCS(\vec{x}_i, \vec{x}_j)|$ is length. LCS is a useful metric for measuring similarity between discrete sequences for two reasons. First, it is not restricted to a location-based one-to-one match—the LCS can be located within different parts of the two original sequences. Second, the LCS length has an optimal substructure property which is the foundation of a well-known dynamic programming algorithm. In particular, the algorithm builds up a table L such that entry $L(i, j)$ is the length of the LCS of the first i symbols in $\vec{x}_i$ and the first j symbols in $\vec{x}_j$. Entry $L(i, j)$ only depends on entries of L for lower values of i and j. Details on the Hunt-Szymanski algorithm which we used to calculate LCS can be found in [10].

The continuous kernel $K_c(\vec{x}_i, \vec{x}_j)$ is inversely proportional to the distance between the SAX representations [14] of $\vec{x}_i$ and $\vec{x}_j$. Briefly, $\vec{x}_i$ and $\vec{x}_j$ are first divided into some number w bins along the time axis. That is, if both vectors have v variables for n consecutive time points each, then they are divided into w consecutive bins with all v variables and of mostly equal time length (all but the last one contain $\lfloor n/w \rfloor$ consecutive time steps and the last bin contains the remainder). The mean value of each variable in each frame is then calculated. So, for example, $\bar{\vec{x}}_{iab}$ is the mean of the values in the bth time interval of the ath variable of $\vec{x}_i$,

$$\bar{\vec{x}}_{iab} = \lfloor n/w \rfloor \sum_{k=\lfloor n/w \rfloor (b-1)+1}^{\lfloor n/w \rfloor b} \vec{x}_{iak}. \tag{8}$$

where $\vec{x}_{iak}$ is the kth time point in the ath variable of $\vec{x}_i$. Then, for each variable, we fit a normal distribution to all the training data, choose a number of bins c_a, and then find equiprobable bins with breakpoints $\beta_{a,1}, \beta_{a,2}, \ldots, \beta_{a,c_a-1}$ such that the area under the normal density function for $x \leq \beta_{a,1}$, $x \in [\beta_{a,k}, \beta_{a,k+1}]$ for all $k \in \{1, 2, \ldots, c_a - 2\}$, and for $x \geq \beta_{a,c_a-1}$ are each $1/c_a$. We assign each bin a discrete label (e.g., letters A, B, C, etc.). We replace $\bar{\vec{x}}_{iab}$ with the corresponding discrete label.

So $\vec{x}_i$ and $\vec{x}_j$ started off having v continuous variables and n time points per variable, and are replaced by a new matrix that still has v variables but has w discrete symbols per variable representing the means of that variable in the w consecutive time windows. The advantages of implementing SAX are that both the time and amplitude discretization result in reduction of the noise content and as well as dimensionality of the data. The distance between the SAX representations of $\vec{x}_i$ and $\vec{x}_j$ is simply the nLCS length as shown above in equation (7). Looking at empirical formulation of the similarity measure and how the above kernels are constructed it is pretty strightforward to understand that both K_d and K_c are symmetric positive semi-definite matrices.

3.3 Baseline Algorithms

Both Orca and SequenceMiner have been chosen as the baseline algorithms. To the author's knowledge, besides Orca no other anomaly detection algorithm exists that can handle both discrete and continuous data streams. Orca [2] is a K-nearest neighbor approach outlier detection algorithm with a modified pruning technique. For continuous data, Orca takes a nominal reference data set and calculates the nearest neighbors' using Euclidean distance to all test points in the original vector space. For binary data points the Hamminging distance is used. Each data point is scored independently and therefore anomalies in the temporal domain are undetectable. SequenceMiner computes outliers by comparing a set of sequences using the normalized longest common subsequence as the similarity metric. Sequences that are similar are clustered together. Outliers are sequences that have a very low similarity values with the clusters medoid. Since sequenceMiner takes into account the order it has the ability to identify anomalies in the temporal domain, however it is unable to handle continuous data and therefore does not have the ability to detect anomalies in continuous parameters. Both the baseline algorithms have their codes open sourced and can be obtained from the following links[2]

To test the robustness of the MKAD method synthetic data was generated with various types of seeded faults, allowing the algorithm to demonstrate its ability to detect each anomaly and for comparison against the existing state-of-the-art algorithms. Finally the MKAD method was compared with the combined performance of Orca and SequenceMiner.

[1] In all equations related to the discrete and continuous kernels, we assume that the discrete and continuous parts of the data points $\vec{x}_i$ and $\vec{x}_j$ are selected. To reduce notational clutter, we will not include operators to select the discrete or continuous parts of the data points.

[2] https://dashlink.arc.nasa.gov/algorithm/orca/ and https://dashlink.arc.nasa.gov/algorithm/sequenceminer-algorithm/

3.4 Synthetic data sets

To simulate an aircraft system it was assumed that the pilot inputs (the discrete switches) are used to influence the measured continuous output parameters. Therefore the data parameters were generated with this in mind. Ten binary parameters were generated with three fundamental behaviors: random flipping, constant throughout, and deliberate switching. One parameter was set to randomly switch between 0 and 1, while two parameters never changed states. For the deliberate switching six channels would hold a value at their initial state and change to the alternate state when a separate channel toggled 0 to 1.

With the binary parameters generated, the underling system state can be used to generate continuous data. To construct the continuous data, each continuous parameter was assigned a set of binary parameters as input variables. A set of Gaussian distributions defined for each possible binary state corresponding to the continuous parameter. For example: if a given continuous parameter was dependent on 2 binary parameters 4 distributions would be generated, if 3 binary parameters, 8 distributions and so forth (see figure 3). At each time step the continuous parameters would draw from its defined distribution for the given state of the binary parameters. This method allows the continuous parameters to vary directly with the state of the binary inputs and therefore have the desired relationship assumed for this problem.

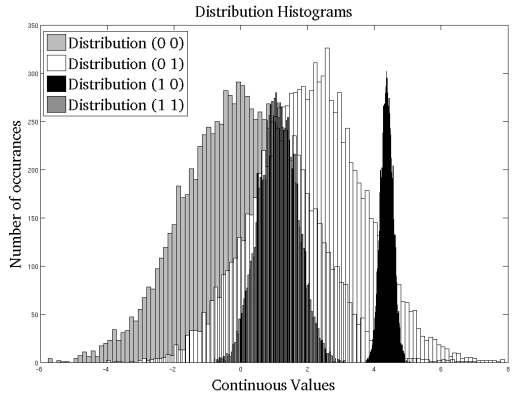

Figure 3: The figure demonstrates a sample distribution for a single continuous parameter that is dependent on 2 binary parameters.

This method was repeated for each flight in the data set. A total of 4000 flights were synthesized (2000 for training and 2000 for testing)[4]. Each flight is 1500 sample points long. Four different fault types, three examples of each, were injected into randomly chosen flights. bodydetails

Fault type I: The first type of fault involves missing switches and this implies that sequences of switching that was expected at a given stage did not occur. For example, such an event takes place when flaps not extended to normal full deployment at landing.

[4]The synthesized data set can be downloaded from https://dashlink.arc.nasa.gov/topic/multiple-kernel-learning-based-heterogeneous-algorithm/

Table 1: The table represents the summary of the performance of all three algorithms in detecting the faults in the synthetic data for each fault category. A total of 12 faults have been randomly injected, out of which 3 are continuous and 9 are discrete. Clearly MKAD was the only algorithm to detect all fault types.

Algorithms	Correct detection (of faults)	
	Discrete	Continuous
Orca	0	100%
SequenceMiner	89%	0
MKAD method	**100%**	**100%**

Fault type II: The second kind of fault involves extra switching where sequence of switching that were not expected did occur, such as landing gear retracted after being deployed on final approach.

Fault type III: The final kind of discrete fault describes out of order switch sequence. A typical example is landing gear deployed before initial flaps when the aircraft was below flaps limit.

Fault type IV: Apart from all the above three, abnormal patterns (independent of discrete variables) were injected in arbitrary continuous channels. Such an anomaly may occur with high bank angles or rate of descent below 1,000 ft.

After the data was generated some preprocessing steps are required before the algorithm can be implemented. The details of the preprocessing steps have been discussed in a later section. Here we propose to conduct a proof- of-concept study that demonstrates the feasibility of using the proposed MKAD algorithm in detecting variety of anomalies those injected in the synthetic data set.

Table 1 shows the summary of the outcomes. Since the actual fault injection incidents are known, we are able to evaluate the performance of all algorithms in detecting those faults. Out of twelve injected faults, Orca was able to find the three continuous anomalies. Even though Orca can handle both discrete (binary) variables and continuous variables, the algorithm is unable to detect sequential anomalies where the time information is embedded in some form. SequenceMiner, using 1-σ threshold calculated from the reference set, was able to detect most of the discrete anomalies and clearly missed all the continuous anomalies. Whereas the MKAD technique stands out accross all the algorithms since it was able to identify all twelve fault types (both discrete and continuous).

3.5 Real World Analysis on FOQA Data

The real world data set chosen for analysis is from a regional carrier in the U.S. who is part of the DNFA. All aircraft analyze were of the same fleet and type (narrow body jet), with a subset of flights that landed on the same runway at a single airport for an entire year, resulting in approximately 3500 total flights. Each flight consists of 160 parameters sampled at 1 Hz with the average flight length approximately 1:45 hours.

3.5.1 Data Preparation

Data analysis was focused on the portion of the flight between 10,0000 ft. Mean Sea Level (MSL) to landing, using the deployment of the thrust reversers as a means to determine touchdown. To account for parameters that recorded bad data, such as noisy sensors or sensor values reaching cut-off value or unreasonable data values, a conservative data quality filter was applied to all 3500 flights, returning approximately 2500 "cleaned" flights. Since the filtering was conservative, to insure that significant anomalies were not removed, some flights that partially contained bad data were not eliminated from the data set. An aggressive data quality filter was applied to the remaining set of flights to determine a nominal set for training (returning approximately 500). For parameter selection a domain expert provided a list of 39 relevant continuous parameters that were extracted for analysis. The flap position parameter was continuously recorded, however is categorical in nature. Using input from the domain expert and statistics from the data the flap parameter was discretizedd into 3 states. These 3 binary state variables were combined with landing gear and ground spoilers for sequence analysis.

The working data set consists of approximately 2500 flights with varying lengths and each of these flights are multidimensional heterogeneous time series. For continuous data, the mean and standard deviation are calculated for each parameter across all training flights. These statistics are then used in both training and testing to z-score normalize each parameter and flight to maintain consistency.

3.5.2 Experimental Details

When using Orca to analyze the flights, the z-score normalized temporal features of all the flights corrosponding to the training set were concatenated. The test set was generated by concatenating all the test flights. The discrete inputs to Orca are in standard binary format and the number of nearest neighbors were set to the default value (k=5). For SequenceMiner the binary states (of the discrete variables) were translated into state transitions where only the bit changes (switching) were logged as a sequence of transitions. Figure 4 represents a snapshot of ten such flights representing sequential data. The SequenceMiner model builds clusters in the sequence space and the number of clusters were determined to be 3 for this analysis. This is because we observed three distinct clusters in the reference set.

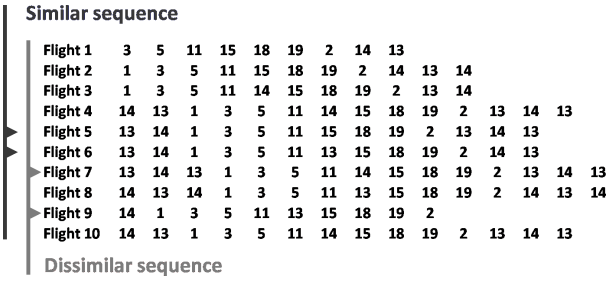

Figure 4: This figure represents a snapshot of a typical sequences generated from the binary input. Each sequence represents an unique flight. Examples of similar sequence (Flight 5 and Flight 6) and dissimilar sequence (Flight 7 and Flight 9) are shown.

For the multiple kernel heterogeneous algorithm, once the sequences are generated (Figure 4) the discrete kernel is computed pairwise across for all possible flight combinations in the training set. For the continuous data, each time series was SAX transformed [5] using the technique described in Section 3.2. In the original version of SAX, the z-score normalization is an integral part of the algorithm. However in this research, we normalized each time series (only once) before it is SAX transformed. We are able to maintain consistency in choosing the alphabet size for both reference and test sets. It is worth mentioning that the window size was also kept fixed through out the analysis. The window size and alphabet size were both set to 10. Once the SAX representations are obtained, another kernel is computed pairwise across for all possible flight combinations. Each element of this kernel is the average of the pairwise comparison across the parameters of any two flights. In the optimization, we have set the ν parameter of one-class SVMs to 0.1. For testing, the support vectors are used to calculate the pairwise similarity between all testing flights. The discrete and continuous kernels for test data were generated in a similar fashion as the training.

3.5.3 Result Summary

The MKAD method reported a total of 227 anomalous flights and assigned an appropriate anomaly score to each of these flights. A simple post-processing method is used to rank the anomalous flights and decompose the anomaly score of individual flight in terms of discrete (parameters) and continuous (parameters) contributions. This results in three distinct categories, a list of flights with anomalies in either discrete parameters or continuous parameters or in both (i.e. heterogeneous). The MKAD algorithm detected 19 discrete, 94 continuous and 114 heterogeneous anomalies. We have observed that the majority of the top ranking anomalies belong to the heterogeneous category. Some of these heterogeneous anomalies have distinct discrete-continuous interactions i.e. a sequence of unexpected events in the discrete parameters results in some abnormal effects in the continuous variables or a series of abnormal event in the continuous parameters prompts some necessary changes in the states of the discrete variables. We will elaborate on this using examples in the analysis section.

3.5.4 Statistically Significant Anomaly

Using the feedback from the domain expert we identified a good number of anomalous flights which are operationally significant while the majority are statistically significant. Most of the flights in the list of discrete anomalies and some from the heterogeneous category were identified anomalous because they fall outside the distribution of (most) observed values. This does not necessarily mean that the detected anomaly is operationally significant. Table 2 represents a typical distribution of the deployment of landing gears as a function of flap settings. According to the domain expert there is nothing unusual in deploying the landing gear before flap-1 setting (10°). In fact it is acceptable if the pilot slows down the aircraft by deploying the landing gear while transiting from cruise to descent. Such an example was picked up as a statistically significant anomaly due to the fact that it is a low occurance event. For similiar reason, we have observed a number of anomalous flights with either flap-1

[5]The source code of SAX can be obtained from the authors' website at http://www.cs.ucr.edu/ eamonn/SAX.htm.

Table 2: The table (top) shows the details on how landing gears are deployed as a function of flap settings. The second part of the table provides statistics on the typical settings of flaps during the landing phase. The numbers in each cell represents the percentage of flights that fall under that particular category. This information helps explaining why some of the statistically significant anomalies are detected by the MKAD algorithms.

Gear ordering	Before flaps 10°	Between flaps $10^\circ - 20^\circ$	Between flaps $20^\circ - 45^\circ$
% of flights	1%	78%	21%
Flaps setting	Flaps 10° above 10k feet	Flaps 10° below 10k feet + full flaps at landing	No full flaps deployed
% of flights	2%	96%	2%

setting before $10,000$ ft of altitude and/or full-flap not at all deployed during landing. Table 2 provides the statistics of various flap settings as a function of altitude. The other important category of anomalies which resulted from this analysis are clusters of flights with a common set of bad data sources (bad sensors) having common tail numbers which is a valuable maintenance information.

3.5.5 Operationally Significant Anomaly: Analysis

In this section we will present some examples identified as operationally significant by a domain expert. The first anomalous flight is a go-around (where the pilot aborted the landing, climbed, circled around, and landed) and is classified by the algorithm as a heterogeneous anomaly. The top anomalous continuous parameters are plotted in Figure 5. All continuous parameters show abnormally high deflections during the maneuver. The anomalies found in the discrete sequence confirmed the maneuver by identifying the extra switching due to the pilot retracting the landing gear and flaps during the climb and redeploying for landing on the second approach.

The second anomalous flight is also identified as heterogeneous anomaly with unusually high air speed when compared to a set of reference flights. Figure 6 shows the relationship between air speed and altitude with the air speed remaining high at 2500 ft MSL. The anomaly identified in the sequence indicates the landing gear was deployed before the flaps. The domain expert said that this ordering may not be unusual but the pilot could be using the landing gear to bleed off air speed, which is evident after the landing gear is deployed. This behavior demonstrates an interaction between the continuous and sequential variables, i.e. due to the aircraft's high speed at a low altitude the pilot was prompted to deploy the landing gear, which in return resulted in a delayed effect in a lower air speed.

The third anomalous flight is also identified as a heterogeneous anomaly with indications of gusty winds. The top contributing parameters are plotted in Figure 7, with the exception of wind speed, since it was not analyzed by the algorithm. The domain expert saw the large swings in drift angle and concluded that there may have been high winds. The wind speed plot shows that there were indeed high gusts of up to 70 mph during the approach, which is also apparent in the control column/wheel and lateral accelerations. The anomaly identified in the discrete sequence was that flaps fully deployed at 45 degrees was not present at landing. The domain expert said that landing with flaps set at 20 degrees is considered an approved landing flaps setting for this particular aircraft and may not indicate an anomaly, however during high cross wind conditions full flaps provide additional lift that can make it hard for the pilot maintain the aircraft's course, and therefore the pilot may not have deployed full-flaps because of these environmental conditions.

The fourth anomalous flight is an abnormal approach and the only one identified as a continuous anomaly (refer Figure 8). The flight shows high control column and fuel flow fluctuations beginning slightly before 10 miles to landing, which coincides with the altitude fluctuations. The domain expert said this is an interesting flight since the pilot had abnormal altitude deflections and was under glide slope, however the pilot was still able to line up on glide slope at 5 miles to landing, which is required to maintain a stable approach.

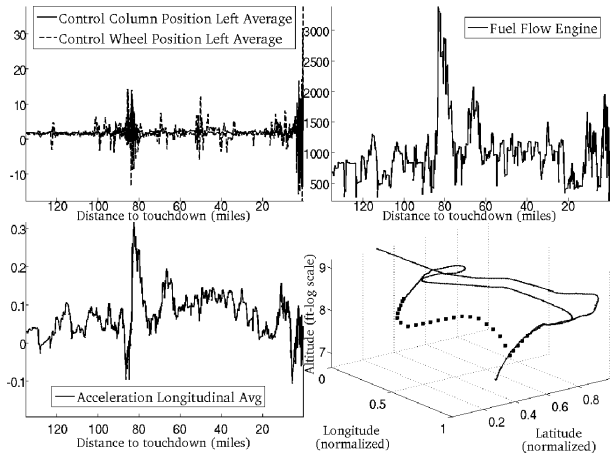

Figure 5: Top anomalous parameters associated with a go-around. The top left plot shows the control column and wheel positions associated with the maneuver. The top right plot shows high fuel flow consumption related to the climb. The bottom left plot shows the high acceleration in the longitudinal direction. The bottom right plot shows the 3 dimensional track of the aircraft (the go-around maneuver is denoted by the dotted lines)

3.5.6 Discussions

It is important to note that the combined anomaly lists from Orca and SequenceMiner were able to detect some but not all of the 4 anomalies just discussed. For the go-around and air speed anomalies SequenceMiner is able to detect the anomalies, however Orca did not. Both the anomalies have components that are sequential in nature (the go-around having additional flaps and landing gear switches, and the air speed anomaly deploying the landing gear early to bleed off speed) and, therefore it upholds that SequenceMiner de-

Figure 6: For the high airspeed anomaly the top anomalous parameter (airspeed) is plotted against altitude. The speed limit threshold at 10,000 ft. is denoted by the dashed line at 250 knots.

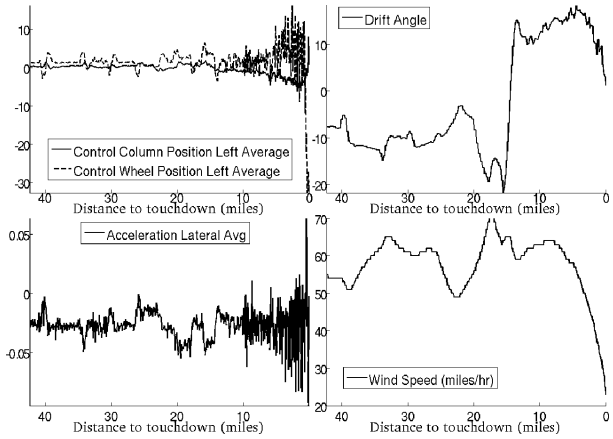

Figure 7: Top anomalous parameters associated with the gusty wind anomaly. The top left plot shows the control column and wheel positions. The top right plot shows the high drift angle. The bottom left plot shows high lateral (side-to-side) accelerations. The bottom right plot shows the wind speed gusts.

tects the anomaly while Orca, who treats all points independently, did not. In the case of the gusty winds and abnormal approach anomalies, both Orca and SequenceMiner were unable to detect these anomalies.

Baseline results were obtained by running Orca, SequenceMiner and compared with those obtained from MKAD method on the FOQA data set described above. It has been observed that the average flight length is approximately 1500 sample points in this data set. Based on this information we allowed Orca to report back the same number of sample points ($\approx 350,000$) which is equivalent to those 227 flights identified by MKAD. Any flight that had less than 15 sample points labeled anomalous which is approximately 1% of the average flight length and this resulted in 674 outliers reported by Orca. For SequenceMiner we have calculated a 2−sigma threshold from the reference set and considered any sequence that was above that threshold in the test set

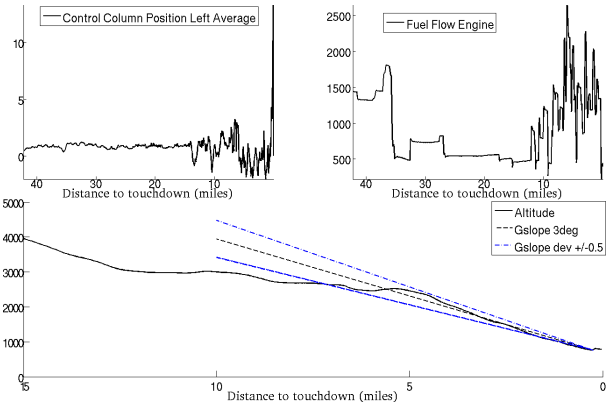

Figure 8: Top anomalous parameters associated with the abnormal approach anomaly. The top left plot shows the high control column position. The top right plot shows the large swings in fuel flow activity. The bottom plot shows the altitude vs. the distance to landing with the glide slope and glide slope deviation superimposed.

as an outlier, resulting in 72 anomalous flights.

Table 3 shows the overlap between the baseline algorithms and MKAD. As expected Orca performs well at detecting mostly the continuous anomalies found by MKAD, while SequenceMiner identifies anomalies related to discrete and/or heterogeneous anomalies found by MKAD. However MKAD still finds a significant number of anomalies that the combined baseline set does not detect. From the proof of concept study we have observed these limitations of Orca and SequenceMiner in identifying anomalies which are respectively discrete and continuous in nature. This is due to the fundamental nature of the baseline algorithms. However the MKAD is able to compress and appropriately combine the information from both discrete and continuous domain, to detect anomalies. This has also been reflected in the analysis of the real world data set, where the baseline algorithms missed some of the operationally significant anomalies detected by MKAD.

4. CONCLUSIONS

The current state-of-the-art algorithms have both strengths and short comings in detecting a variety of anomalous conditions, as discussed in the detection of the 4 real world anomalous flights. The MKAD method aims to combine both strengths into a single approach to allow for detection of a variety of anomalies. This is not to say the proposed algorithm is able to find all possible anomalies in the data, but rather that it is robust enough to find a significant overlap with the current state-of-the-art methods while also detecting additional operationally significant anomalies in heterogeneous data sources. Other approaches such as exceedance queries can be very efficient in detecting specific anomalies, however the goal is not to continue to find anomalies that are already being studied, but to develop a novel method that "*..answers some of the questions what we didn't think to ask..*" while analyzing FOQA data set.

5. ACKNOWLEDGMENTS

This project was supported by the NASA Aviation Safety

Table 3: In this table we compare the detection performance of Orca, SequenceMiner and the MKAD algorithm on aviation safety data set. The objective is to show the statistics of overlapping anomalous flights which fall under either discrete, continuous and heterogeneous categories. Total number of anomalous flights detected by each algorithms are also reported. The results of the MKAD algorithm is also compared with the combined outcome of Orca and SequenceMinier.

Algorithms	Overlap of anomalous flights (with MKAD)		
	Discrete	Continuous	Heterogeneous
Orca	21%	59%	34%
SequenceMiner	30%	0%	54%
Combined Orca and SequenceMiner	58%	59%	67%
MKAD method	**19**	**94**	**114**

Program, Integrated Vehicle Health Management Project. The authors would like to thank Robert Lawrence for his invaluable domain expertise and Dr. Irv Statler for his insightful discussions.

6. REFERENCES

[1] F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *International Conference on Machine Learning*, 2004.

[2] S. D. Bay 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.

[3] N. T. S. Board. Loss of control and impact with pacific ocean alaska airlines flight 261. 2002.

[4] S. Budalakoti, A. N. Srivastava, and M. E. Otey. Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 39(1):101–113, 2008.

[5] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *DMKD*, 2:121–167, 1998.

[6] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 2009.

[7] O. Chapelle and P. Haffner. Support vector machines for histogram-based classification. *IEEE Transactions on Neural Networks*, 10(5):1055–1064, 1999.

[8] S. Das, B. Matthews, K. Bhaduri, N. Oza, and A. Srivastava. Detecting anomalies in multivariate data sets with switching sequences and continuous streams. In *NIPS 2009 Workshop: Understanding Multiple Kernel Learning Methods*, 2009.

[9] Z. Harchaoui and F. Bach. Image classification with segmentation graph kernels. In *Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[10] J. Hunt and T. Szymanski. A fast algorithm for computing longest common subsequences. *Communications of the ACM*, 1977.

[11] D. L. Iverson. Inductive system health monitoring. *Proceedings of the 2004 International Conference on Artificial Intelligence (IC-AI'04), CSREA Press, Las Vegas, NV*, 2004.

[12] H. Kashima, K. Tsuda, and A. Inokuchi. Kernels for graphs. *Kernel Methods in Computational Biology*, 39(1):101–113, 2004.

[13] G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Bartlett, and M. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[14] P. Patel, E. Keogh, J. Lin, and S. Lonardi. Mining motifs in massive time series databases. In *International Conference on Data Mining*, 2002.

[15] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[16] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

[17] D. M. Tax and R. P. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(1113):1191–1199, 1999.

[18] A. S. Team. Statistical summary of commercial jet airplane accidents. Technical report, Boeing Commercial Airplanes, 2007.

[19] H. Zhang, A. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *Computer Vision and Pattern Recognition*, pages 2126–2136, 2006.