Operation Mode determination and Regime-based anomaly detection using Unsupervised methods

Navid Zaman[1], Evan Apostolou[2], Heng (Jason) Jia[3], Yan Li[4], Jacek S. Stecki[5], Chris Stecki[6]

[1,2,3,4,5,6] *PHM Technology, Melbourne, Victoria, Australia*
navid.zaman@phmtechnology.com

Abstract

Most modern engineering systems have multiple functions and thus often operate in various, distinct modes. Therefore, diagnosing such systems for anomalies and failures must take into account the specific mode that the system is operating within. An operating mode or regime in which a machine is functioning is defined by the various causal relationships between component functions within the system. This is often governed by a control system, affecting certain parameters which in turn influence the function and thus such control mechanisms, which are often disregarded during diagnosis, should become an integral part for determining failure status. Diagnostics, having known and understood the context of the operating mode of the system, is mostly accomplished using machine learning nowadays; given the large extent of the data available from sensors (Big Data), such technologies are quickly becoming the norm. However, in industry, there is a misplaced preference in utilizing purely probabilistic methods to accomplish this task without context. These methods are often affected by spurious correlation, rendering them unstable for mission critical systems. In this paper, we present a causation-based approach to the problem of reliable failure detection and isolation within the world of multi-purpose machines. It utilizes the contextual information of control systems and correlative methods using unsupervised machine learning algorithms. The correct domain knowledge is captured whilst taking advantage of algorithmic computations for quick predictions.

Keywords: Failure Isolation, Fault Detection, Machine Learning, Operating Modes, Regime, Correlation, Causation, Functional Flow, Control System, Big Data

Introduction

Failures or a disruption in a system's function are generally heralded by anomalies measured by sensors. However, almost all engineering systems may operate in various modes and these are likely to exhibit failures and anomalies differently. Since the failures and anomalies are specific to a mode, the methodology described here manage different information and train a group of models based on each mode. This paper outline a workflow using both domain knowledge and machine learning to determine the mode that a system may be operating in as well as any anomalies that occur. The terms Operating Mode (OpMode for short) and regimes are used interchangeably.

Operating Modes

Understanding which items of a system are active within an Operating Mode is crucial for diagnostics of anomalies and failures, since it allows for an assessment of potential breakdown time [1]. Before performing diagnostics on a failed system, it must be understood how Operating Modes are detected, and prior to this, we need to know what is an Operating Mode. This section will provide the background and definition of an Operating Mode.

What is an Operating Mode?

Most of the systems which are used in industry are usually built and programmed for specific tasks or operations under specific loading conditions, yet there are existing systems which perform more than one task or operation in different loading conditions [2], [3]. The tasks, operations and loading conditions which the system performs are called regimes or Operating Modes.
Operating modes are different modes in which a system is expected to behave for a segment of a mission, specified time or for a specific task [4]. Many machines and systems have more than one function or operate in various settings, where not all items in the system are active.

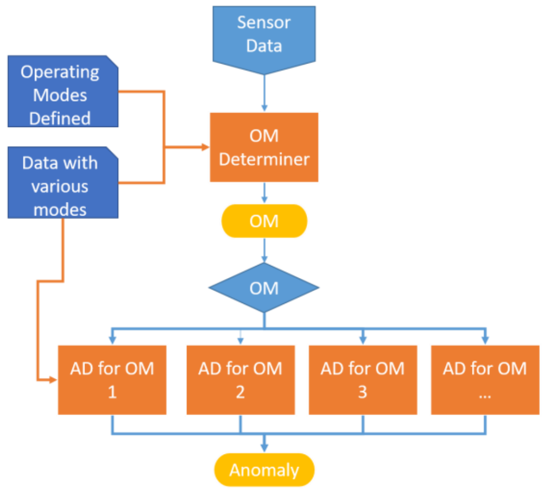


Figure 1. Workflow

An example of a system with Operating Modes is a Lifting System. This system will have 3 Operating Modes; lifting up, lowering down and stationary. Other systems will have discrete Operating Modes such as the Lifting System.

In each of the Operating Mode examples stated above are systems which have items in an active or inactive state but could also have items which are under different Operational Loading Conditions [5]. Active items in an Operating Mode are those which are operational, and are performing their intended functions (not in stand-by). Operational loading conditions describe the stress intensity or load at which an active item is performing. For example, a pump operating at 20% capacity compared to 80% capacity [3], [5].
It is these active components and operational loading conditions which give the regime detection algorithms the information to detect and determine different Operating Modes.

Data for Regime Detection & FDI

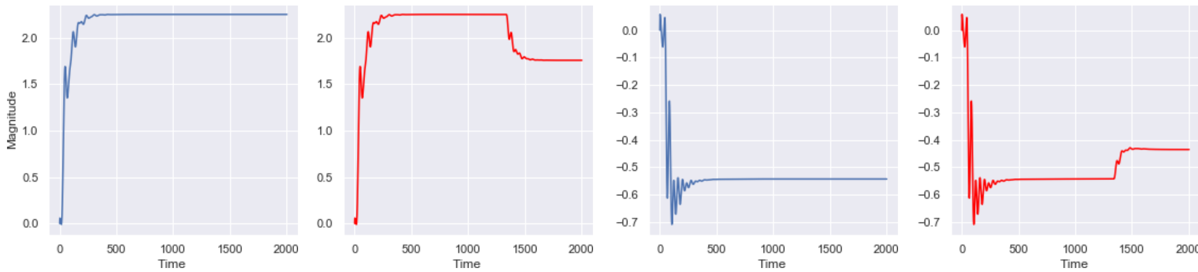
Before Operating Modes are identified by a regime detection algorithm, it is important to establish the data which is used. An extension to regime detection which has been mentioned earlier in this paper, is the diagnosis of failures in each Operating Mode. In this section, the data requirements for regime detection and Fault Detection and Isolation (FDI) will be described.

For regime detection and FDI capabilities to function as expected, the data which is fed into the algorithms needs to be accurate and proper. The most crucial information for these capabilities is system monitoring data - this is time-series data which comes directly from the system sensors [5], [6]. Figure 2 is an example of system monitoring data for Luffing System Discharge Boom velocity, where the different behaviour of the Discharge Boom is shown in two Operating Modes, as well as the behaviour in those Operating Modes when a failure occurs. The direction in which the Discharge Boom is travelling is defined by the sign of the velocity value - negative is up, and positive is down. For each failure curve (Figure 2b and 2d at 1350 seconds), the direction of the Discharge Boom remains the same, but the magnitude of the velocity decreases after each failure.
It is clear in these curves that the behaviour of this item changes between Operating Modes, and can be used to determine which Operating Mode the system is in.

While regime detection uses system monitoring data, there is supporting information, or secondary data which can be utilised in these algorithms to improve the confidence of determining which regime a system is operating within. As a way of improving regime detection algorithm results, it is proposed that the secondary data can be created through the use of a model-based engineering tool and creating a digital behavioural twin of the system. This behavioural twin can be used to simulate the expected behaviour of items in different Operating Modes, but certain information for modelling this behavioural twin is required - this information is listed below.

When modelling Operating Modes, it is important to consider the characteristics listed below:

1. Sequence of expected Operating Modes
    - Time dependent conditional regime changes



(a) Down (No Failure) (b) Down (Failure) (c) Up (No Failure) (d) Up (Failure)

Figure 2. Discharge Boom Velocity - Up and Down Regimes

- Conditional regime changes (time independent)

2. Active/inactive items
3. Operating loading conditions

To accurately model a system, it is crucial to understand the system configuration as well as the items which will be active/inactive in certain regimes and operational loading conditions - this will aid in the understanding of which failures are expected to occur during particular Operating Modes. In addition to this, understanding the conditions in which a system may switch regimes, aids in defining the triggers in the model which cause the system to change regimes.
The secondary data described is capable of being used as training data for the regime detection algorithms and ultimately, anomaly detection and FDI.

ANOMALIES BASED ON OPERATING MODES

Types of anomalies

Outliers or anomalies (used interchangeably, since they will be detected the same way in this paper) come in three distinct flavors [7]:

- **Point/Global** These are individual anomalies present across the dataset which exhibit more globally and can usually be detected regardless of the mode that the machine is operating in. These are often difficult to detect on account of their more random nature.
- **Group/Sequential** Anomalies that often cluster due to similar behaviour or temporal dependencies on previous values. These may be incorrectly classified as separate regimes entirely when attempting to determine the mode.
- **Contextual** Outliers which correspond and exhibit themselves based on previous signal values and the OpMode that the machine is in. Without some understanding of OpMode, these are easily confused as either point anomalies or more often, healthy samples.

While a global anomaly detector may be used for point and group anomalies, the contextual type is the one that is usually more difficult to detect and is the one dealt with in this paper.

DETERMINATION OF OPMODE

Machine Learning to determine OpMode

Unsupervised clustering learning is suitable in operation mode determination. Comparing to using supervised learning to determine operation modes, unsupervised clustering learning has the flexibility of no requirement on labels. Labeled system monitoring data is rarely readily available, especially for complex engineering systems with various operation modes. Another advantage of clustering learning that is it does not require all system data under all operation modes since clustering algorithms will find the patterns from operation mode features and characteristics.

In general, we can categorise time series clustering into: Clustering by time point (similarity in time), clustering by shape (similarity in space), and clustering based on deep learning. Popular machine learning algorithms like K-Means[8], DBSCAN[9] requires extracted statistical features from time series signals. The most popular clustering algorithms perhaps, that requires prior knowledge of number of clusters *k*. It initializes *k* respective centroids randomly and calculate the distance from each data point to the centroids, find the nearest centroid, then it belongs to that cluster. It will update the centroid of each cluster, repeat until each point is very close its cluster centroid.

With the development of deep learning, clustering has been developed into a new direction that is Deep Clustering. Popular choice of deep clustering algorithms are based on autoencoder which can transform time series data into low dimensional latent space. For example, current variational auto encoder can tolerate noise or anomalies, however, it is lack of a universal method to capture the characteristics of time series data, so as to obtain effective latent space. In the latent space obtained, a suitable similarity measure is needed to consider the characteristics of time.

The current state of art method, Deep Temporal Clustering (DTC)[10], takes advantage of auto encoder to generate latent representation and the clustering principle of K-Means. In the encoder, a convolutional layer captures short-range fluctuations between sequence. The outputs dimensions are reduced in the max pooling layer. It effectively compress the time-series sequences and remain structured information between sequences. The dimensionally reduced data is further passed through Bi-LSTM layer, where the data is further compressed into a more compact latent representation.

The latent representation is passed to a clustering layer where calculates the t-Distribution probability q of each sequence belonging to each cluster centroid. Centroids are updated like K-Means by maximising confident assignment using target distribution p. The clustering layer is iteratively trained till reach the minimal Kullback–Leibler (KL) divergence loss between p and q.

This network's architecture extracts spatio-temporal features from signal sequences, which takes consideration of both clustering-by-point and clustering-by-shape as stated as above. Additionally, the autoencoder layer and clustering layer is jointly optimised by Mean Squared Error and KL divergence correspondingly. This joint optimisation approach outperforms the traditional approach whose data compression and clustering are optimised separately.

Context for predictions

Although unsupervised learning is more suitable for operation mode detection, it has the limitation of not producing labels that are mapped to the corresponding regimes. Thus, it is inevitable to validate the true representation of predicted labels.

In the case where training data are labelled, the labels predicted by clustering and the ground truth can be efficiently mapped and coupled by the Hungarian algorithm or Kuhn-Munkres Algorithm. Both algorithms solves best assignment problem. A more straightforward solution is to found the best matching couples from confusion matrix between prediction and ground truth. An example is demonstrated in the the Figure 3a. We can see the best coupling between ground truth and prediction labels in the example confusion matrix. Prediction label '4' best matches to true label '2'.

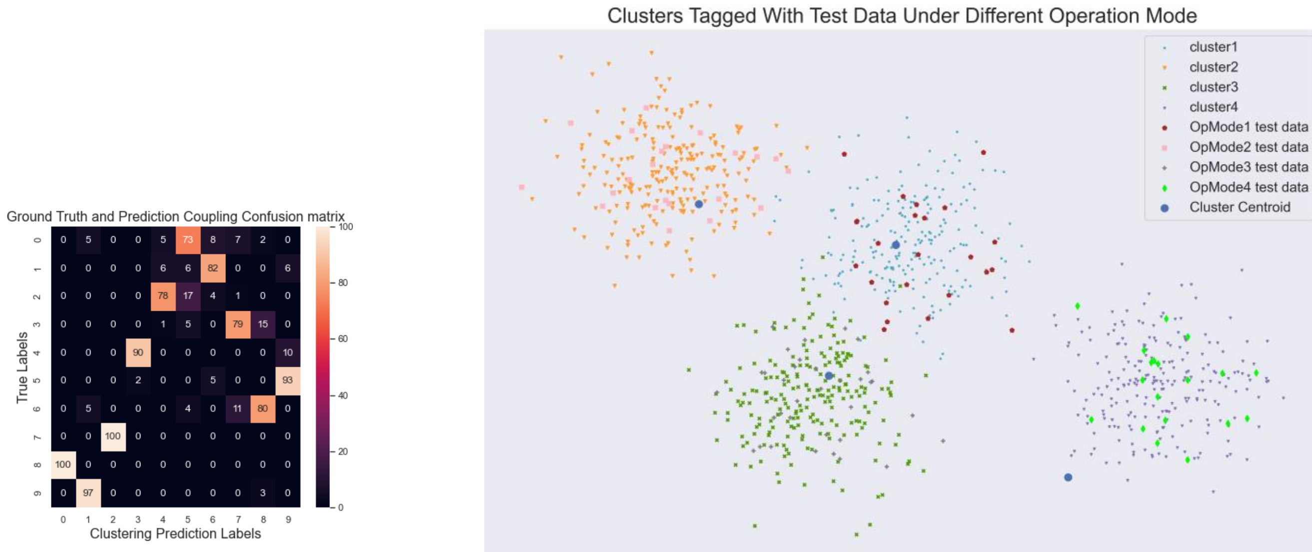
In the case of training data is not labelled, we could collect a small dataset (compare to training data) from different operation modes that is labeled in conjunction with domain knowledge. This label-validation data is expected to present a complete cycle of operation modes. This method allows clustering algorithm to be trained with numerous amount of unlabelled system data, which is more practical in prognostic industry. Additionally, we can avoid solely relying on any prior assumptions and domain knowledge on historical events. In Figure 3b, we use synthetic data to demonstrate an example of four clusters found post-training, with the true labels from the later acquired data. We can then couple the cluster to its corresponding operation mode.

ANOMALY DETECTION

Anomaly detection refers to identifying observations that may be considered as anomalous given the distribution of samples. Any observation belong to the distribution deems as an inlier and any outlying observation is referred to as an outlier or anomaly. As mentioned previously, the operating mode will affect the type of anomalies that the system might

Machine Learning to detect anomalies

Machine learning is widely applied in anomaly detection. Anomaly detection can be interpreted as imbalanced classification problem if training data is labelled, in which each class label is not balanced and only a small group of outlying samples in the dataset. In general, labels are recommended to be used if available. However, due to availability restraints of anomalous labels of system monitoring data, two common approaches for anomaly detection: unsupervised, semi-supervised (Novelty) detection that are more practical in the context of prognostic maintenance. Three detector learning results are compared and visualised in Figure 4.



(a) Confusion matrix between true labels and opmodes

(b) Trained Clusters Tagged With Test Data Under Different Operation Mode

Figure 3. Visualisation of Coupling Techniques

Unsupervised

Unsupervised detection is ideal when training data contains a mixture of both normal and anomalous observation. During the training process, the model identifies outliers. It is researched that this method is suitable when anomalies are defined as points which in low density regions among the data. Thus, any new observations do not belong to high-density regions are deemed as anomalies.

Because the distribution of anomalous points is different from the normal points, the similarity is low, and a series of algorithms are derived to identify abnormal points through similarity. For example, the simplest K-Nearest Neighbor can be used for anomaly detection, and the distance between a sample and its *k-th* neighbor can be regarded as an outlier. Obviously, the *k-nearest* neighbor distance of an anomalous point is larger. In the same way, based on density analysis such as LOF[11], LOCI[12] mainly detect abnormalities through local data density. Obviously, there are few data points in the space where the anomalous point is located, and the density is low.

In a low-density space (the space where anomalous points are located), a sample of an isolated case requires fewer divisions. Another similar algorithm ABOD[13] is to calculate the variance of the angle formed by each sample and all other sample pairs. The anomalous point is far away from the normal point, so the variance of the variance is small.

Semi-supervised (Novelty) Detection

In the context of anomaly detection, semi-supervised detection requires training data consisting only normal observations. Algorithms are fit on the training data to form decision boundaries, then used to evaluate new observations. This approach is suitable for where anomalies are defined a points differing from the distribution of the normal training data. During evaluation on new observations, any differing from the training data within a learnt threshold will be deemed as anomalies, even it is from a high-density region.

PCA[14] can be employed in this scenario. One method is to find *k* eigenvectors, and calculate the reconstruction error (reconstruction error) of each sample after the *k* eigenvectors are projected, while the normal point reconstruction error should be smaller than the abnormal point. In the same way, it is also possible to calculate the weighted Euclidean distance from each sample to the hyperspace formed by the *k* selected eigenvectors (the smaller the eigenvalue, the greater the weight). Under similar principle, we can also directly analyze the covariance matrix, and use the Mahalanobis distance of the sample (the distance between the sample and the center of the distribution when considering the relationship between features) as the abnormality of the sample.

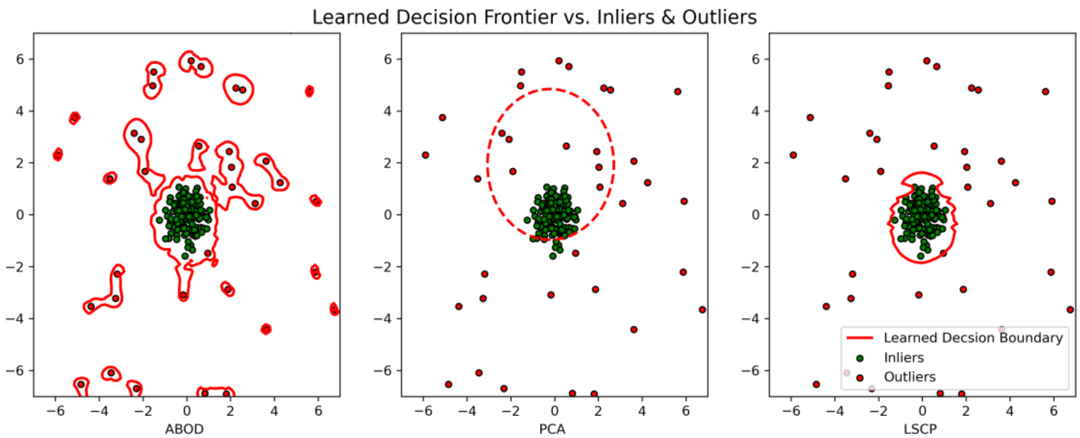


Figure 4. Comparison of Detectors' Learned Decision Boundaries

Detector Ensemble

Due to anomaly detection being usually unsupervised, complicated and works on imbalanced data, it is important to improve the performance and robustness of the model, so ensemble learning is very useful. The earliest integrated detection framework feature bagging[15] is very similar to the random forest in the classification problem. The training data is first divided randomly (d/2 to d features of all samples are selected each time, and d represents the number of features) to get multiple sub-training sets, and then train an independent model on each training set and finally merge all the model results (such as by averaging). It is worth noting that because there is no label, anomaly detection is often through bagging and feature bagging, while boosting is relatively rare. In the case of boosting, anomaly detection generally needs to generate pseudo-labels.

Given that local and global outliers are difficult to be detected by a single detector, Locally Selective Combination in Parallel Outlier Ensembles (LSCP) [16] takes a collection of base-detectors and dynamically selects suitable base-detectors based on pseudo-labels for each local region divided from original dimensions. A simple stacking ensemble does not select the base-detector, but finally directly select the average/maximum value of abnormal scores produced by all models as the final abnormal score of the sample. For such methods, if no model selection exists, some models with poor performance will affect the performance of the combined model. LSCP supporting heterogeneous or isomorphic base-detectors enables 'smart' job allocation. As illustrated in Figure4, LSCP learned fairly well and formed a reasonably accurate decision boundary comparing to the other two single detectors.

CONCLUSION

Regimes or operating modes dictate the type of anomalies that can occur. Various techniques to determine the mode and then detect anomalies based on the sensor signals received are described. The domain information required to model such aspects are mentioned, which compliment the correlation component. Hence, the physical definitions of the system are absorbed into the predictions, reducing spurious results.

REFERENCES

1. C. CEMPEL, "Multidimensional condition monitoring of mechanical systems in operation," *Mechanical Systems and Signal Processing*, vol. 17, no. 6, pp. 1291–1303, 2003.
2. P. Zhang, "Chapter 2 - industrial control engineering," in *Advanced Industrial Control Technology* (P. Zhang, ed.), pp. 41–70, Oxford: William Andrew Publishing, 2010.
3. Y. JIN, Z. SUN, Y. SONG, X. LIN, X. NIU, and J. DING, "Mission segment division of the whole aeroengine loading spectrum based on flight actions," *Chinese Journal of Aeronautics*, 2021.
4. K. T, "Risk analysis of phased-mission systems with multiple failure modes," in *IEEE Reliability Society 2008 Annual Technology Report*, 2008.

5. S. Yang, *An Adaptive Prognostic Methodology and System Framework for Engineering Systems under Dynamic Working Regimes*. PhD thesis, University of Cincinnati, 2016.

6. C. Bhattacharya and A. Ray, “Data-Driven Detection and Classification of Regimes in Chaotic Systems Via Hidden Markov Modeling,” *ASME Letters in Dynamic Systems and Control*, vol. 1, 08 2020. 021009.

7. G. Sebestyen, A. Hangan, Z. Czako, and G. Kovacs, “A taxonomy and platform for anomaly detection,” in *2018 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, pp. 1–6, 2018.

8. J. B. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability* (L. M. L. Cam and J. Neyman, eds.), vol. 1, pp. 281–297, University of California Press, 1967.

9. M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD’96, p. 226–231, AAAI Press, 1996.

10. N. S. Madiraju, S. M. Sadat, D. Fisher, and H. Karimabadi, “Deep temporal clustering : Fully unsupervised learning of time-domain features,” *CoRR*, vol. abs/1802.01059, 2018.

11. M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “Lof: identifying density-based local outliers,” ACM, 2000.

12. S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos, “Loci: fast outlier detection using the local correlation integral,” IEEE, 2003.

13. H.-P. Kriegel and e. a. Zimek, Aurthur, “Angle-based outlier detection in high-dimensional data,” ACM, 2008.

14. C. C. Aggarwal, *Outlier Analysis*. Springer, Cham, 2017.

15. A. Lazarevic and V. Kumar, “Feature bagging for outlier detection,” in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD ’05, (New York, NY, USA), p. 157–166, Association for Computing Machinery, 2005.

16. Y. Zhao, Z. Nasrullah, M. K. Hryniewicki, and Z. Li, “LSCP: locally selective combination in parallel outlier ensembles,” in *Proceedings of the 2019 SIAM International Conference on Data Mining, SDM 2019*, (Calgary, Canada), pp. 585–593, SIAM, May 2019.