Applications of Active Learning in Predictive Maintenance

You-Jung Jun[1], Navid Zaman[2], and Daniel Chan[3]

[1,2,3] *PHM Technology, Melbourne, Victoria, 3068, Australia*
youjung.jun@phmtechnology.com
navid.zaman@phmtechnology.com

Abstract

Nowadays, the common choice in maintenance strategies is predictive maintenance (PdM), deprecating the corrective and preventive kinds. Even with various machine learning techniques to get advanced predictive models to achieve PdM, difficulties remain in the data acquisition process. While there is a plethora of unlabeled data from sensors, most of those available techniques can only process labeled data, i.e, supervised learning. To combat the fact that the availability of the labeled data is limited, this paper proposes the use of Active Learning to *label* and *annotate* the informative instances while minimizing overall processing time. This approach maintains high performance and decreases the number of labeled instances, with support from experimental results and a discussion of the applicability of this method.

1. Introduction

Data is more than abundant in most Reliability, Availability, Maintenance and Safety (RAMS) solutions and implementations. Scalable and reliable infrastructures have been used to store these datasets (Aydin, Hallac, & Karakus, 2015), while the sensor readings themselves are collected and warehoused efficiently, the context behind it is not sufficient. Labeled data, which is annotated with some context, is crucial in the field of RAMs to provide meaningful utilization of sensor readouts. Amongst various reasons that render work with the low quality of the data, lack of human readibility and poor recording practices matter the most (L'Heureux, Grolinger, Higashino, & Capretz, 2017; Teh, Kempa-Liehr, & Wang, 2020). To alleviate this problem, various solutions have been proposed, ranging from machine-aided manual labeling to enhancing the quality of labeling (Woodward & Kanjo, 2020; Rosenthal & Dey, 2010). However, these methods require the person(s), an oracle to do all the annotation, to label the instances. This paper contributes to the literature by showing the use of Active Learning to efficiently choose particular samples for the oracle, while the rest is labeled via the machine. This improves cost-effectiveness and substantially reduces time and effort.

2. Data

Labeled data has several uses in the RAMS field. The main focus of this paper is failure detection isolation (FDI).

2.1. Failure Detection and Isolation

Diagnostics into the possible functional issues and/or root cause of a catastrophic failure may trigger immediate maintenance action or a post-mortem analysis may educate future endeavors (an example for gear tooth FDI shows such a process (Roan, Erling, & Sibul, 2002)). Data labeled with the correct failure or fault information within a reasonable window of when the event occurred is crucial for reliable performance.

2.2. Residual Useful Life

The knowledge of how long or how many cycles a system or component has left in performing a certain task is invaluable information for system operations and also for scheduling maintenance. This is where the residual useful life (RUL) is used (Wen et al., 2021). With knowledge of how similar components have run into failure, prediction methods may be informed to estimate the RUL of a monitored item. The label represents either the failure of the timestamp when it occurred or the remaining life in intervals prior to failure.

3. Problems with FDI Data

While the industry is employing constant improvement methods of collecting, storing and processing data for FDI-specific applications, the actual data often lacks quality for various reason noted in this section.

3.1. Lack of Useful Labels

In most cases, the work done in labeling is out of sync with the actual event occurrence. This means that such annotations are often neither reliable nor correct. This may cause poor performance in case when using machine learning methods or threshold-based strategies.

3.2. Ignored Data for Labeling

Machines are often monitored throughout their useful life without catastrophic failure events. This limits the useful trends and patterns that are required for detecting failures - the anomalous sections. Since there is no event to report, in those cases, none of the labels are recorded.This can occur with both healthy states and less important anomalies, however, a supervised FDI tool will have significantly reduced performance if events are not specified.

3.3. Poor Labeling Procedures

In certain instances, the proper reporting procedures are not followed and as a result, poor quality information are recorded. This may include descriptions lacking in detail or misleading information, therefore such labels are less useful and in worse cases, equivalent to no label at all.

4. ACTIVE LEARNING

Active learning, as one of the popular machine learning methods, is often deployed to deal with data sparsity by interactively querying an oracle (e.g., human annotator and expert). By selecting the most useful samples within a certain pool of unlabeled data points, it reduces the effort for the data labeling . While active learning is applicable in various situations (e.g., speech recognition and information extraction), the specific role of active learning is investigated by labeling queried data points and producing the model with the highest performance. This method is often implemented to improve the model's efficiency as it keeps the minimum number of supervised data points while achieving a higher level of performance. The general workflow of active learning includes selecting scenarios, estimators, parameters, and query strategies, demonstrated in Figure 1.

Figure 1 shows the active learning workflow, specifically designed for a pool-based scenario, which are used in our case study in section 5. As demonstrated, when the data comes in, it decides the type of scenario that would be implemented. Then, depending on whether it is labeled or not, classify them into either training data or pool data. Along with this, select the appropriate estimator and parameter. After that, based on the selected query strategy, the learner iteratively trains the model until this goes beyond the threshold.

4.1. Scenarios

Depending on the characteristics of original datasets (such as how many samples, how many labels, etc), different scenarios in active learning can be utilized. Three scenarios that are frequently used when the learning algorithm asks queries to label instances are: pool-based active learning, membership query synthesis, and stream-based selective sampling.

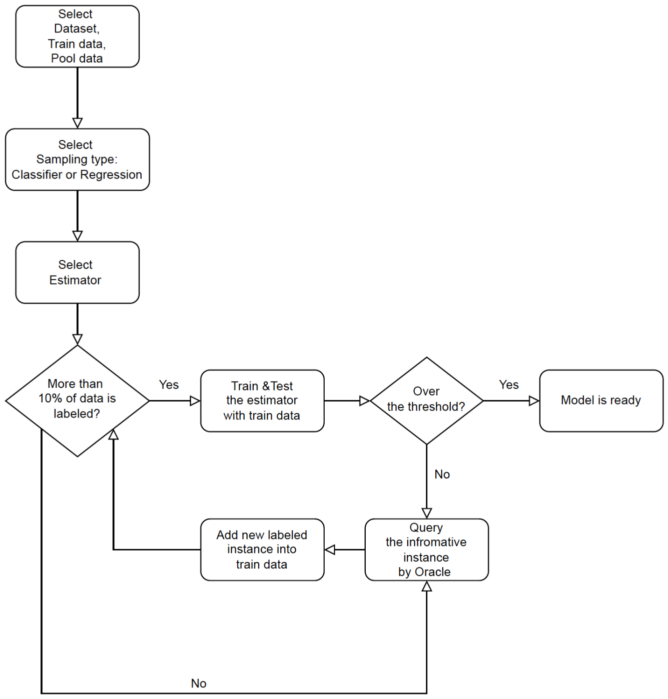

Figure 1. Active Learning Workflow.

First, the pool-based active learning, which is utilize in this work, is used when a large number of unlabeled instances are readily available. The learner evaluates the informativeness of ***all*** unlabeled instances at once and chooses the most informative instance to query the oracle. As this pool-based active learning only requires a small number of data points to get high-level labeled data, it is cost-effective. Also, it is a useful method as this is adaptable to numerous kinds of datasets. In section 5, we apply this pool-based scenario for our case study.

Membership query synthesis is the other form of scenario that is used when the learner wants to generate its own unlabeled instances rather than relying on the sample with the natural underlying distribution. This approach can be effective only when accessibility of the real data is limited as this only requires a smaller number of labeled data to train an accurate model. Although it is straightforward to create a data instance due to its compatibility, membership query synthesis has the possibility of misidentifying data.

Lastly, stream-based selective sampling works when there exists a continuous stream of data (e.g., online learning or real-time applications). In this scenario, the learner evaluates the informativeness of the unlabeled instances one at a time and only queries the data when the informativeness of the instance is over the threshold. While this scenario is efficient to handle the data with changing distribution and prioritizing the level of informativeness in a relatively shorter period of time, this might be a challenging scenario as it requires highly precise design and evaluation.

4.2. Query Strategies

Active learning process evaluates the informativeness of unlabeled data based on query strategies. As every active learning methodology evaluates the informativeness of unlabeled data, selecting the informative instance is a crucial process. More specifically, labeling instances is a costly task in terms of its time due to its limited number of queries. The three most commonly-used frameworks are Uncertainty Sampling, Margin Sampling and Query-by-Committee (QBC).

The Uncertainty-Sampling query strategy is one of the active learning approaches that iteratively trains the selected data points to classify them. With this sampling, the learner finds the nearest to the model's decision boundary by using a probabilistic model (Settles, 2009). The model chooses the most uncertain data with the lowest confidence in their predicted values, where it estimates the level of uncertainty based on the probability distribution of all possible labels.

Similar to the Uncertainty-Sampling query strategy, the Margin-Sampling query strategy is an effective method to classify the data points by actively selecting the informative ones to label. While both strategies are often used in active learning to improve the prediction capability of the model, they are different in ways measuring uncertainty. Margin-Sampling (Balcan, Broder, & Zhang, 2007) chooses the data points where the margin (i.e., difference) of the two predicted classes is the smallest. This assumes that with the small size of the margin, this implies that the area is more uncertain and needs to be improved for the classification.

QBC selects informative instances by querying a committee of learners rather than relying on a single learner (Settles, 2009). Multiple learners in QBC are trained with the same training data and each committee member votes for the unlabeled instances. The instance with the highest number of votes is considered the most informative one and is queried to the oracle for the label. While this alleviates the potential bias caused by actual learners from uncertainty sampling, it comes with significant computation costs, especially when working with high-dimensional data.

5. Case Study

In this section, by using predictive maintenance (PdM) data, a case study demonstrates the usability of active learning, examine how much this achieves the enhancement of model accuracy, and verify whether active learning solves the data sparsity issue. No clear rules for the required number of data points for the initial training dataset do exist, since this depends on various factors such as the complexity of the problem, the overall size of the dataset, and the availability of resources for annotation. Yet, the larger the initial training dataset is, the higher quality of performance is likely to be guaranteed, unless it does not drain the monetary budget assigned for annotating data.

	Failure Types N. of Obs.
No Failure	2363
Flex Shaft Coupling Angular Velocity Low	2110
Check Valve Hydraulic Flow High	1995
Check Valve Hydraulic Flow Low	1923
Oil Nozzle B1 Hydraulic Flow High	1901
Oil Nozzle B1 Hydraulic Flow Low	1817

Table 1. Types of failures and the number of observations per failure

5.1. Data

As a case study, a lubrication system for an aircraft engine is used. Figure 2 shows a basic view of the engine and lubrication flow. This data consists of 13 sensors and five failures accordingly and includes 12,019 observations. As Table 1 shows, the data is well balanced between the failures given that the number of counts for each failure is similar to each other. For the experiments, 30 randomly sampled observations labeled as initial training data. An additional 30 observations were used to test the models. The remaining data were considered as unlabeled data.

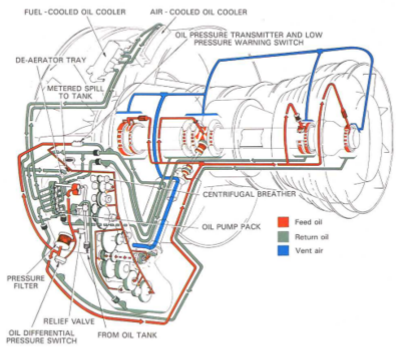

Figure 2. Simple schematic of the lubrication system

Figure 3 uses the Principal Component Analysis (PCA), which is often used to reduce the dimensions(Aggarwal, 2017). This demonstrates the classes are divided into five failures and one nominal state, and in most cases, they are located farther away from each other, and relatively well dispersed. However, there are a handful of cases where there exists an overlap and this is where the model needs to focus more due to its uncertainty.

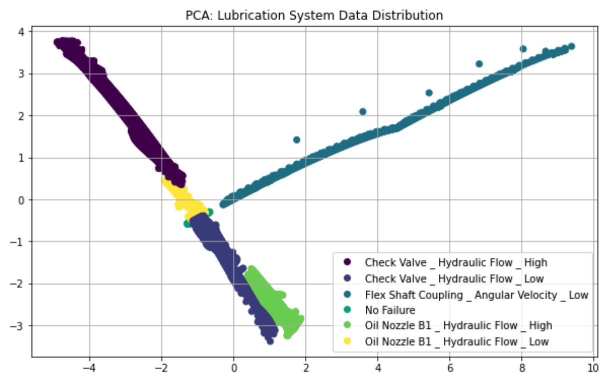

Figure 3. Visualization of the reduced dimension of the lubrication system data

	Total training size (Sample size per failure)		
	42 (7)	60 (10)	90 (15)
Decision Tree	0.794	0.844	0.844
Random Forest	0.794	0.844	0.844
KNN	0.552	0.603	0.681
Gradient Boosting	0.794	0.874	0.807
XGBoosting	0.768	0.855	0.896

Table 2. Performance based on Random Sampling

5.2. Results

5.2.1. Random Sampling

First, Random Sampling assessed model performance by comparing accuracy across various machine learning algorithms with different initial training dataset sizes. A relatively small training dataset encompassing all failures was used.

Based on the PdM data, Table 2 demonstrates the changes in the level of accuracy of each model. Each row represents the different machine-learning models implemented in this case study. Each column shows the total number of trained data—initial training data—and the numbers in parentheses represent the sample size per failure.

In the first two rows, the Decision Tree and Random Forest are examined, respectively. The Decision Tree represents the decisions and corresponding potential results in a tree-like algorithm model, where each internal node and leaf show a decision and an outcome, respectively. Random Forest is an extended version of the Decision Trees by incorporating more samples and additional factors (e.g., the count of votes for each classification) and handling and identifying the important features and their interactions.

Results in the third row represent the model accuracy based on K-Nearest Neighbour (KNN) algorithm, a non-parametric strategy. With the Euclidean distance, KNN searches for the K nearest points to determine the class for the input according to the average label of the neighbors.

	Total training size (Number of Queries)		
	42 (12)	60 (30)	90 (60)
Decision Tree	0.623	0.623	0.623
Random Forest	0.727	0.927	0.999
KNN	0.555	0.58	0.64
Gradient Boosting	0.844	0.886	0.886
XGBoosting	0.844	0.943	0.97

Table 3. Performance based on the Uncertainty-Sampling query strategy

Results based on the Gradient Boosting are presented in the fourth row in Table 2. By training the base model on the residuals, Gradient Boosting (Bentéjac, Csörgő, & Martínez-Muñoz, 2021) applies the predictions based on the weaker models (e.g., Decision Trees) to enhance the performance and handle the complicated associations between features by decreasing the errors of the earlier models. By improving its scalability, speed, and accuracy, Gradient Boosting can be transformed into a higher version of its own, the eXtreme Gradient Boosting (XGBoosting). The last row in Table 2 shows the level of performance based on this XGBoosting algorithm.

In Table 2, regardless of the type of machine learning algorithms, higher accuracy was observed when a large initial sample size was used. This table with Random Sampling serves as the reference (or base model) that can be used to measure the level of improvement by using the query strategies, presented in Tables 3 and 4.

5.2.2. Uncertainty Sampling

Table 3 compares the changes in performance depending on the different number of training data sizes to identify the learning curve for each algorithm. To be aligned with Table 2, the same number of total training sizes for each column were used. The number in parentheses, however, shows the number of queries for each trained dataset, which is different from that of Table 2.

Table 3 demonstrates improved overall performance compared to Table 2 for Random Forest, Gradient Boosting, and XGBoosting, indicating the superiority of Uncertainty-Sampling over Table 2's Random Sampling.

For Random Forest, it reaches up to 92.7 % accuracy with the query strategy only with the 60 number of the total trained dataset; yet, in Table 2, it only achieves 84.4 % with the same amount of training size. For the Gradient Boosting, the accuracy level in Table 3 is consistently higher than that of Table 2 regardless of the size of the data points. With the XGBoosting, while it gets only 85.5% accuracy with the size of 60 in Table 2, Table 3 shows that active learning with an Uncertainty-Sampling query strategy reaches up to 94.3% only with 30 queries. Along this line, with the size of 90 total

	Total training size (Number of Queries)		
	42 (12)	60 (30)	90 (60)
Decision Tree	0.623	0.623	0.623
Random Forest	0.86	0.86	0.86
KNN	0.555	0.58	0.64
Gradient Boosting	0.788	0.622	0.685
XGBoosting	0.839	0.869	0.99

Table 4. Performance based on the Margin-Sampling query strategy

training data points and 60 queries, XGBoosting achieves an accuracy level of 97% and significantly outperforms the Random Sampling in Table 2. Since this level of accuracy in the last column presents sufficient performance, it also suggests that 60 queries are the optimal number of queries to effectively get the results.

However, Decision Tree shows minor improvement in Table 3 due to its sensitivity to data distribution and overfitting with a small dataset. For K-Nearest Neighbors (KNN), it's a suboptimal choice due to computational costs, sensitivity to outliers, and inability to adapt to input distribution changes.

5.2.3. Margin Sampling

In Table 3, the accuracy of each machine learning algorithms based on Margin-Sampling query strategies are shown. As in Table 3, each column represents the total number of trained data and the numbers in parentheses show the number of queries used. For XGBoosting and Random Forest, the levels of accuracy are enhanced than those results based on Random Sampling (shown in Table 2). Random Forest with Margin Sampling consistently shows 86.0% as the accuracy level regardless of the number of queries used. XGBoosting with a Margin-Sampling query strategy reaches 86.9% with 60 trained data and 30 queries. With 60 queries, its performance improves up to 99.0%, demonstrating that it outperforms the Random Sampling in Table 2.

Random Forest and XGBoosting show better results with Margin-Sampling compared to Random Sampling. This is likely due to Random Forest's Majority voting and XGBoosting's tree pruning, which mitigate overfitting (Chen & Guestrin, 2016). Margin Sampling's diverse selection and noise handling then improve performance further by assessing data uncertainty and extracting valuable patterns.

Similar to results based on Uncertainty Sampling (shown in Table 3), Margin Sampling in Table 4 does not improve the accuracy level of KNN and Decision-Tree algorithms compared to Random Sampling. In addition, this table shows the possibility where even Gradient Boosting may underperform compared to one Random Sampling. All these under-performances are expected due to the possibility of over-fitting. Since the query strategy continuously selects the most uncertain data, all learners only recognize the data points with high uncertainty without considering the distribution for each failure. This may lead to misrepresentation of the data distribution, while Random Sampling in Table 2 is suitable to learn the overall distribution of the data for each failure.

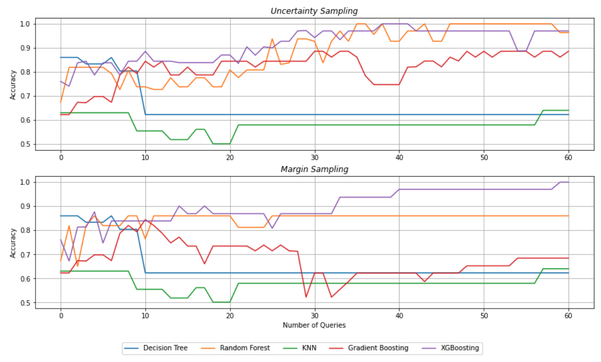

Figure 4. Performance Comparison between Query Strategies

5.2.4. Query Strategy Comparison

Examining two distinct query strategies, Uncertainty Sampling and Margin Sampling, Figure 4 compares their performance levels. The top graph represents the changes in performance for the Uncertainty Sampling, and the bottom one is for the Margin Sampling. These two graphs mainly illustrate the possibility that the learner's results can be different depending on which query strategy is implemented.

In both graphs, the performance level for KNN and Decision Trees are relatively lower compared to XGBoosting and the Random Forest across the different number of queries (when it is higher than 10). This is consistent with what was observed in Table 3 and Table 4 above. Unlike the KNN and Decision Trees, XGBoosting and Random Forest show proportionally-increased performance as the number of queries increases for both Uncertainty-Sampling and Margin-Sampling strategies. This is because XGBoosting and Random Forest are less likely to over-fit than other machine learning algorithms, ultimately helping both query strategies to select more informative data points than Random Sampling.

When comparing two different query strategies *per-se*, it was observed that Gradient Boosting with Uncertainty-Sampling produces results with higher accuracy than Margin Sampling. This difference arises because of the characteristics of the dataset. While Margin Sampling works well with highly imbalanced data, the Uncertainty-Sampling query strategy is more suitable with well-balanced data that have relatively straightforward decision boundaries. Therefore, as demonstrated in section 5.1, our PdM dataset, which is highly balanced with a reasonable number of classes, is expected to perform better with Uncertainty Sampling.

To summarize, depending on the type of datasets used, the optimal query strategies and machine learning algorithms may all vary; however, given that the optimal active-learning methods were chosen, this substantially improves the performance compared to simply using Random Sampling.

6. Conclusion

This paper extensively explores the workflow of active learning from scenarios to query strategies and conducts the case study by implementing two query strategies (i.e., Uncertainty Sampling and Margin Sampling) with PdM data. By comparing them with the results from Random Sampling, it is optimal active learning methods could enhance the performance even with the relatively smaller size of the data. While there are a plethora of previous papers that examine active learning methods, our paper specifically examines the performance of the two query strategies and contributes to the literature by applying the well-balanced PdM data that has not been extensively used with the active-learning strategies. As this data is well-balanced and is with relatively simpler decision boundary, for future works, more query strategies (e.g., QBC or Diversity Sampling) can be applied to this data to generalize the performance of the active learning methods. Also, if researchers further compare the results of various query strategies between balanced and imbalanced data, this type of future work can be promising.

Acknowledgment

We thank Dr. Jacek Stecki for initiating the Syndrome Diagnostic project which spans various areas of research such Active Learning. We would also like to mention the data science, development and engineering teams for their input and aid in writing this paper.

References

Aggarwal, C., C. (2017). Principal component analysis. In *An introduction to outlier analysis* (p. 75-87). doi: 10.1007/978-3-319-47578-3

Aydin, G., Hallac, R., I., & Karakus, B. (2015). Architecture and implementation of a scalable sensor data storage and analysis system using cloud computing and big data technologies. *Hindawi, Journal of Sensors*.

Balcan, M., Broder, A., & Zhang, T. (2007). Margin based active learning. In *Learning theory* (Vol. 4539). doi: https://doi.org/10.1007/978-3-540-72927-3$_5$

Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, *54*, 1937–1967. doi: https://doi.org/10.1007/s10462-020-09896-5

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In (p. 785–794). doi: 10.1145/2939672.2939785

L'Heureux, A., Grolinger, K., Higashino, W. A., & Capretz, M. A. (2017). A gamification framework for sensor data analytics. In *2017 ieee international congress on internet of things (iciot)* (p. 74-81). doi: 10.1109/IEEE.ICIOT.2017.18

Roan, M., Erling, J., & Sibul, L. (2002). A new, non-linear, adaptive, blind source separation approach to gear tooth failure detection and analysis. *Mechanical Systems and Signal Processing*, *16*(5), 719-740. doi: https://doi.org/10.1006/mssp.2002.1504

Rosenthal, S., & Dey, A. K. (2010). Towards maximizing the accuracy of human-labeled sensor data. In *Conference: Proceedings of the 2010 international conference on intelligent user interfaces.*

Settles, B. (2009). In *Active learning literature survey.*

Teh, H., Kempa-Liehr, A., & Wang, K. (2020). Sensor data quality: a systematic review. In *J big data 7.* doi: 10.1186/s40537-020-0285-1

Wen, C., B., Xiao, Q., M., Wang, Q., X., Zhao, X., Li, F., J., & Chen, X. (2021). Data-driven remaining useful life prediction based on domain adaptation. In *Peerj. computer science, 7.* doi: 10.7717/peerj-cs.690

Woodward, K., & Kanjo, A. e. a., E.and Oikonomou. (2020). Labelsens: enabling real-time sensor data labelling at the point of collection using an artificial intelligence-based approach. In *Pers ubiquit comput 24,.* doi: 10.1007/s00779-020-01427-x

Biographies

Y. Jun You-Jung Jun is a data scientist at PHM Technology. Prior to this, he worked at FoodLegal as a machine learning engineer. He received a bachelor's degree in industrial engineering from Soongsil University in Seoul, South Korea, and a master of information technology degree from the University of Melbourne in Melbourne, Victoria, Australia.

N. Zaman Navid Zaman is a master of electrical engineering graduate from the University of Melbourne, Australia since 2020, where he focused on signals, systems and control theory. He has interned at Outotec Ausmelt for a few months before joining PHM Technology as lead data scientist. He has co-authored a RAMS paper previously, centered around the causation-based AI tool, Syndrome Diagnostics.

D. Chan Daniel Chan is the COO and Engineering Manager at PHM Technology. Since graduating as a systems engineer in 2016, Daniel has been involved with various projects across multiple industries including automotive, industrial engineering and aerospace and defense. His interests are in the design of optimized model-based processes and methodologies to enable efficiency and consistency for engineering analyses.