Active Learning Strategies to Reduce Anomaly Detection False Alarm Rates

Adwait Sahasrabhojanee [1] **David L. Iverson** [2] **Shawn R. Wolfe** [2] **Kevin M. Bradner** [2] **Nikunj C. Oza** [2]

Abstract

A significant focus within the commercial aviation community is the discovery of unknown safety events in flight operations data. The rule-based methods which are currently used in aviation are only able to look for known issues. Data-driven unsupervised anomaly detection methods are better at capturing unknown safety events; however, they have a higher false-alarm rate—the statistical anomalies that they identify are often not operationally significant (i.e., safety concern). Subject Matter Experts (SMEs) must spend significant time reviewing these statistical anomalies individually to identify a few operationally significant ones. Our previous work (Sharma et al., 2016; Das et al., 2017) developed an active learning algorithm that builds a classifier to distinguish uninteresting anomalies from operationally significant anomalies by obtaining labels from SMEs and their rationales for choosing the labels. In this paper, we apply our past work to multi-Unmanned Aerial Vehicle (UAV) missions. However, the active learning strategy previously used is inadequate for our chosen setting. Therefore, we experiment with different active learning strategies. We describe how we went about developing our final chosen strategy and show its superiority in experimental results over two baseline strategies including the cited previous work.

1. Introduction

New technologies are being developed to handle complexities of the Next Generation Air Transportation System (NextGen) because of the expected increase in air traffic and the increased use of UAVs. This means that we need to identify and address both current and future safety concerns along with the operational, environmental, and efficiency issues within the National Airspace System (NAS). The airline industry and others involved in commercial aviation primarily look for known examples of anomalous operations in the form of exceedances. These are rules that specify thresholds over several variables which, if violated, trigger the exceedances. In many cases, information about the exceedance's severity is also included. However, by definition, exceedance-based methods cannot find previously unknown anomalies. Additionally, developing and refining these exceedance-based systems to include UAV traffic as its volume and density increase would require excessive domain expert time.

The National Aeronautics and Space Administration (NASA), the Federal Aviation Administration (FAA), and industry have been developing new technologies to identify previously undiscovered safety events by mining large heterogeneous aviation data sets that are continually collected. In particular, our Data Sciences Group has developed several machine learning methods for anomaly detection, e.g., (Iverson, 2004; Das et al., 2010; Janakiraman, 2018) and tested them on aviation data. These methods require significantly less domain expert time to develop, and are complementary to exceedance-based methods, in that they are expected to have a lower missed detection rate, but a higher false alarm rate. Because of the high safety of the aviation system, it is reasonable to assume that all statistically normal data represent normal operations, and all operationally significant anomalies fall within the statistically anomalous data. Though this implies that among the statistical anomalies there are previously-unknown operationally significant anomalies waiting to be found, it is also likely that among the statistically anomalous data, not all data are operationally significant. A high false alarm rate leads to the anomaly detection system being ignored. SMEs do not have the time to label all statistically anomalous data as operationally significant (OS) or non-operationally significant (NOS). Therefore, we developed an active learning algorithm that accepts SME labels with rationales that classifies the statistically anomalous data points as either OS or NOS (Sharma et al., 2016; Das et al., 2017). Note that

[*]Equal contribution [1]Universities Space Research Association (USRA) National Aeronautics and Space Administration (NASA), Moffett Field, CA 94035, USA [2]National Aeronautics and Space Administration (NASA), Moffett Field, CA 94035, USA. Correspondence to: Adwait Sahasrabhojanee <adwait.sahasrabhojanee@nasa.gov>, Nikunj C. Oza <nikunj.c.oza@nasa.gov>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 108, 2020.

this approach does not require modifications to the anomaly detection model, and so can be applied to any anomaly detection algorithm. The rationales are used as additional features beyond those used in the anomaly detection. We applied our algorithm to radar track data consisting only of commercial air traffic and only used one active learning strategy—the most likely positive (MLP) strategy, which always selects the data point classified as OS that is the furthest away from the OS/NOS decision boundary.

We attempted to use our algorithm with data from simulated multi-UAS missions as part of one of our projects. As part of this, we tested the anomaly detection/active learning approach and found that the MLP strategy did not always work well. For the past work, MLP was a reasonable strategy because of the very small number of anomalies—MLP is a way of oversampling the minority class to address the class imbalance and, intuitively, for their problem of identifying safety anomalies, it was important to have SME's spend their time labeling potential OS anomalies. However, we thought that the larger fraction of anomalies in our problem led to MLP being less effective. This motivated us to add and develop multiple active learning strategies and experiment with various methods by which to choose the strategy. In this paper, we describe these strategies and methods and their results. In Section 2 we describe our project and the simulated data that we generated. In Section 3, we describe our anomaly detection and active learning pipeline, and also describe the active learning strategies that we developed. In Section 4, we describe the results of our experiments. We end with Section 5, giving some concluding remarks and our planned upcoming work.

2. Background

In this section, we first briefly describe the simulated multi-UAS missions that we ran and then describe the data that we produced, including the anomalies that we generated.

The NASA Autonomy Teaming and TRAjectories for Complex Trusted Operational Reliability (ATTRACTOR) project aims to build a basis for certification of autonomous systems by establishing metrics for trust and trustworthiness in multi-agent systems by using natural interaction, explainable Artificial Intelligence (AI), and persistent modeling and simulation, in the context of mission planning and execution with analyzable trajectories (Alexandrov, 2018). In particular, these technologies were implemented in the context of simulated search and rescue missions where we assume a single target that needs to be found (e.g., a hiker who has become lost and perhaps needs medical attention) for which multiple Unmanned Aerial Vehicles (UAVs) are deployed.

Our specific role within the project included implementing anomaly detection and active learning for the simulated multi-UAS missions. The concept of operations is that there are multiple past missions for which we have data and run an anomaly detection method to learn what constitutes normal missions and identify anomalous mission behavior. We assume that, as with typical data-driven anomaly detection methods, many of the identified anomalies are false alarms, in that they are only statistically significant and not operationally significant. Then active learning is used to solicit SME feedback to learn, among the statistical anomalies, which ones are OS and which ones are NOS, and also obtain the SME's rationale for his/her decision. This will reduce the false alarm rate when looking for anomalies in future mission data. Additionally, when new missions are running, if an anomaly appears that is similar to one that an SME has labeled in the past, then the mission managers' display can show the SME's label and rationale as a candidate explanation, which can help the mission manager decide on an appropriate mitigation.

The primary data source used in this work is a search and rescue inspired drone simulation. The simulation was implemented in the Unity game engine, and it provides for a team of drones flying a precomputed set of trajectories (Figure 1). For this paper, anomalies were generated by placing trees in the route of a drone. Failures in the obstacle avoidance algorithm would result in a loss of separation and an eventual collision with the tree. More details on the data will be given in Section 4.

3. Anomaly Detection / Active Learning System

In this section, we first briefly describe sequence of two anomaly detection algorithms that we use, although the active learning component of our system is designed to be agnostic to the anomaly detection algorithm being used. We then describe the active learning algorithm. The complete anomaly detection / active learning pipeline is depicted in figure 2.

3.1. Inductive Monitoring System

We use a sequence of two anomaly detection algorithms in our work. The first is the Inductive Monitoring System (IMS) (Iverson, 2004). It operates in two phases. The first phase is the model-building (i.e., training) phase, where IMS builds a model of nominal operations (figure 2, arrows a, b, c). It takes, as input, numeric data vectors capturing system state; in our case, these are telemetry from the simulated drone, e.g. position, speed, objects detected, and others. IMS clusters these vectors to build its model of nominal operations. IMS does not need any examples of anomalies to create its model. IMS does not need any expert input except for possible adjustment of data vector parameter weights and clustering arguments.

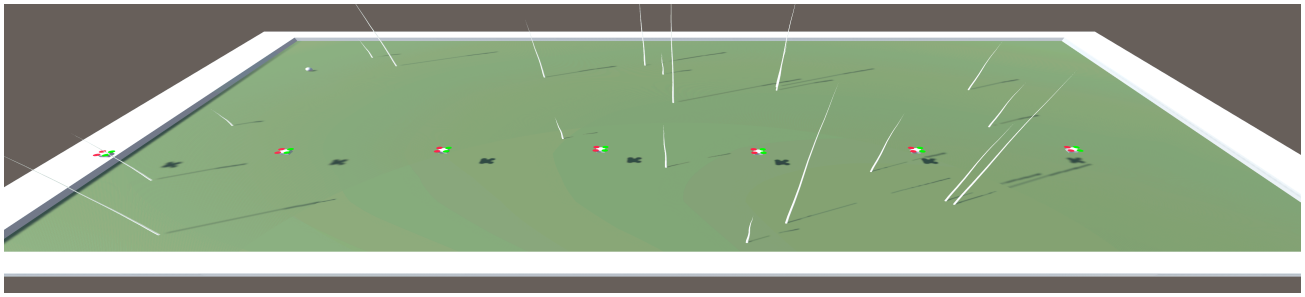

Figure 1. Simulation with seven drones and twenty trees.

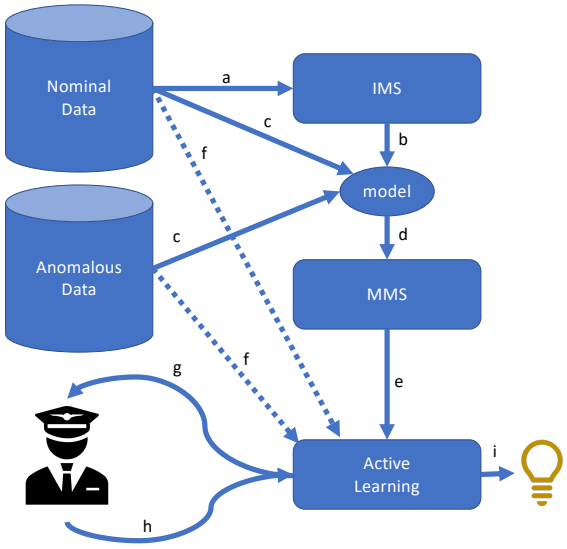

Figure 2. Anomaly Detection Pipeline

The second phase of IMS is the testing phase where it monitors the system for anomalies. In this phase, IMS takes, as input, the same sort of data vectors that it used to build its model, though here anomalous data may also be included. The output is a deviation score, representing how much the system's state deviates from expected behavior. This deviation estimate is the distance of the data vector from the clusters in IMS's model, either the closest point in the nearest cluster or a weighted average distance from the nearest clusters representing a designated number of training data points (k-nearest neighbors). The latter option was chosen for our results. IMS can also produce an analysis of parameter contributions to the deviation score, i.e., which parameters contributed most to the deviation score.

3.2. Meta Monitoring System

The Meta Monitoring System (MMS) is an add-on to IMS, developed to address some difficulties when using IMS. The first issue is interpreting the deviation scores. The deviation scores have a clear relative interpretation, namely that lower deviation scores are less anomalous, but not a clear absolute interpretation. The second issue is fleeting but high deviation scores that often occur from data noise or unusual intermediate states. To address both issues, MMS produces a deviation score, based on IMS's output, that is more easily interpreted.

Like IMS, MMS (figure 2, arrows d and e) has two phases: a model-building training phase, and a monitoring phase. In the first phase, MMS takes IMS's deviation scores as its only input and does not use the original data vectors.

MMS primarily uses deviation scores from nominal data, but unlike IMS, it can also make limited use of results from anomalous data. It builds two models: one of nominal deviation scores and one of anomalous deviation scores, each a probability distribution of deviation scores.

In the second phase, incoming IMS deviation scores are passed to the learned model, and probabilities of producing the observed deviation score are calculated for both models. The probabilities are combined to create an MMS score, which ranges from 0 (definitely nominal) to 1 (definitely anomalous). To convert the MMS score to a binary classification, we use an arbitrary threshold of 0.5 to distinguish nominal from anomalous. When anomalous deviation scores are available, the anomalous distributions are adjusted to maximize performance on the training data, typically for accuracy, but MMS can also be trained to favor false positives or false negatives as desired. MMS can also take as input IMS parameter contributions and translate those to parameter contributions to the MMS score. More details of MMS are outside the scope of this paper.

3.3. Active Learning

IMS and MMS work together to characterize normal performance and identify when anomalies occur. However, there are shortcomings that may arise in practice. First, not all statistical anomalies are operationally significant, and too many false alarms degrade the utility of anomaly detection. This problem is exacerbated when false negatives have been reduced, to avoid missing OS anomalies, at the cost of increased false positives. Second, although IMS and MMS can justify their scores in terms of the observed data, these justifications can be difficult to comprehend. Both of these issues can be addressed through active learning.

In our active learning approach, a two-class support vector machine (SVM) classifier is used to distinguish OS anomalies (true positives) from NOS anomalies (false positives). In the training phase, the active learning module takes input from a subject matter expert (SME) to train the SVM on the original data. For each iteration of the training process, the SME is presented with a data point previously identified by IMS/MMS as anomalous, to be marked as OS or NOS. The SME is required to provide a rationale for his/her label. If the anomaly is OS, the SME also marks a time window in that anomaly that shows exactly where in the data (flight) the anomaly occurred. The rationale is parsed and is used along with the data in the time window to build new features that are added to the data. Then, the active learning module retrains its classifier, using this new truth value along with all data gathered so far. It repeats this process until a budget is reached, typically constrained by the SME's availability. Normally, only a small fraction of the data need be classified to achieve excellent performance. The rationales provided by the SME can also be reused as explanations for similar instances.

The classifier has to be initialized with some labeled examples first with a healthy mix of different types of OS and NOS anomalies, which we attempt to achieve by means of one of the strategies detailed below. We first use the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm (Ester et al., 1996) to cluster the statistical anomalies found by IMS/MMS based on the percentage contribution of each variable to the event being an anomaly. The anomalies can be clustered based on any variables that can describe an anomaly in a continuous space. Thus, similar anomalies would be clustered together, with any anomalies that do not fall into any of those clusters being put in a separate 'outlier' group. If there is a large class imbalance, the outlier group was observed to often be populated mostly with the minority class.

After clustering, we use our domain knowledge of what kind of OS anomalies can happen and how common OS anomalies are to pick one of the following initial sampling strategies:

1. DBSCAN outliers only:

   Pick the initial set only from the outlier group, which has points that DBSCAN could not put into any cluster and which are thus outliers in the data based on the DBSCAN parameters. This is useful when most OS anomalies look different from each other, while NOS anomalies sometimes look similar to each other and sometimes different. Some of the outliers would then be NOS and some would be OS. If OS anomalies are not significantly less frequent than NOS anomalies or especially if NOS anomalies are a little less frequent than OS anomalies, it might be worthwhile to use the strategy "DBSCAN Clusters and Outliers" below.

2. DBSCAN Clusters Only:

   Pick the initial set only from points that DBSCAN could put into a cluster. This is useful when OS anomalies as well as NOS anomalies can generally be grouped into clusters because of similar behavior. So some of the clusters are OS clusters and some are NOS clusters, and therefore both get represented in the bootstrap set.

3. DBSCAN Clusters And Outliers:

   Pick one point from each cluster, and then pick as many outlier points as there are clusters. This is useful when NOS anomalies mostly have other NOS anomalies similar to them and thus can generally be put into clusters, while OS anomalies are generally different from each other. So the points picked from the clusters are generally NOS, and the points picked from the outliers are generally OS.

We chose a baseline set of data with 15 examples using the "DBSCAN Clusters and Outliers" strategy, since that is most appropriate given our data, and trained with these. After this training, we selected individual data points for SME labeling using combinations of the following active learning strategies:

- Most Likely Positive (MLP): Pick the data point that is furthest from the decision boundary on the OS side.
- Most Likely Negative (MLN): Pick the data point that is furthest from the decision boundary on the NOS side.
- Least Likely Positive (LLP): Pick the data point that is closest to the decision boundary on the OS side.
- Least Likely Negative (LLN): Pick the data point that is closest to the decision boundary on the NOS side.
- Check for Misclassified OS (CM_OS): If the model changed the label of more than one anomaly to OS during the previous update, then build a list of anomalies to check for potential misclassification and pick the first one. The list is made up of all the labels that have been changed relative to the last iteration, starting from the point closest to the boundary and moving outwards.
- Check for Next Misclassified OS (CNM_OS): Pick the next potentially mislabeled OS anomaly to check in the list.
- Check for Misclassified NOS (CM_NOS): If the model changed the label of more than one anomaly to NOS during the previous update, build a list of anomalies to check for potential misclassification and pick the first one. The list is made up of all the labels that have been changed relative to the last iteration, starting from the point closest to the boundary and moving outwards.
- Check for Next Misclassified NOS (CNM_NOS): Pick the next potentially mislabeled NOS anomaly to check in the list.

The data used in (Sharma et al., 2016; Das et al., 2017) had an extreme class imbalance: OS anomalies were extremely rare. Therefore, the MLP strategy was used at each iteration, to increase the probability of an OS point being selected for labeling. Our dataset does not have such an extreme imbalance, causing some unwanted behavior with just using MLP. For example, after a few iterations, a precision of 0.2 and recall of 1.0 was achieved: most of the flights had been classified as OS by the model. In this specific scenario, the classifier had not learned a good bias, even though the Receiver Operating Characteristic (ROC) curve looked relatively good. Because we were using MLP, the model kept picking actual OS points—of which there were many—to be labeled. When the SME labeled these points, the model did not update itself, as it had correctly predicted these labels. If however, at this point, we had used the LLP strategy, the point picked would probably have been labeled NOS by the SME, so the model would have been incorrect, and thus updated. This scenario demonstrates the need for adjusting strategies dynamically for this problem.

Our first thought was that we should simply alternate between MLP and LLP. We quickly saw that this would cause a similar problem: if the precision now was 1.0 and recall was 0.2, the model would keep picking points from the OS side of the decision boundary, which have already been correctly classified. This would again cause the model to stagnate and waste the SME's time. At this point, we decided to test dynamically switching between four strategies: MLP, LLP, LLN, MLN.

We first tried alternating between MLP and MLN. Results improved, but we still noticed a problem: MLP and MLN are both strategies that sample points far away from the decision boundary, so as the model improves, these points are generally the ones classified correctly. As the model keeps asking labels for points it has already correctly classified, it does not update enough for many iterations. This showed us the need to include LLP and LLN in our pool of strategies. Our next few tests all dynamically switched between MLP, LLP, LLN, and MLN: the strategy selected at each iteration being dependent on the strategy at the previous iteration and the label of the point sampled at the previous iteration.

For example, consider an iteration where the previous strategy was MLP, and the SME provided the label "OS" for the point sampled using that strategy. This tells the model that it had correctly predicted that point, and it does not update itself enough. So for the next iteration, we switch our strategy to either LLP or LLN, to maximize our chances of getting an incorrectly predicted point. A variety of different combinations and flows of strategies were tested, and we noticed that often when the model updates itself, it mispredicts some points that were previously predicted correctly. This led us to add a final few strategies to our pool of strategies.

Take for example an iteration where the previous strategy was MLP, and the SME provided the label "NOS" for the point sampled using that strategy. This tells the model that it had previously mislabeled the point, and the model updates itself. We noticed—with multiple datasets—that when the model corrected itself such that this newly labeled point now lay on the NOS side of the decision boundary, the model also often ended up switching the labels of some actually OS flights to NOS. The new strategies we added check for such potentially misclassified points when one or more points switch labels. The final strategy we ended up with was extensively tested on multiple left-out datasets,

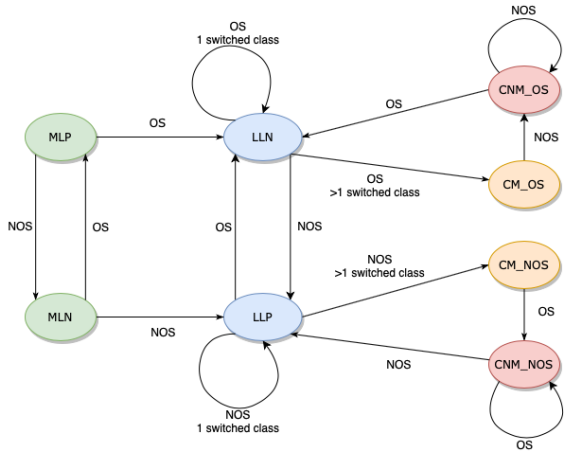

Figure 3. Dynamic switching of Active Learning sampling strategies. The texts by the arrows are the label provided by the SME in the last iteration, and the arrows show the progression of strategies depending on the previous label, the previous strategy, and, in some cases, the number of other examples for which the label got switched due to updating the classifier.

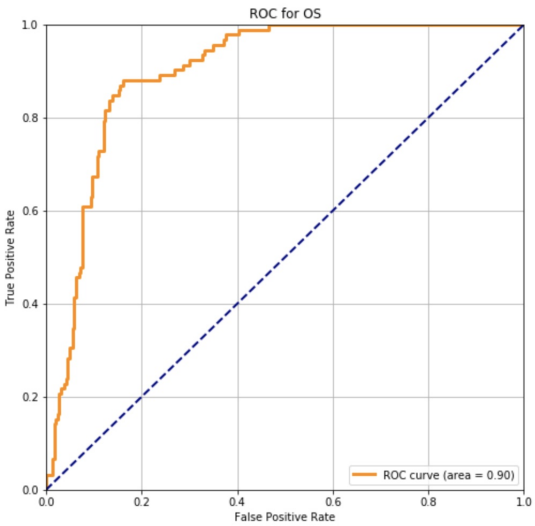

Figure 4. ROC curve for the active learning strategy of always choosing the most uncertain point for SME labeling

and was observed to get consistently good results. Figure 3 shows the final flow of strategies that was selected.

4. Results

In this section, we experimentally compare our dynamic active learning strategy selection method with baseline methods. We first describe the simulated data that we collected and then describe our experimental results.

4.1. Data

We used the Unity simulation described in Section 2 to generate the data to train and evaluate our anomaly detection and active learning methods. We performed 586 simulations total. Within these simulations, we varied the size of the search area, number of drones, and individual drone operating characteristics to reduce the chance of our algorithms making their inferences based on trivially memorized details. The drones explore the search area in a "lawnmower" pattern, in which each drone moves across the environment in long, alternating lines over the search area, shifting a little at the end of each line. As they move, these drones also search for a target that moves slowly along the ground. Should a drone fly over the target, it completes its search-and-almost-rescue mission and falls to the ground. Within the 586 simulations, 204 simulations were held out for testing and 382 were used in training. Within the test set, half had OS anomalies and half had NOS anomalies. Within the 382 training simulations, 92 had OS anomalies and 290 had NOS anomalies.

4.2. Experimental Results

We first compare our dynamic selection strategy with the most typical active learning strategy used for classification problems, which is to choose the data point for which the model is most uncertain, which is often the data point closest to the boundary between classes. This strategy is often not suitable for anomaly detection because of class imbalance and because of different misclassification costs (false alarms are normally much less "costly" than missed detections). Nevertheless, choosing the most uncertain point is used frequently enough to be worth using as one baseline. For this experiment, we used each strategy (baseline most uncertain point strategy vs. our dynamic strategy) to choose 33 flights for an SME to label and derived the ROC curve for each case.

Figure 4 shows the ROC curve for the baseline strategy. The area under ROC curve is 0.9. Figure 5 has the ROC curve for our dynamic strategy, for which the area under ROC curve is 0.98. The baseline strategy did what we expect, which is to initially learn a classifier and then update it very slowly with each additional labeled point.

We now describe a more detailed comparison of our method against the MLP strategy that was used in the past work (Sharma et al., 2016; Das et al., 2017) that inspired this work.

In all of our learning curves, the x-axis gives the number of data points labeled by the SME while the y-axis shows the precision and recall. Figure 6 shows the learning curve for the MLP strategy and figure 7 shows the learning curve for our dynamic strategy for up to 33 SME-labeled examples

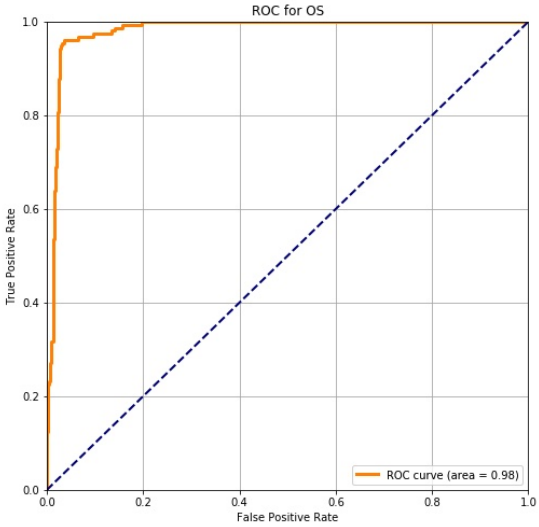

Figure 5. ROC curve for our dynamic active learning strategy

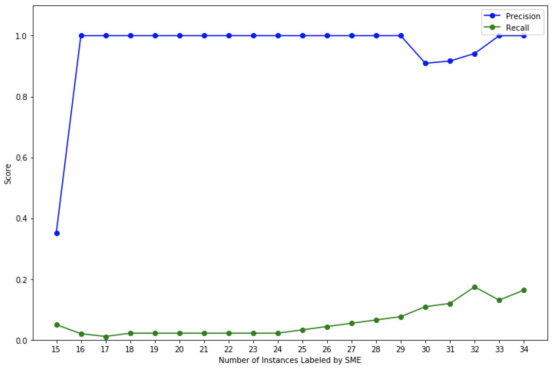

Figure 6. Learning curve for the active learning strategy of always choosing the most likely positive (MLP) point for SME labeling

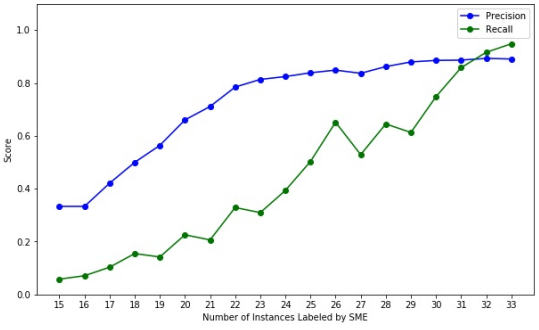

Figure 7. Learning curve for our dynamic active learning strategy

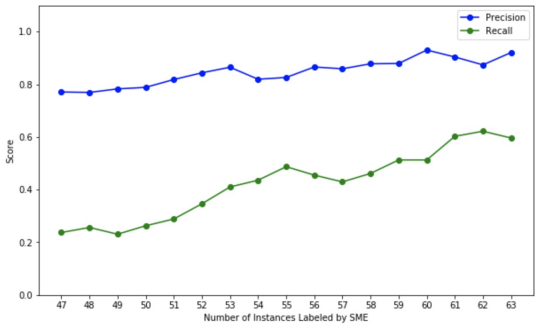

Figure 8. Learning curve for our dynamic active learning strategy but where the SME provides labels and builds rationales based only on the full flight's data rather than identifying the window within the flight for which the label and rationale applies.

where the SME labels flights as OS or NOS and, for the OS flights, provides rationales for the window within the flight when the OS anomaly occurred, if it was OS. The MLP strategy worked largely as we expected—the SVM latched onto one model and largely gave the SME examples for which it had already guessed the correct label and then updated the model very slowly, leading to little change in the performance with additional examples. The dynamic strategy was notably better, especially in recall. We decided to experiment with limiting the SME's effort further and only having him/her provide labels and rationales based on the full flight's data rather than just for a particular window that the SME identifies. Figure 8 shows the learning curves for this case. The results are clearly inferior to the case where more targeted labels and rationales are provided.

5. Conclusions

In this paper, we developed active learning strategies to augment anomaly detection for simulated multi-UAS missions and demonstrated the benefit of varying active learning strategies. We are currently working to transition this work toward real multi-UAS flight test data that we did not produce. Obviously, such data are more realistic. However, since these are controlled tests, they are less likely to have OS anomalies in them and are; therefore, more difficult to test. We will also work to implement the approach described in the introduction to find, in real time, cases where the data are close to anomalies that SMEs have labeled and provided rationales for, and display these labels and rationales to the mission manager to help in choosing mitigations. We are additionally working to implement our system for use by airline safety analysts. Additional future work includes allowing for integrating multiple data sources and allowing for multiple domain experts to label anomalies, reconciling disagreements among these experts, and learning these experts' biases and tendencies to choose the best expert(s) to

label each anomaly.

6. Acknowledgements

We thank Bryan Matthews and Daniel Weckler for their help with the previous work's code upon which we built. We thank the NASA Aeronautics Research Mission Directorate's (ARMD) Convergent Aeronautics Solutions (CAS) program and System-Wide Safety (SWS) program, as well as the Ames Research Center's Center Innovation Fund (CIF) program for funding the work described in both this and the previous work.

References

Alexandrov, N. Trust, trustworthiness, and risk in autonomous decision making. In *AIAA AVIATION 2018, Atlanta, Georgia, USA*, 2018.

Das, K., Avrekh, I., Matthews, B., Sharma, M., and Oza, N. Ask-the-expert: Active learning based knowledge discovery using the expert. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery (ECML-PKDD)*, 2017.

Das, S., Matthews, B. L., Srivastava, A. N., and Oza, N. C. Multiple kernel learning for heterogeneous anomaly detection: algorithm and aviation safety case study. In *Proceedings of KDD*, pp. 47–56, 2010.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. 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)*, pp. 226–231. AAAI Press, 1996.

Iverson, D. L. Inductive system health monitoring. In *In Proceedings of The 2004 International Conference on Artificial Intelligence (IC-AI04), Las Vegas*. CSREA Press, 2004.

Janakiraman, V. M. Explaining aviation safety incidents using deep temporal multiple instance learning. In Guo, Y. and Farooq, F. (eds.), *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pp. 406–415. ACM, 2018. doi: 10.1145/3219819.3219871. URL https://doi.org/10.1145/3219819.3219871.

Sharma, M., Das, K., Bilgic, M., Matthews, B., Nielsen, D., and Oza, N. Active learning with rationales for identifying operationally significant anomalies in aviation. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery (ECML-PKDD)*, 2016. URL http://www.cs.iit.edu/~ml/pdfs/sharma-ecmlpkdd16.pdf.