Anomaly Detection with Sensor Data for Distributed Security

Brian Quanz, Hongliang Fei, Jun Huan, Joseph Evans, Victor Frost, Gary Minden, Daniel Deavours,
Leon Searl, Daniel DePardo, Martin Kuehnhausen, Daniel Fokum, Matt Zeets, Angela Oguna
Information and Telecommunication Technology Center, University of Kansas, Lawrence, KS 66045
Contact Email: {bquanz, hfei, jhuan}@ittc.ku.edu

Abstract—**There has been increasing interest in incorporating sensing systems into objects or the environment for monitoring purposes. In this work we compare approaches to performing fully-distributed anomaly detection as a means of detecting security threats for objects equipped with sensing and communication abilities. With the desirability of increased visibility into the cargo in the transport chain and the goal of improving security, we consider the approach of equipping cargo with sensing and communication capabilities as a means of ensuring the security of the cargo as a key application. We have gathered real sensor test data from a rail trial and used the collected data to test the feasibility of the anomaly detection approach. The results demonstrate the effectiveness of our approach.**

I. Introduction

We consider the problem of fully distributed security for sensor networks, and particularly the application to objects equipped with sensing and communication abilities. Already embedded radio-frequency identification tags are becoming ubiquitous. With ever-improving technology, it is not too much of a leap to expect to see tiny embedded sensing devices in many physical objects in the near future. Here we consider the idea of embedded security sensing systems associated with individual objects, capable of determining the objects' "safety" at any given time [1], for instance ensuring hazardous chemicals are properly handled [2] and ensuring health and safety compliance for industrial workers [3]. In particular, as a driving application, we consider transport chain security, embedding sensor nodes into objects such as cargo in the transport chain, forming a network of sensor nodes. We address this problem by anomaly detection. An anomaly or outlier in a set of data is defined as an observation that deviates from historical patterns.

Traditional signature based automatic detection methods have been widely used in anomaly detection systems. However, signature based methods suffer from their inability to detect new types of attack. Furthermore the database of the signatures grows very large as new types of attack are detected, which may affect the efficiency of the detection.

Data mining based methods have also been well investigated for anomaly detection. Based on whether data samples are labeled or not, these method fall into two categories: unsupervised anomaly detection, such as clustering [4], [5] and supervised anomaly detection, such as one class support vector machine [6], [7] and artificial immune system [8], [9]. In unsupervised methods, data samples have no labels and the key assumption is that normal data instances belong to large and dense clusters, while anomalies do not belong to any significant cluster. In supervised techniques, given a set of normal data samples for training and a set of new samples for testing, the goal is to determine whether the test data belongs to normal or anomalous patterns.

Maintaining security for groups of objects embedded with sensors, processors, and communication capabilities also represents a task of anomaly detection, in this case fully distributed anomaly detection since each object must be able to maintain its own security. In this paper we investigate several approaches for performing fully distributed anomaly detection and particularly examine their application to the task of transportation chain security. The goal behind our approach is to use anomaly detection techniques for each individual sensor node to predict threats such as theft. The algorithms used should be able to automatically learn the "normal" concept, since coming up with rules from the perspective of the particular set of sensors would be difficult and dynamic. Secondly, online learning is desirable to allow real-time training of the detectors and to avoid expensive data collection. Furthermore, sensor nodes typically have limited resources in terms of memory and processing power, so algorithms that can handle resource constraints and operate in an online training fashion are preferable. Also, to allow real-time operations the algorithms should have the capability to continuously learn and adapt while running (online), to account for concept drift present in many real systems.

Additionally, while still allowing fully distributed security, we want to utilize sharing of information between nodes in the same situation, for instance, sensor nodes associated with cargo in the same container in the transport chain with capabilities to communicate with each other. Thus we form predictions for the network, or group, of sensor nodes by sharing opinions with nodes in the same group through communication. The security is still fully distributed since each node can make predictions of its own and send alarms, its predictions are only influenced through communication with its neighbors. We focus here on seeking opinions from neighbors when an anomaly is detected, to keep communication costs down and to help reduce the potentially costly false positives.

We use three base methods for resource-constrained online anomaly detection, a resource-constrained online one-class support vector machine, a real-valued artificial immune system, and a simple feature threshold approach for comparison. We then consider three different ways of forming an overall group prediction in a distributed manner through communications, a baseline approach of individual detections, an event passing approach and an anomaly indication passing approach. In addition we describe our data collection of real sensor data from a short haul rail trial, also used to test our general transport security sensor network currently in development, and present and analyze the results of evaluating our anomaly detection approaches using the collected sensor data.

II. Related Work

Anomaly detection in data mining for networks is not a new topic. For unsupervised anomaly detection, Loo *et al.* [5] have proposed a cluster based intrusion detection scheme for anomaly detection. A similar approach was also presented in [4], [10] to the same problem. Palpanas *et al.* [11] and Subramaniam *et al.* [12] took a different view and proposed a method using kernel density estimators for online distributed anomaly detection in sensor networks. Supervised anomaly detection, which uses purely normal instances as training data, has been studied extensively in the academic community. Most research in supervised anomaly detection can be considered as building a model and identifying samples that deviate from historical patterns. A comprehensive study was carried out in [13], [14]. The concept of a separating hyperplane [15] and hypersphere [16] motivated the use of support Vector Machines (SVMs). Navia-Vazquez *et al.* [6] and Flouri *et al.* [7] have proposed distributed and incremental techniques for training SVMs in sensor networks. Davy *et al.* [17] and Rajasegarar *et al.* [18] employed online support vector machines for abnormal events detection. Another widely used supervised anomaly detection technique is Artificial Immune Systems [8], [9] (AIS), which model the human immune system by using detector strings modeled after lymphocytes to achieve anomaly detection. Recently, graph models have also been applied to anomaly detection. Hill, Minsker and Amir [19] employed dynamic Bayesian networks to identify anomalies in environmental streaming data. Khanna and Liu [20] used Hidden Markov Models to detect anomalies.

The application of anomaly detection is not constrained in wireless sensor networks, but also in biological networks and social networks. Papadimitriou, Dasdan and Garcia-Molina [21] measured the similarity of consequent web graphs created by search engines to discover anomalies. Overall *et al.* [22] used scan statistics for anomaly detection in Genetic Regulatory networks.

III. Methodology

As mentioned in the introduction, our approach to the distributed anomaly detection task consists of base anomaly detection methods combined using different methods through communication with neighboring objects in the network, to form an anomaly prediction. The three base resource-constrained, online anomaly detection algorithms we use are described in section III-B below and the group methods are described in section III-C.

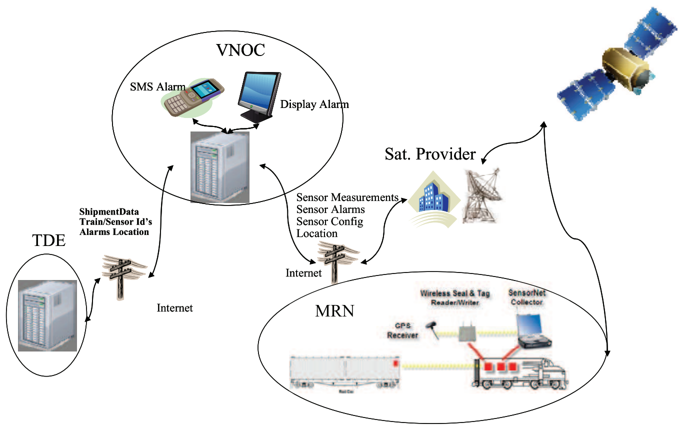


Fig. 1. Transportation Security SensorNet Implementation

This distributed anomaly detection fits into a larger network architecture being developed by the SensorNet group at the University of Kansas for the target application of transport chain security, the current implementation of which is depicted in Figure 1. Here cargo monitoring operates over a mobile rail network (MRN) which in turn communicates with a virtual network operations center (VNOC) which handles transmitting alarm events and interfacing with the trade data exchange (TDE) which supplies such information as shipment data. The sensor anomaly detection fits into the sensor level of the MRN, responsible for detecting events which can then be reported and handled by higher levels, aside from any local action.

A. Extracting Event Vectors

In order to use the learning algorithms for anomaly detection, it is first necessary to extract some type of features from the streaming sensor data. We use the simple but effective approach of piecewise aggregate approximation (PAA) [23], for each sensor. The streaming data is divided into frames, and the dimensionality reduced by taking the mean value in a given frame. In order to capture events forming patterns over time, we form vectors for each sensor combining the new frame with some number of previous frames. We combine the vectors for each sensor by concatenation to form one long event vector.

B. Brief Description of Base Anomaly Detection Algorithms

Here we present a brief description of the three anomaly detection algorithms used for individual objects.

1) Online One-Class Support Vector Machine: The first base detection method we use is an online version of the one-class support vector machine (OCSVM). The OCSVM estimates the support of the training data distribution by finding a hyperplane separating the training data from the origin trading off maximum offset, training error, and function regularization [15]. With a proper kernel function, $K(\mathbf{x}, \mathbf{x}') : \mathcal{X} \times \mathcal{X} \rightarrow R$, representing a non-linear mapping of the training data ($\mathbf{x} \in \mathcal{X}$

) to a feature space, the OCSVM computes a boundary around the training data. The penalty given to a training error in the objective function is weighted proportionally to $1/\nu$. This key tuning parameter ν controls the outlier penalty, and can be seen as controlling the fraction of outliers from the training data.

An online OCSVM (OOCSVM), works by testing each new training instance with the current model; if it is misclassified than it is used to update the current model along with the stored training instances [24], [17]. To form the resource-constrained version, we only keep a maximum number of training instances as resources allow. When more instances are received than the maximum, the instance farthest from the decision boundary is removed. We can make OOCSVM adaptable [1] by giving a fixed lifetime to stored vectors or by training over a shifting window with incremental learning [25].

2) Online Real-Valued Artificial Immune System: For the second base detection method, we extended a real-valued artificial immune system (AIS) approach to an online setting. A key component of the AIS is negative selection; the goal is to define nonself (anomalies) by randomly generating a coverage of detectors around the normal, *self* region of the event space. Random detectors are generated and those that match with self data points (training data) are deleted, so that over time a coverage is formed. Detectors are real-valued vectors with associated radii, i.e. a hypersphere, and a match consists of a data point being within a distance from the detector less than its radius. We have extended the real-valued negative selection to an online learning algorithm. Overall, this approach incorporates aspects of real-valued negative selection [26] and the online components of an AIS framework [9]. To handle resource constraints in this online setting, we limit the total number of active and candidate detectors, and update existing detectors by forming replacement candidates by combining nearest-neighbors in order to improve coverage. Other modifications include online updating of the hypercube boundary and normalization factors with corresponding shrinking and shifting of existing detectors, and shifting detectors away from a matched self-points. We also include a self-radius fraction (SRF) to allow tuning between false alarm rate and detection rate. When the SRF is in $(0,1)$, a negative self-radius is added to the self points, and when the SRF is greater than 1, a positive self radius is added to self points. We can also make the AIS adaptable by giving detectors finite lifetimes through a probability of deletion at each iteration, so that the coverage slowly changes over time with concept drift, with new detectors replacing expired ones [9].

3) Feature Threshold Method: The third detection method is a simple threshold approach, where the max. and min. value for each feature in the training data is stored, and an anomaly is detected if a value is exceeded in the test instance. Additionally, tuning between false alarm rate and detection rate is controlled by decreasing or increasing the threshold values.

[1]Note, in our experiments here we do not test the ability to adapt to concept drift, we simply emphasize here that the algorithms have some capability to handle it

C. Group Predictions

In the following we describe the three approaches we use for forming the group predictions from individual detectors.

1) Group OR: This method describes uses individual detectors for each node to make an anomaly prediction and whenever an individual node detects an anomaly it is counted as a detection for the whole group (i.e., each node independently sends an alarm when it detects an anomaly) - no information of opinion is shared between the nodes. In effect this corresponds to an “OR” operation of the current prediction for each sensor node at a given time point.

2) Group Event Passing: This method follows a similar approach as used in [27] for computer network intrusion detection, found to reduce false alarms, performing distributed sequential hypothesis testing. Here when a sensor node detects an anomaly, it sends the event vector for which the anomaly was detected to its neighbors, who test the event vector with their own trained models. When a specified threshold of positive detections over the total number of predictions is reached, the event is considered an anomaly.

This process is similar to a model averaging approach such as bagging (bootstrap aggregating [28]), which reduces prediction error by reducing the variance of predictions. One key difference is here the samples are drawn from separate data sources (sensor nodes), which would decrease the correlation as opposed to resampling from the same set of data. There are some key reasons why we do not combine models from separate nodes on each single node nor use a bootstrap resampling strategy. The main reason is due to resource constraints. The typically small amount of available resources would make it impractical in many cases to store the many models needed and would also increase processing time for both training and testing events since each model would have to be trained and tested. The task itself offers a natural distributed structure and separate sampling sources so it makes sense to exploit this. Additionally, if adaptive versions of the algorithms are used, the different models would eventually converge to be the same on a given node as the learning continued for each model under concept drift, unless some sampling strategy was used so that the models did not all train on the same data.

One problem with this method is that there are often differences in the sensor readings for different sensor nodes, for example from sensor bias. Thus, we also test a second version of this method to account for biases of sensor nodes, using a simple batch-effect removal technique, of mean re-centering. Here the mean vector from the training data is stored by each detector, and the difference of an event vector from the mean vector is passed to neighbors and added to their stored means for testing. For more general differences in distributions we are currently investigating transfer learning approaches, learning with non-identical distributions, as future work.

3) Group Indication Passing: We can bypass the problem of biases in different sensor nodes, and heterogeneous sensors in general, by sharing only a detection indication between nodes. For this method, the individual detection methods for time-series anomaly detection are used for each node, and a message is broadcast to the neighbor nodes when an anomaly is detected. If a threshold fraction of the other nodes (we use the majority vote in our experiments) also detect a change and broadcast messages within a specified time window the node will be reassured and no anomaly reported. Thus this method focuses on anomaly events that target individual objects in the group, for instance theft of a single package.

IV. Experiment

We have collected data sets modeled after transport chain security scenarios using wireless Sun Small Programmable Object Technologies (SPOTs) demo boards as the sensor nodes, and the associated cases together as the objects. A Sun SPOT contains a 3-axis accelerometer, a light sensor and a temperature sensor as well as a 180MHz 32-bit ARM920T core processor with 512K RAM and 4M Flash memory, a 2.4GHz radio with an integrated antenna on the board, and a 3.7V rechargeable, 750 mAh lithium-ion battery. The SPOTs are only a few inches in size, and run a Java Micro Edition. The goal here was to collect sensor data from within a transport chain, allowing us to test as a proof of concept the utility of using multiple sensor nodes to detect abnormal events or states including intrusion and disruption within the transport chain, under the noise and dynamic environment experienced by sensor nodes in the transport chain. Here we focus on the results and data collected for a short haul rail trial. As an initial phase, the two key events of interest we consider are the removal of an object from a group of objects in the same container, and the movement of a container containing a group of objects, corresponding for example to the theft of cargo and theft, or other anomalous events, happening to an entire container of cargo.

A short haul rail trial was conducted to test the TSSN implementation of Figure 1 and to collect the sensor data for evaluating our approach in a rail transport environment. Since our experiments were restricted to an engine car during the trip, we used a cardboard box to represent the container for the objects and grouped the Sun SPOTs together in the box, keeping them from moving around with Velcro tape, since typically cargo will be fixed to the container in some way. We used seven SPOTs to collect data, six were in the box together and experimented on, and the seventh was held separately as a control to gain an idea of baseline noise. The container with the SPOTs was placed on the floor of the engine compartment, near the experimenters traveling in the engine. Periodically throughout the trip, three events were performed in sequence, a SPOT was removed from the box, about ten minutes later, returned to the box, and ten minutes after that, the box was moved. Also during one such box movement event during the trip, the box is rotated by 90 degrees, and during another, each SPOT's orientation is changed in the box. This sequence of three actions was repeated six times throughout the trip. To collect the data for the trip, the SPOTs were programmed to continuously read and aggregate the sensor value for each sensor. The PAA value for a 390 ms frame (see Section III-A) for each sensor was periodically stored to the internal Flash memory, along with a system time stamp.

A. Evaluation

To evaluate anomaly prediction performance for the various methods, we use the standard receiver operating characteristic (ROC) curves. These allow us to evaluate the performance of the algorithms for different parameters, since typically tuning between false alarm rate and detection rate is application-dependent. Here we define an event as one of the object removal or box movement events. For each event, a detection in a time window of 30 seconds around the beginning and ending of the event is counted as a true-positive detection. Only one true positive is counted per anomaly event, since the concern here is whether or not they can be detected. Outside of these defined event periods, any detection is counted as a false-alarm detection. The true positive rate (TPR), also detection rate, is given as the total number of true detected anomaly events over the total number of anomaly events, and the false positive rate (FPR), also false alarm rate, is the total number of incorrect detected anomaly events over the total number of received non-anomaly event strings from the sensor data. ROC curves are then plotted as detection rate vs. false alarm rate for each method as the associated parameter which tunes between false alarm reduction and increased detection is varied. An ideal outcome is achieved at a detection rate of 1 and a false alarm rate of 0, and an ideal curve is one with an area under the curve (AUC) of 1.

B. Experimental Results

Here we show some of the results from applying the anomaly detection approaches described in section III to detect object removals and box movement. We obtained results for several different feature representations, including different numbers and types of sensor values (e.g. individual x, y, and z accelerometer readings and acceleration magnitude). Here we report results for event vectors formed from three consecutive time frames (as described in Section III-A), where each time frame corresponded to a period of 390 ms, for each x, y, and z accelerometer reading, stacking together the vectors for each dimension. We used the data collected during a period of about 45 minutes from the locomotive loading area, up to about 2 minutes before the start of events for online training of the detection algorithms, and the rest of the data for the trip as test data. This resulted in 6459 9-dimensional training event vectors and a little over 15000 test event vectors for each sensor node. Using the recorded time stamps, these vectors were fed in order incrementally to each algorithm.

We generated ROC curves by varying the tuning parameter for each method.

1) Parameters: For the SVM we used a radial basis function kernel ($K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma||\mathbf{x} - \mathbf{x}'||^2)$) and varied the associated parameter gamma approximately incrementally on a $\log_2$ scale from 0.001 to 4.Here we show the results for gamma fixed at 0.001. The tuning parameters were varied to obtain results for 20 parameters, the self-radius fraction for AIS methods varied from 4.4 to 0.4, ν for SVM methods varied from 0.0001 to 0.7, and the threshold multiple for the feature threshold method varied in the same way as the self-radius fraction.

2) Results for Individual and Group Events Combined: First we give the results for detecting both object removal, an event effecting individual objects, and box movement, an event effecting the entire group of objects. Since all the results were similar we only show the resulting ROC curves in this case for the resource-constrained online one-class SVM, the AIS method with a buffer, and the feature threshold method, given in Figure 2. In this case, even the individual detection methods (corresponding to "Group OR") were able to perform very well, achieving a perfect (area under curve of 1) ROC score for all methods, except for the simple feature threshold approach which had a slight false positive rate at perfect detection rate (a FPR of 1.95×10^{-4} corresponding to 3 false positives). The group event passing methods, "Event Passing" and "Event Passing v2" (corresponding to the mean re-centering version), also both allowed perfect performance and were both able to correct the slight false positive rate of the feature threshold approach. The three group methods have overlapping curves in Figure 2. In order to assess what benefit, if any, the event passing methods resulted in, we also performed an analysis of the true positive and false positive reduction. The results are shown in Figure 3 with bar plots, in which the bar plots above the x-axis represent the difference between the FPR for individual detectors ("Group OR") and the FPR for each group method ("Event Passing" on the left and "Event Passing v2" on the right), for each of the 20 parameters. The bars extending below the x-axis show the amount of true positive reduction in the same manner. We found that the event passing approaches typically provided significant reduction of false positive rate, and usually little or no decrease in true positive rate. In fact, for several methods, the group event passing approaches increased the number of parameters for which perfect performance was achieved. The event passing and event passing with mean re-centering respectively increased the number of perfect scores: for non-resource-constrained OOCSVM from 1 to 4 and 4, for resource-constrained OOCSVM from 2 to 4 and 5, for AIS with a buffer there was no change, for AIS without the buffer from 2 to 3 and 2 (no change), and for the feature threshold method from 0 to 1 and 1. By allowing a wider set of parameters to achieve good performance, this increases the usability of the approach. There did not appear to be much difference between the two versions (with and without mean re-centering); the results suggest the distribution difference between neighboring sensors was not significant for this data so the approach to account for it was not necessary in this case. As expected, the group indication methods (tested for time windows of size 30s and 10s, see Section III) performed poorly here since they typically miss the group events. For visual clarity, we do not show the group indications plots here, however we note that in this case the group indication methods also effected the relationship between the change in tuning parameters and the change in FPR and TPR rates; instead of a typical ROC curve, in which parameter variation results in a predictable TPR vs. FPR variation, the points in the ROC space jumped around.

3) Results for Individual Events Only: We also obtained results as in the previous section for only the removal events, for which we expected improved performance of the group indication approach. In this case, the results achieved were similar, so to save space we do not show them here. As before the group event passing methods were able to reduce false positives with little to no effect on true positive rate.

In this case, the group indication methods were able to achieve perfect scores as well, and in general the results formed a typical ROC curve, with FPR and TPR varying as expected with the tuning parameter variation. However there were still some outlying points, points for which the TPR and FPR did not vary as expected when changing the tuning parameters. Additionally the group indication methods did not improve over the base methods in terms of FPR reduction. However, there are still more variations to try with this approach that may improve its usefulness, such as more window sizes voting thresholds as well as different base algorithms, in particular incremental type algorithms that merely measure change with respect to a set number of previous time series entries.

V. Conclusion and Future Work

In this paper, we investigated several approaches to performing fully distributed anomaly detection with sensor data for maintaining the security of groups of smart objects, objects equipped with sensor nodes containing processors, communication capabilities, and sets of sensor. We have particularly focused on the goal of improving transportation chain security via anomaly detection with sensor data. We have evaluated these ideas as a proof of concept on real sensor data collected during a short haul rail transport trial. Experiments on the data set have demonstrated the effectiveness and feasibility of our approach.

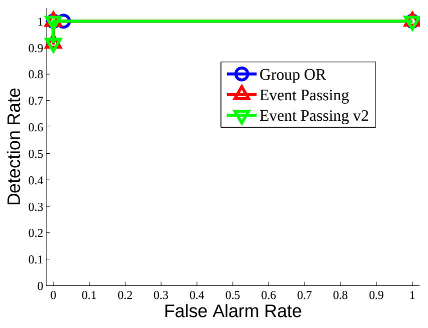
In the future, we will combine individual level and network level anomaly detection together, developing graph-based techniques for the network level anomaly detection. We will also evaluate performance for additional events including unauthorized cargo addition, and explore the adaptability of the proposed approaches under concept drift.

Acknowledgment

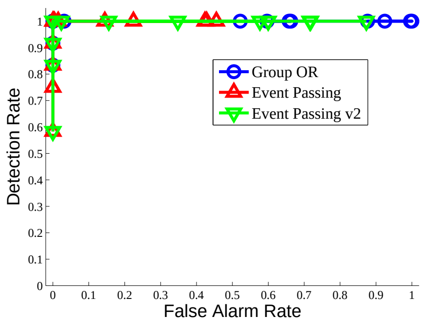
This work was supported in part by the Office of Naval Research under award number N00014-07-1-1042

References

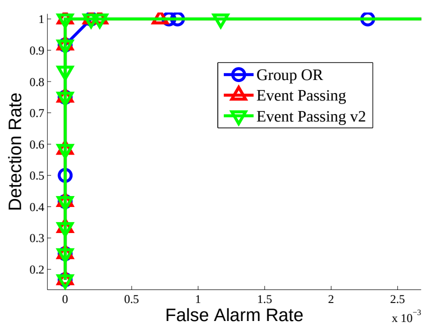
[1] B. Quanz and C. Tsatsoulis, "Determining object safety using a multi-agent, collaborative system," in *Environment-Mediated Coordination in Self-Organizing and Self-Adaptive Systems (ECOSOA 2008) Workshop*, Venice, Italy, October 2008.



(a) resource-constrained OOCSVM

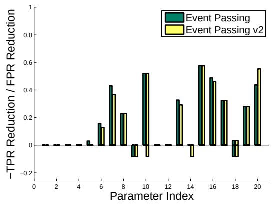


(b) AIS with buffer

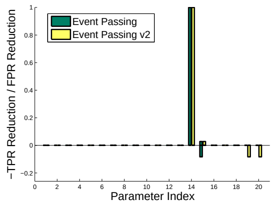


(c) feature threshold method (zoomed in)

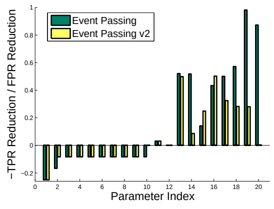
Fig. 2. ROC curves for 3 different base detection methods and all group methods, for both removal and group movement events



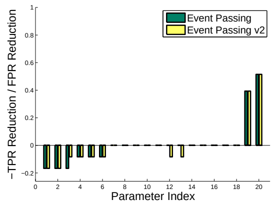
(a) OOCSVM



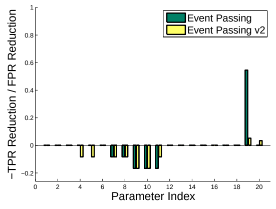
(b) resource-constrained OOCSVM



(c) AIS with buffer



(d) AIS without buffer



(e) feature threshold method

Fig. 3. FPR reduction (extending above x-axis) and TPR reduction (extending below the x-axis) for each base detection method and both group event passing methods, for both removal and group movement events, for all 20 tuning parameters (parameter index is on the x-axis). The y-axis ranges from -0.26 to 1.

[2] D. Dobre and E. Bajic, "Smart object design for active security management of hazardous products," in *1st International Workshop on Design and Integration Principles for Smart Objects in Conjunction with Ubicomp 2007*, Innsbruck, Austria, 2007.

[3] N. Davies, C. Efstratiou, J. Finney, R. Hooper, G. Kortuem, and M. Lowton, "Sensing danger-challenges in supporting health and safety compliance in the field," in *8th IEEE Workshop on Mobile Computing Systems and Applications (HotMobile2007)*, Tucson, AZ, 2007.

[4] J. Oldmeadow, S. Ravinutala, and C. Leckie, "Adaptive clustering for network intrusion detection," in *Proceedings of the Third International Pacic-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2004)*, 2004.

[5] C. Loo, M. Ng, and M. Palaniswami, "Intrusion detection for routing attacks in sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2.4, pp. 313–332, 2006.

[6] A. Navia-Vazquez, D. Gutierrez-Gonzalez, E. Parrado-Hernandez, and J. Navarro-Abellan, "Distributed support vector machines," *IEEE Trans. on Neural Networks*, vol. 17-4, pp. 1091–1097, 2006.

[7] K. Flouri, B. Beferull-Lozano, and P. Tsakalides, "Training a svm-based classifier in distributed sensor networks," in *EUSIPCO*, 2006.

[8] S. M. Garrett, "How do we evaluate artificial immune systems?" *Evolutionary Computation*, vol. 13, no. 2, pp. 145–177, 2005.

[9] S. A. Hofmeyr and S. Forrest, "Architecture for an artificial immune system," *Evolutionary Computation*, vol. 8, no. 4, pp. 443–473, 2000.

[10] K. Leung and C. Leckie, "Unsupervised anomaly detection in network intrusion detection using clusters," in *The 28th Australasian Computer Science Conference*, 2005.

[11] T. Palpanas, D. P. V. Kalogeraki, and D. Gunopulos, "Distributed deviation detection in sensor networks," in *SIGMOD2003*, 2003.

[12] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Online outlier detection in sensor data using nonparametric models," *VLDB2006*, pp. 187–198, 2006.

[13] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection," in *Proceedings of the Third SIAM International Conference on Data Mining SDM03*, 2003.

[14] A. Lazarevic, A. Banerjee, V. Chandola, V. Kumar, and J. Srivastava, "Data mining for anomaly detection," in *Tutorial at the European Conference on Principles and Practice of Knowledge Discovery in Databases(PKDD08)*, 2008.

[15] B. Scholkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[16] D. M. J. Tax and R. P. W. Duin, "Support vector data description," *Journal of Machine Learning*, vol. 54-1, pp. 45–66, 2004.

[17] M. Davy, F. Desobry, A. Gretton, and C. Doncarli, "An online support vector machine for abnormal events detection," *Signal Processing*, vol. 86, pp. 2009–2025, 2006.

[18] S. Rajasegarar, C. Leckie, M. Palaniswami, and J. C. Bezdek, "Quarter sphere based distributed anomaly detection in wireless sensor networks," in *IEEE International Conference on Communications*, 2007.

[19] D. J. Hill, B. Minsker, and E. Amir, "Real-time bayesian anomaly detection for environmental sensor data," in *32nd Congress of the International Association of Hydraulic Engineering and Research (IAHR 2007)*, 2007.

[20] R. Khanna and H. Liu, "Control theoretic approach to intrusion detection using a distributed hidden markov model," *IEEE Wireless Communications*, vol. 15-4, pp. 24–33, 2008.

[21] P. Papadimitriou, A. Dasdan, and H. Garcia-Molina, "Web graph similarity for anomaly detection," Tech. Rep., 2008.

[22] C. C. Overall, J. L. Solka, J. W. Weller, and C. E. Priebe, "Using scan statistics for anomaly detection in genetic regulatory networks," in *Quantitative Methods in Defense and National Security 2007*, 2007.

[23] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality reduction for fast similarity search in large time series databases," *Knowledge and Information Systems*, vol. 3, no. 3, pp. 263–286, 2001.

[24] Z. Zhang and H. Shen, "Application of online-training svms for real-time intrusion detection with different considerations," *Computer Communications*, vol. 28, pp. 1428–1442, 2005.

[25] P. Laskov, C. Gehl, S. Krüger, and K. Müller, "Incremental support vector learning: Analysis, implementation and applications," *The Journal of Machine Learning Research*, vol. 7, pp. 1909–1936, 2006.

[26] Z. Ji and D. Dasgupta, "Real-valued negative selection algorithm with variable-sized detectors," in *Genetic and Evolutionary Computation Conference (GECCO)*, vol. 3102. Springer, 2004, pp. 287–298.

[27] S. G. Cheetancheri, J. M. Agosta, D. H. Dash, K. N. Levitt, J. Rowe, and E. M. Schooler, "A distributed host-based worm detection system," in *Proceedings of the 2006 SIGCOMM workshop on Large-scale attack defense*. New York, NY, USA: ACM, 2006.

[28] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.