Military Cyber Affairs
The Journal of the Military Cyber Professionals Association
ISSN: 2378-0789

Volume 2 | Issue 1 Article 6

2016

Evaluating Machine Learning Classifiers for Defensive Cyber Operations

Michael D. Rich
Air Force Institute of Technology, michael.rich@afit.edu

Robert F. Mills
Air Force Institute of Technology, robert.mills@afit.edu

Thomas E. Dube
Air Force Institute of Technology, thomas.dube.us@ieee.org

Steven K. Rogers
Air Force Research Laboratory, steven.rogers@us.af.mil

Evaluating Machine Learning Classifiers for Defensive Cyberspace Operations

MICHAEL D. RICH, Air Force Institute of Technology
ROBERT F. MILLS, Air Force Institute of Technology
THOMAS E. DUBE, Air Force Institute of Technology
STEVEN K. ROGERS, Air Force Research Laboratory

Today's defensive cyber sensors are dominated by signature-based analytical methods that require continuous maintenance and lack the ability to detect unknown threats. Anomaly detection offers the ability to detect unknown threats, but despite over 15 years of active research, the operationalization of anomaly detection and machine learning for Defensive Cyberspace Operations (DCO) is lagging. This article provides an introduction to machine learning concepts with a focus on the unique challenges to using machine learning for DCO. Traditional machine learning evaluation methods are challenged in favor of a value-focused evaluation method that incorporates evaluator-specific weights for classifier and sensitivity threshold selection specific to the values associated with cyber defense. A comprehensive unknown threat detection experiment is proposed to quantify a classifier's ability to detect previously unseen threats. The proposed experiments and evaluation methods are applied to a Department of Defense (DoD) Cyber Defense Exercise (CDX) dataset to validate the methodology.

1. Introduction

In the age of big data, analytics, and cloud computing, the value of algorithms is being proven in domains such as medicine, finance, and scientific research. This algorithmic value transfers to Defensive Cyberspace Operations (DCO). The current state of DCO relies heavily on signature-based systems to detect threats against networks and computer systems. Signature-based systems perform well identifying *known threats* and achieve low false positive rates when finely-tuned signatures are used. However, these systems are incapable of detecting *novel attacks* for which no signatures exist.

Anomaly detection is a commonly recommended solution to detect novel attacks. Anomaly detectors are trained with *normal* data, then alert on patterns that do not fit the normal model. This assumes it can differentiate between *unknown threats* and *normal*. A limitation with this approach is that anomaly detectors target outliers of the *normal* model, but unknown threats that occur within the normal model are overlooked. Similarly, benign activity may fall outside the normal model. The result is false negatives and false positives, respectively. Even when the anomaly detector can calculate a confidence measure, it will always make mistakes: both false negatives and false positives.

Despite over 15 years of active research, the operationalization of anomaly detection continues to lag in a market dominated by signature-based systems, regardless of seemingly high levels of performance exhibited in research. Machine learning is a commonly used tool for intrusion and anomaly detection research. While machine learning has been very successful in some domains, there are unique challenges applying these techniques to DCO that do not exist in other domains. Contributing challenges have been identified as (i) high cost of errors (time to investigate false alarms and cost of missed attacks); (ii) lack of labeled training data (ground truth); (iii) semantic gap between classification output and operational interpretation; (iv) variability in input data; and (v) fundamental difficulties conducting sound evaluations[1]. The

[1] Challenges are presented by developers of Bro, a leading open-source anomaly detection and IDS, in Sommer, Robin and Paxson, Vern. "Outside the closed world: On using machine learning for network intrusion detection." *Security and Privacy (SP), 2010 IEEE Symposium on.* IEEE, 2010. 305-316.

challenges of this domain require innovative machine learning solutions, as typical methods used in other domains are not going to work.

This article focuses on the challenge of evaluating machine learning models used in DCO applications. This article will provide an introduction to machine learning concepts, discuss the specific challenges to using machine learning for intrusion detection, and recommend a Value-Focused Thinking (VFT) decision-making method to evaluate machine learning classifiers. Empirical evidence is provided to validate the proposed value-focused evaluation method.

2. Background

In this section, an introduction to machine learning and classifier evaluation methods is provided. A background of machine learning in the intrusion detection domain is also provided. Finally, the VFT decision-making approach is introduced.

2.1. Introduction to Machine Learning

Machine learning is a multidisciplinary field that uses knowledge from the fields of artificial intelligence, statistics, computational complexity theory, control theory, information theory, philosophy, psychology, neurobiology, and others[2]. Machine learning focuses on learning algorithms that build models from data that can be used to make decisions or predictions. Machine learning can be considered a data-driven method of knowledge discovery, where knowledge is contained within the model built by the algorithm.

Classification is one use of machine learning in which algorithms are trained to discriminate between classes. To do this, data is arranged into individual instances comprised of features. Feature selection (or feature engineering) is necessary to determine appropriate features that allow for discrimination between classes[3]. Machine learning algorithms build models during a training phase using training samples. Once the model is built, it is tested by extracting the same features from a different set of samples which were not used for training. Results from the empirical tests are reported as matrix of correct and incorrect classifications, known as a confusion matrix (example shown in Table 1). From the confusion matrix, each instance of the test data will be evaluated as a true positive (TP), true negative (TN), false positive (FP), or false negative (FN). Metrics such as accuracy $\left(\frac{TP+TN}{TP+FP+FN+TN}\right)$, recall $\left(\frac{TP}{TP+FN}\right)$, precision $\left(\frac{TP}{TP+FP}\right)$, and false positive rates $\left(\frac{FP}{FP+TN}\right)$, can be derived from the matrix.

Table 1 - Two-Class Confusion Matrix

		Classified as	
		Positive	**Negative**
Actual	**Positive**	TP	FN
	Negative	FP	TN

[2] Mitchell, T. M. *Machine Learning.* 1st ed. New York, NY: McGraw-Hill, Inc., 1997.

[3] Kanter, James M., and Kalyan Veeramachaneni. 2015. "Deep Feature Synthesis: Towards Automating Data Science Endeavors." *Data Science and Advanced Analytics (DSAA), 2015 IEEE International Conference on.* IEEE.

DOI: http://doi.org/10.5038/2378-0789.2.1.1005

Most classifiers also have sensitivity adjustment, which can be viewed as a sliding scale that can be used to fine tune the performance. Using a higher level of sensitivity will classify more instances as positive, at the expense of more false positives. Conversely, using a lower sensitivity will classify more instances as negative, at the expensive of more false negatives (misses). Receiver Operating Characteristic (ROC) curves are used to visualize the tradeoff between true and false positive rates, and thereby establish an "optimal" [4] sensitivity setting. Precision-Recall (PR) curves perform a similar function, showing the tradeoff between precision, the rate of correctly classified instances to false positives, and recall, the true positive rate.

Many machine learning algorithms exist, such as Artificial Neural Networks (ANNs), decision trees, Support Vector Machines (SVMs), and Bayesian networks[5]. While the inner workings of these algorithms differ, they can all be viewed as a black box with adjustable settings used to create the "optimal" classifier model; the model with the best value for the chosen evaluation metric. The process of selecting an optimal algorithm and settings is mostly an empirical process. Using the same dataset, experiments can be conducted to compare multiple algorithms using various settings for each, with the goal of declaring an algorithm and/or settings as the optimal configuration for the dataset, with respect to a specific metric (e.g., accuracy, lowest FP rate, highest TP rate).

Beyond optimizing classifier settings, feature selection is another method of improving classification results. Experimenting with various features for data being classified can offer great insight and improve classification results. This typically requires a domain "expert" who can identify an ideal set of features to discriminate between classes.

The two main types of learning algorithms are supervised and unsupervised. Supervised learning algorithms are trained with data instances comprised of features that are labeled. For example, training samples could be labeled as malicious or normal. The learning algorithm performs statistical analysis of the features of each training instance to build a model that will discriminate between malicious and normal. Unsupervised learning algorithms are trained with data instances comprised of features that are not labeled. With no label provided for each instance, the unsupervised classifier only determines similarity between instances. Anomaly detectors primarily use unsupervised algorithms to create clusters of normal, labeling outliers as abnormal.

Research methods in the field of machine learning are much more involved than discussed here and require advanced statistical knowledge. This section is primarily meant to prepare the reader for the following discussion on using machine learning for DCO.

2.2. Machine Learning in Intrusion Detection

Machine learning has made great strides in domains such as speech recognition, image classification, and object detection[6]. Ground truth data, to train algorithms, exists in these domains to produce good models. Many researchers attempt to use similar techniques for cyber security, using data from audit logs, network traffic captures, system log files, and malware

[4] The term "optimal" is used with the awareness of Wolpert's "no free lunch" theorem, which there is no overall "optimal" and the term should be expressed towards a specific problem and not generalized over all problems, presented in Wolpert, David H. and Macready, William G. "No free lunch theorems for optimization." *Evolutionary Computation, IEEE Transactions on* (IEEE) 1, no. 1 (1997): 67-82. There is no intent of refuting this theorem.

[5] A technical review of common machine learning tools and techniques is well-presented in Witten, I. H., Frank, E., and Mark, A. 2011. Data Mining: Practical machine learning tools and techniques. 3rd ed. San Francisco: Morgan Kaufmann.

[6] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep learning." Nature 521 (7553): 436-444.

samples. Published research results often show relatively high classification results (greater than 95% accuracy in some cases), which leads us to question why anomaly detection systems are not widely deployed in operational environments.

Machine learning only works well when the operating conditions are completely captured by the training data. In the intrusion detection domain, the complete operating conditions cannot be captured, which is preventing the operationalization of machine learning for DCO. This requires innovation in how we choose to apply machine learning to this problem as the typical machine learning experiment and evaluation methods are not going to work. The problem should be reframed to account for the values associated with what we are trying to detect and/or prevent. With signature-based detectors doing a satisfactory job detecting known threats, the true need for machine learning is detecting unknown threats.

2.3. Detecting the Unknown Threat

The signature-based methodology is, and will always be, incapable of detecting novel attacks for which a signature does not yet exist. Signature-based systems should not be dismissed as they perform well for detecting known threats, but they should be augmented with machine learning algorithms to enable detection of unknown threats[7]. The value of using the two systems together can maximize classification of known and unknown threats. To truly understand the problem, we need to discuss the differences between known and unknown threats from the views of a human expert and a learning algorithm.

Much research has been conducted to operationalize anomaly detection[8], the notion of creating a model of normal then classifying outliers as anomalous. The terms "anomaly detection" and "machine learning" are commonly intermixed, stemming from the common use of clustering, an unsupervised machine learning approach, to the anomaly detection problem[9]. In fact, these terms should not be considered the same because many machine learning methods train using examples of both positive and negative classes, malicious and normal for example.

Sommer and Paxson[10] relate the outlier detection method of anomaly detection to the "closed world" assumption[11], where outliers are assumed to belong to the negative class. The authors continue to explain that domains where machine learning is successful rely on true classification problems, using samples of positive and negative classes to train a learner. Finally, the authors suggest that machine learning would be better suited for finding *variations of known attacks* rather than discovering *unknown attacks*.

Symons and Beaver[9] argue the semantics of *variations of known attacks* versus *unknown attacks* or *previously unseen attacks* presented by Sommer and Paxson, taking the position that normal traffic can be completely different from anything previously seen and that previously unseen attacks may not appear anomalous in the original feature space, but may have

[7] Dua, Sumeet, and Xian Du. 2011. Data mining and machine learning in cybersecurity. CRC press.
[8] Bhuyan, Monowar H., Dhruba K. Bhattacharyya, and Jugal K. Kalita. 2014. "Network anomaly detection: methods, systems and tools." Communications Surveys & Tutorials, IEEE (IEEE) 16 (1): 303-336.
[9] Symons, Christopher T. and Beaver, Justin M. "Nonparametric semi-supervised learning for network intrusion detection: combining performance improvements with realistic in-situ training." *Proceedings of the 5th ACM workshop on Security and artificial intelligence.* ACM, 2012. 49-58.
[10] Sommer, Robin and Paxson, Vern. "Outside the closed world: On using machine learning for network intrusion detection." *Security and Privacy (SP), 2010 IEEE Symposium on.* IEEE, 2010. 305-316.
[11] Witten, I. H., Frank, E., and Mark, A. *Data Mining: Practical machine learning tools and techniques.* 3rd ed. San Francisco: Morgan Kaufmann, 2011.

DOI: http://doi.org/10.5038/2378-0789.2.1.1005

distinguishing features that are more similar to known attacks than normal traffic. Their argument concludes by suggesting the problem is finding the correct view, and the appropriate expert-derived feature set, through which these distinctions can be made.

We agree with the issues associated with the "closed world" assumption, while also adopting the assumptions of Symons and Beaver regarding *unknown attacks*. We recommend using supervised learning algorithms which are trained with labeled input data from both the positive and negative classes. The algorithms develop a model to make classifications using input features. The nonlinear combination of input features used to form the model may or may not resemble that of the original feature representation. The similarity between known threats, which the system has seen, versus unknown threats, which the system has not seen, is not a simple variation of the known threat in the original feature representation observed by a human expert, but a similarity in the internal feature representation of the classifier model. This similarity in feature space may or may not represent a variation of a known attack or a novel attack in terms of the original feature space which a human expert is familiar. While the classifier's representation may or may not align easily to a human's representation, confidence in the model can be gained through testing and validation.

Classifiers will never be perfect; therefore, the risk and costs associated with false positives and false negatives must be assessed. This risk assessment is domain specific. The cost of Amazon recommending a product that a consumer does not want to purchase differs from the cost of a cyber operator researching false alerts. On the other hand, false negatives can be equally or more expensive. The cost of Amazon not recommending a product to a consumer may be a missed sales opportunity. A false negative in intrusion detection can result in a large data breach of personal information.

Decisions made by machine learning algorithms should correlate to values specific to the organizational mission and assets being protected. Value-Focused Thinking (VFT) is a decision-making approach that can be leveraged to evaluate machine learning models, incorporating explicit values into the selection and optimization of algorithms we use for cyberspace operations.

2.4. Value-Focused Thinking

VFT is a decision-making approach from the operations research domain where values are weighted in a manner that is relevant to the decision situations of an individual or organization[12]. The basis is that decisions are made by considering values rather than simply comparing alternatives.

Values are defined by the decision-maker (evaluator) and captured in a hierarchical structure of tiers where lower branches represent sub-values of the parent values. The tiers are also referred to as objectives, functions, tasks, and subtasks. All values are explicitly weighted by the evaluator, with the constraint that weights in the branches of each tier must sum to one. A simple mathematical function incorporates the provided weights and metrics for each value to compute a score which is used to evaluate candidate decisions. The "best" decision is the one with the highest score.

The VFT decision-making model is best explained with an example. A notional value hierarchy for an automobile purchase decision is provided in Figure 1. In this hierarchy, the tier

[12] Keeney, Ralph L. "Value-focused thinking: Identifying decision opportunities and creating alternatives." *European Journal of operational research* (Elsevier) 92, no. 3 (1996): 537-549.

1 values are safety, performance, and cost. The evaluator must provide weights for each of the tier 1 values and the weights must sum to one. A notional weighting scheme is shown where the tier 1 weights are .45 for safety, .30 for performance, and .25 for cost, summing to 1.0. The tier 2 values are the sub-values of the tier 1 values. Under the safety value, the sub-values are crash ratings and safety systems. These sub-values must be measurable since they are the child nodes in the hierarchy. A scaling system can be used for measurements that are not quantitative in nature, such as a crash rating scale from 1 to 5. The weights for each sub-value branch must also sum to one, as shown with weights of .50 for crash ratings and .50 for safety systems.

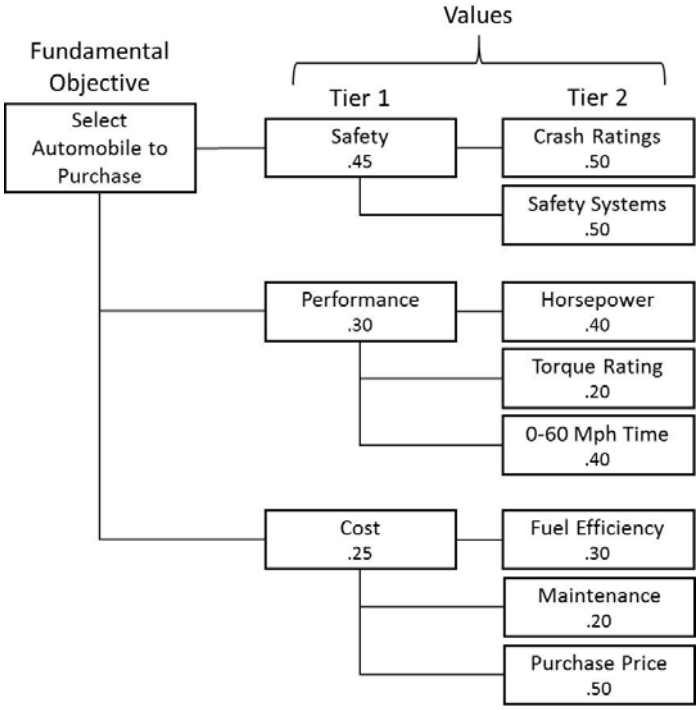


Figure 1 - Value Hierarchy Example

Using weighting schemes for values and related sub-values with a range between 0 and 1 produces an overall VFT score between 0 and 1. With the auto purchasing example, each measureable sub-value would be multiplied by the tiered weighting scheme for each vehicle subject to the decision. Each vehicle would receive an overall score between 0 and 1, the highest overall score being the "best" purchase decision based on the value hierarchy and weighting scheme used for the calculation. While this is a trivial example, VFT can be applied to more complex decisions while maintaining an easily understood model for the average user or manager to apply explicit weights.

3. Value-Focused Evaluation Methodology

Traditional classifier evaluation methods involve comparisons between metrics from the confusion matrix and performance throughout a range of thresholds using ROC or PR curve analysis. In this section, we propose using the VFT decision-making model to evaluate machine learning algorithms rather than simply comparing alternatives of traditional metrics.

Selecting a machine learning classifier and sensitivity threshold is a decision-making process in which cost, benefit, and risk must be considered. By applying the value-focused evaluation method, we can use evaluator-specified weighed values to compare classifier performance. To use the value-focused evaluation method for a DCO application, we must first build the hierarchy of values. At a high level, our values are to **detect known threats** to confirm or deny alerts from a signature-based detector, **detect unknown threats** to identify threats that the signature-based detector cannot detect, and maintain a suitable level of **precision** to ensure our cyber operators are not overwhelmed by false positives.

The proposed value-focused evaluation method is motivated by a scenario-based approach to mitigating insider threats[13] which considers benefits and costs associated with implementing security controls to detect insider threat activities. The scenario-based approach considered only binary security controls that were either on or off. This approach was extended to consider attack classifications as scenarios and the adjustable prediction threshold ability available in probabilistic machine learning classifiers. Figure 2 illustrates how the figures of merit (FOM) are calculated, where FOM_{XYZ} considers detection capabilities for each classifier by attack classification and threshold, and FOM_{XZ} considers the overall classifier performance across all attack classifications by threshold. This model easily extends to other applications with adjustable sensitivity thresholds.

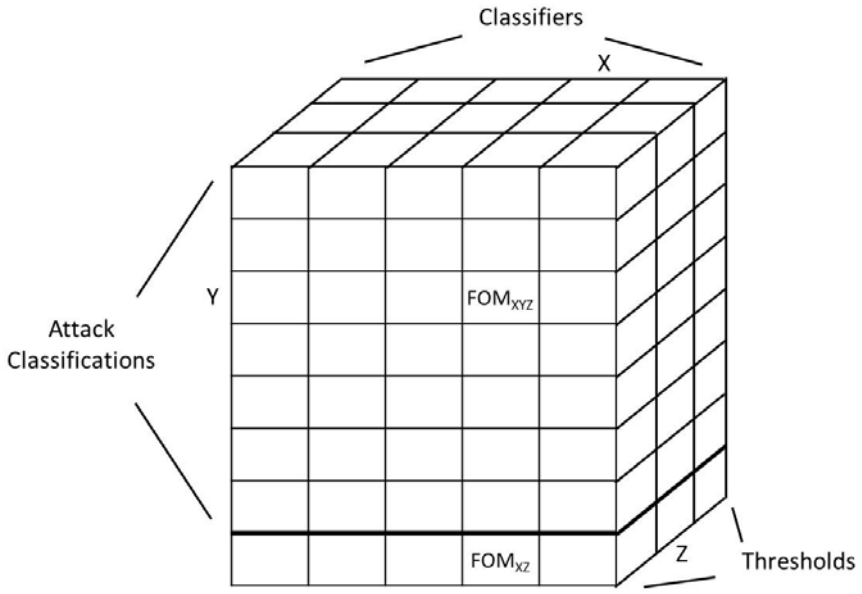


Figure 2 - Calculating Figures of Merit (adapted from (Mills 2011))

3.1. Value Hierarchy for Cyber Defense

The value hierarchy presented in Figure 3 represents the values relevant to using a machine learning classifier in a hybrid IDS configuration with a signature-based system. The fundamental objective is to select the *optimal*[14] classifier threshold. To calculate scores for each

[13] Mills, R. F., Grimaila, M. R., Peterson, G. L., and Butts, J. W. "A scenario-based approach to mitigating the insider threat." *Information Systems Security Association* 9, no. 4 (2011): 12-19.

[14] The term o*ptimal* is used again with awareness of Wolpert's "no free lunch" theorem with no intent of refuting the theorem.

threshold, we consider the tier 1 values as the known threat detection rates, precision, and unknown threat detection rates. Furthermore, we consider the detection rate for each attack class in tier 2, allowing an evaluator to weight the value of detecting specific attack classes. The term "attack class" is intentionally vague in this model to allow for interpretation to a specific problem. For example, attack classes could be the signature category for IDS alerts, network protocols for network-based IDS alerts, or the malware type for anti-virus systems. Any categorical metadata available for the data could be used to define the sub-tiers.

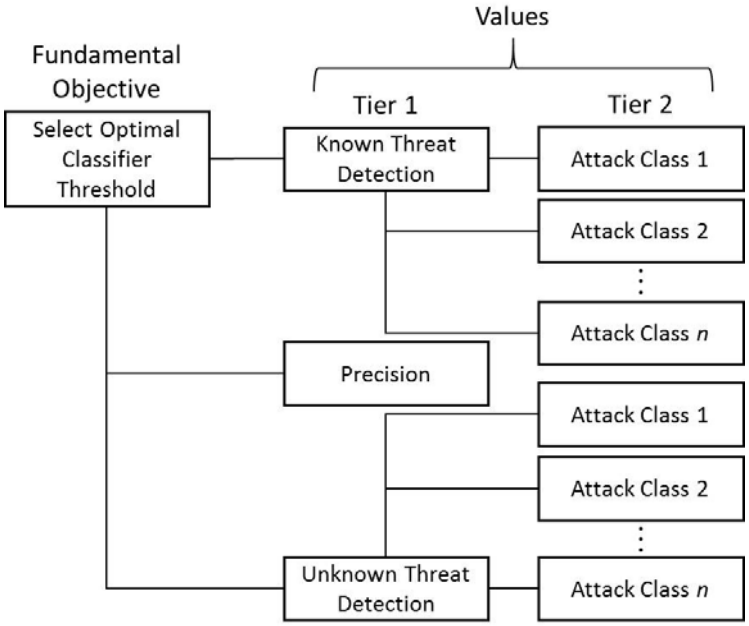


Figure 3 - Cyber Defense VFT Hierarchy

The value hierarchy in Figure 3 leads to the calculation of each FOM_{XZ} from Figure 2, then the optimal threshold setting is found by calculating the $max(FOM_{XZ_1}, FOM_{XZ_2}, \ldots, FOM_{XZ_n})$, where n is the number of thresholds for classifier X. The optimal thresholds can then be compared across multiple classifiers to select the optimal classifier and threshold pair: the$max(FOM_{X_1Z_1^*}, FOM_{X_2Z_2^*}, \ldots, FOM_{X_nZ_n^*})$, where n is the number of classifiers and Z_i^* represents the optimal threshold for classifier X_i.

3.2. Experiment Methodology

Two machine learning experiments are conducted to capture performance metrics required to compute FOM_{XYZ} and FOM_{XZ} for the value-focused evaluation. A typical classifier performance experiment is performed as well as an unknown threat detection experiment.

3.2.1. Classifier Performance Experiment

The classifier performance experiment is a typical stratified *k*-fold cross-validation experiment – a preferred method of testing performance of machine learning algorithms[15]. Using stratified

[15] Witten, I. H., Frank, E., and Mark, A. *Data Mining: Practical machine learning tools and techniques.* 3rd ed. San Francisco: Morgan Kaufmann, 2011.

cross-validation ensures the distribution of classes, threat and normal, is consistent across each fold.

The goal of the cross-validation experiment is to evaluate each classifier's performance of correctly classifying the data instances as threat or normal. There is an assumption made that each instance of the dataset used for training and testing is correctly labeled as a threat or normal, and if it is a threat, the attack classification is provided. This can be achieved by using a signature-based system for labeling of the dataset and using human expert verification of the labels. Essentially, this experiment measures each classifier's ability to match the output of the signature-based system. Performance metrics for known threat detection and precision are produced from this experiment.

3.2.2. Unknown Threat Detection Experiment

The unknown threat detection experiment is a unique attempt to quantify a classifier's ability to detect unknown threats. We are aware it is not possible to test for all unknown threats, as you cannot evaluate the unknown. We can, however, simulate unknown threats by withholding specific samples from each classifier's training process, then testing the classifier on samples which it has not been trained. Rather than hand-selecting unknown threats, leaving questions whether the unknown instances were similar to other training samples, we present a comprehensive approach similar to cross-validation, leaving less chance for misrepresentation of a classifier's ability to detect the unknown.

Training and testing datasets are developed similar to the cross-validation method, except folds are created for each attack classification rather than creating randomly stratified folds. All instances of a specific attack classification are withheld as the test set while the remaining instances are used as the training set. The test samples are unknown to the classifier as no instances containing the same attack class was used for training. This process is repeated for each attack classification, allowing each set of attack class instances the chance to be considered as unknown threats. Therefore, our definition of unknown threat is an instance that contains an attack classification in which the classifier has not been trained.

The unknown threat testing and training datasets are subjected to the same classifier configurations used in the classifier performance experiment to allow for comparison of performance measures from each experiment.

3.3. Performance Metrics

Performance metrics required for the value-focused evaluation are known threat detection rates by attack class, unknown threat detection rates by attack class, and precision. An explanation and formulas of each metric are provided in the following sections.

3.3.1. Known Threat Detection

Known threat detection, represented as DR_K, is the classifier's ability to detect known malicious connections of a specific attack classification. The metric used for DR_K is the recall rate from the classifier performance experiment,

$$DR_{K_{XYZ}} = Recall_{K_{XYZ}} = \frac{TP_{K_{XYZ}}}{TP_{K_{XYZ}} + FN_{K_{XYZ}}},$$

where X, Y, and Z are the classifier, attack classification, and threshold, respectively. A high DR_K will prove higher confidence when using alerts from the classifier to support true and false positives from the signature-based detector. A perfect DR_K would result in a classifier output that perfectly matches the output of the signature-based detector. While impressive, this model would not add much value over the signature-based detector alone without considering precision and the ability to detect unknown threats.

3.3.2. Unknown Threat Detection

Unknown threat detection, represented as DR_U, is the classifier's ability to detect unknown threats of a specific attack classification. The metric used for DR_U is the recall rate from the unknown threat detection experiment,

$$DR_{U_{XYZ}} = Recall_{U_{XYZ}} = \frac{TP_{U_{XYZ}}}{TP_{U_{XYZ}} + FN_{U_{XYZ}}},$$

where X, Y, and Z are the classifier, attack classification, and threshold, respectively. DR_U is arguably the most valuable metric used to evaluate a classifier for use in a hybrid configuration, alongside a signature-based system. The premise is if the signature-based system is well-suited to detect known threats, then the primary value of using a classifier is to detect unknown threats. A high DR_U will prove higher confidence that the system is capable of detecting threats undetected by the signature-based system.

3.3.3. Precision

Precision, represented as P, is the classifier's ability to detect malicious connections while not misclassifying normal connections as malicious. The metric used for P is the overall precision calculated from the classifier performance experiment,

$$P_{XZ} = Precision_{XZ} = \frac{TP_{XZ}}{TP_{XZ} + FP_{XZ}},$$

where X and Z are the classifier and threshold, respectively. P cannot be calculated per attack classification as it considers FP metrics that do not relate to an attack class. A higher P equates to less FPs per alert. The perfect rate of precision is 1, where every alert will be a TP.

3.4. Value-Focused Evaluation Calculations

The final calculations are used to compute the score for each classifier and threshold. The calculation used to compute each FOM_{XZ} from the value metrics discussed in the previous section is

$$FOM_{XZ} = W_{DR_K}\left(\sum_{i=1}^{n} W_i DR_{K_{XY_iZ}}\right) + W_{DR_U}\left(\sum_{i=1}^{n} W_i DR_{U_{XY_iZ}}\right) + W_P(P_{XZ}),$$

where W_{DR_K}, W_{DR_U}, and W_P are the evaluator-specified tier 1 weights for known threat detection, unknown threat detection, and precision, respectively. The evaluator-specified tier 2 weights for each attack classification are represented as $(W_1, W_2, \ldots, W_n)$, where n is the number of attack classes. Attack classes could be weighted differently for known threat detection and

DOI: http://doi.org/10.5038/2378-0789.2.1.1005

unknown threat detection. Weights must be selected so that the tier 1 weights sum to 1, $W_{DR_K} + W_{DR_U} + W_P = 1$, and the tier 2 weights sum to 1, $\sum(W_1, W_2, \ldots, W_n) = 1$.

The optimal classifier threshold, $FOM^*_{X_iZ}$, is then defined as threshold with the highest FOM_{XZ}

$$FOM^*_{X_iZ} = \max\left(FOM_{X_iZ_1}, FOM_{X_iZ_2}, \ldots, FOM_{X_iZ_n}\right),$$

where n is the number of thresholds. Each classifier will have an optimal threshold.

Finally, the optimal classifier with threshold, FOM^*_{XZ}, is defined as classifier with the highest $FOM^*_{X_iZ}$

$$FOM^*_{XZ} = max\left(FOM^*_{X_1Z}, FOM^*_{X_2Z}, \ldots, FOM^*_{X_nZ}\right),$$

where n is the number of classifiers.

4. Preliminary Results

In this section, preliminary empirical results are provided using network-based intrusion detection data. Traditional performance results are presented first using PR curve analysis, followed by the value-focused evaluation results. The value-focused evaluation method is validated using notional, but plausible, weighting schemes and comparing classifier selection results with the traditional evaluation method. Fallacies are identified in these comparisons which would lead to a less-than-optimal classifier selection for a specific value set if only the traditional evaluation methods were considered.

4.1. Dataset Preprocessing

The data used in this research is from the Cyber Defense Exercise (CDX), an annual cyber warfare exercise sponsored by the US National Security Agency (NSA). Network traffic was captured on the boundary router at the Air Force Institute of Technology (AFIT) for the duration of each exercise[16]. The CDX data is in raw packet capture format (libpcap) and does not have ground truth labeling of attacks. A total of 23,750,535 packets were available from the exercises for years 2003 through 2007, and 2009.

Security Onion, a network security Linux distribution, is used as a sensor to process the raw network data. Snort is used as the signature-based IDS, configured with the latest available signature sets. Bro is used to extract connection-level features. Tcpreplay is used to replay the network traffic over the sensor. Snort generated 732,709 packet-level alerts (3.085% of total packets), with 169 unique signatures triggering from 16 attack classes. Bro logged 3,841,291 connections.

The features logged by Bro include connection-level statistics such as duration, byte counts, connection end states, connection state history (TCP flags), and packet statistics. The connections are labeled threat or normal by correlating the packet-level alerts generated by Snort. If a packet is malicious, then the connection containing that packet is considered malicious. Without manually verifying each Snort alert, we assume there are no false positives or negatives in the set of alerts. While a large assumption, it is acceptable for this research as we are strictly

[16] Mullins, Barry E., Tim H. Lacey, Robert F. Mills, Joseph M. Trechter, and Samuel D. Bass. 2007. "How the cyber defense exercise shaped an information-assurance curriculum." Security & Privacy, IEEE 5 (5): 40-49.

demonstrating evaluation methods without the goal of developing production-ready models. The connection-alert correlation resulted in 146,531 connections containing one or more packet-level alerts, 3.81% of the total connections. Even with a dataset collected in an environment where attacks are imminent, the ratio of malicious to non-malicious connections is minuscule. It is expected that this difference is amplified in a typical operational network, resulting in an even larger imbalance between the positive and negative classes.

The data cleanup process involved removing timestamps, IPs, and port information so these features would not impact the classification decisions. String-based features were removed or converted to nominal features. No attempt to balance the positive and negative classes were made, imbalanced data was used to better represent the target environment where there is an expected imbalance between normal and malicious connections. Random sampling was used to reduce the size of dataset to reduce computation time. All threat class instances were retained and a 10% random sample of the normal class instances. A final dataset description is presented in Table 2.

Table 2 - Final Dataset Description

Connections	Normal	Threat	% Threat	# of Features
384,129	237,598	146,531	38.15	34

4.2. Machine Learning Environment

The Waikato Environment for Knowledge Analysis (WEKA) machine learning suite from the University of Waikato[17] is used for both experiments. Six supervised classification algorithms are used: BayesNet, Sequential Minimization Optimization (SMO), J48 decision trees, multilayer perceptron (MLP) neural network, and adaptive boosting (AdaBoost) applied to BayesNet and J48[18]. Parameter optimization is not considered in these experiments, therefore default settings are used for each classifier.

4.3. Traditional Performance Evaluation

As discussed in Section 2.1, there are many metrics that can be derived from the confusion matrix results of a machine learning experiment. For this article, PR curve analysis is used as the traditional performance evaluation method. Similar to ROC curves, PR curves consider metrics throughout all possible sensitivity thresholds. ROC curves depict a trade-off between TP rates and FP rates, while PR curves depict a trade-off between TP rates (recall) and precision. PR curves were preferred for this experiment as they are better suited to analyze imbalanced data, where the majority class is the negative class[19] .

The PR curves for each classifier presented in Figure 4 show the performance of each classifier across the range of thresholds. By visual inspection alone, curves closer to the upper right of the chart, where $Precision = 1$ and $Recall = 1$, are considered superior classifiers. The curves for AdaBoostJ48 and J48 dominate the PR space at all thresholds, followed by

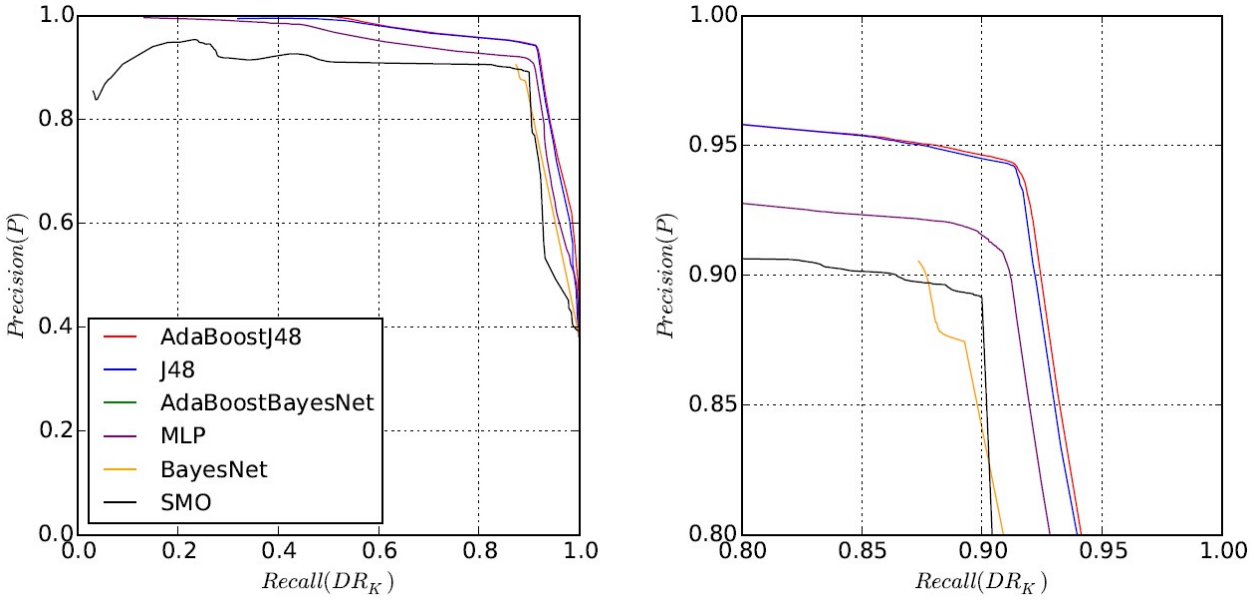
[17] Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. "The WEKA data mining software: an update." ACM SIGKDD explorations newsletter (ACM) 11 (1): 10-18.

[18] A detailed technical explanation of these algorithms along with configurable settings can be found in Witten, I. H., Frank, E., and Mark, A. 2011. Data Mining: Practical machine learning tools and techniques. 3rd ed. San Francisco: Morgan Kaufmann.

[19] Davis, Jesse and Goadrich, Mark. 2006. "The relationship between Precision-Recall and ROC curves." *Proceedings of the 23rd international conference on Machine learning.* ACM. 233-240.

DOI: http://doi.org/10.5038/2378-0789.2.1.1005

AdaBoostBayesNet. Ultimately, once a classifier is selected, one threshold point along the curve would be selected to deploy the model into a production environment. Confidence intervals of points along the curve would also be considered in a more detailed analysis. The complete rank ordering of classifier selection using a visual PR curve analysis would be AdaBoostJ48, J48, AdaBoostBayesNet, MLP, SMO, and BayesNet.



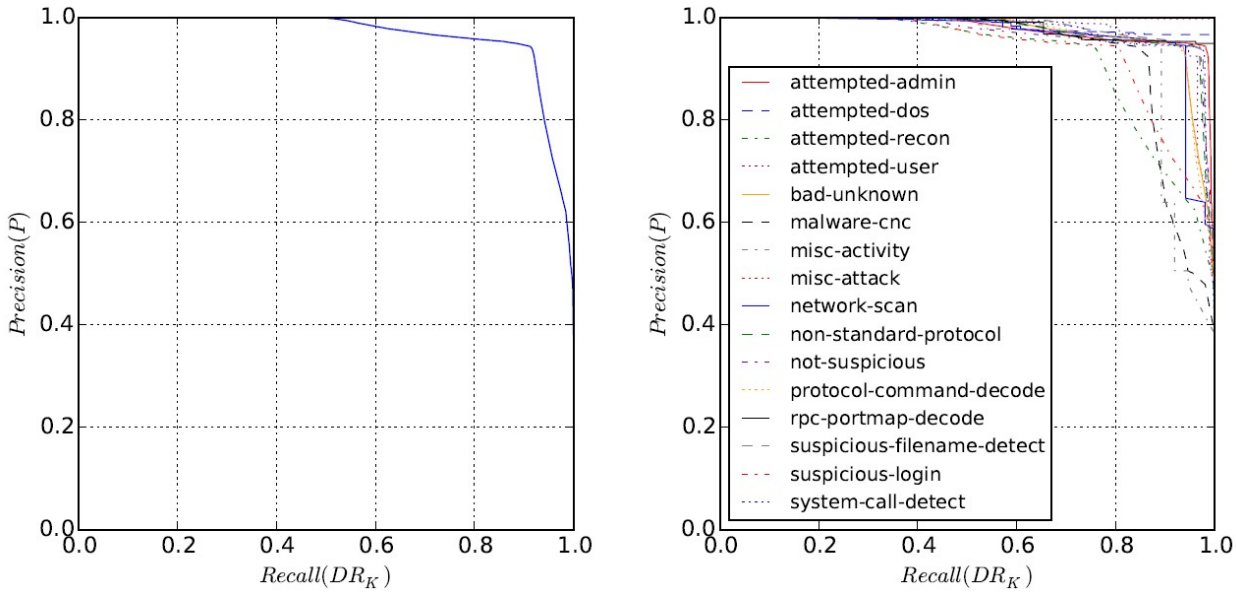
(a) Full PR Curves

(b) Zoomed View of PR Curves

Figure 4 - Traditional PR Curve Comparison

4.4. Value-Focused Evaluation

Unique to this research, as part of the value-focused evaluation, PR curve analysis is extended by extrapolating detection rates by alert classification across the range of prediction thresholds. This is accomplished by reversing the method used to correlate packet-level alerts to connections after the instance has been classified. A mapping of alert classes to packets and from packets to connections is maintained as part of the bookkeeping. This extrapolation produces the detection rates for each alert class at everything threshold, metrics required for the value-focused evaluation. A visual representation of a PR curve extrapolated by alert class for one classifier is shown in Figure 5. While not entirely intuitive, the value-focused evaluation method provides a means to evaluate each point on the PR curves by alert class for both the known threat detection rates and unknown threat detection rates. With 16 alert classes present, there are 32 curves being evaluated for each classifier and the evaluator is able to explicitly weight each curve. With this intuition, we start to see the benefit in extrapolating these data points and weighing them according to the value of detecting each type of threat. An organization will likely value detecting command and control malware on their network over a port scan.



(a) Single PR Curve

(b) PR Curves by Alert Class

Figure 5 - PR Curve Extrapolated by Alert Class

Notional weighting schemes were developed to demonstrate the value-focused evaluation methodology. The weighting schemes and classifier selection results are summarized in Table 3 and are explained in detail below. Default alert classifications and priorities from Snort, shown in Table 4, were used to generalize the tier 2 weighting into high, medium, and low priority alerts[20].

Table 3 - Value-Focused Notional Weighting Schemes and Classifier Selection

Weighting Schemes			
	Balanced	Alert Class Priority	Detect Unknown
Tier 1 Weights			
DR_K	0.25	0.25	0.00
P	0.50	0.50	0.50
DR_U	0.25	0.25	0.50
Tier 2 Weights			
High Pri	0.0625	0.14	0.14
Med Pri	0.0625	0.06	0.06
Low Pri	0.0625	0.01	0.01
Classifier Selection Results			
1	AdaBoostJ48	AdaBoostJ48	MLP
2	J48	J48	J48
3	AdaBoostBayesNet	MLP	AdaBoostJ48
4	MLP	AdaBoostBayesNet	BayesNet
5	BayesNet	BayesNet	AdaBoostBayesNet
6	SMO	SMO	SMO

[20] Explanations of Snort alert classes can be found in the Snort User's Manual, http://manual.snort.org.

Table 4 - Default Snort Alert Class Priorities

High Priority	Medium Priority	Low Priority
attempted-admin	attempted-dos	misc-activity
attempted-user	attempted-recon	network-scan
malware-cnc	bad-unknown	not-suspicious
	misc-attack	protocol-cmd-decode
	non-standard-protocol	
	rpc-portmap-decode	
	susp-filename-detect	
	suspicious-login	
	system-call-detect	

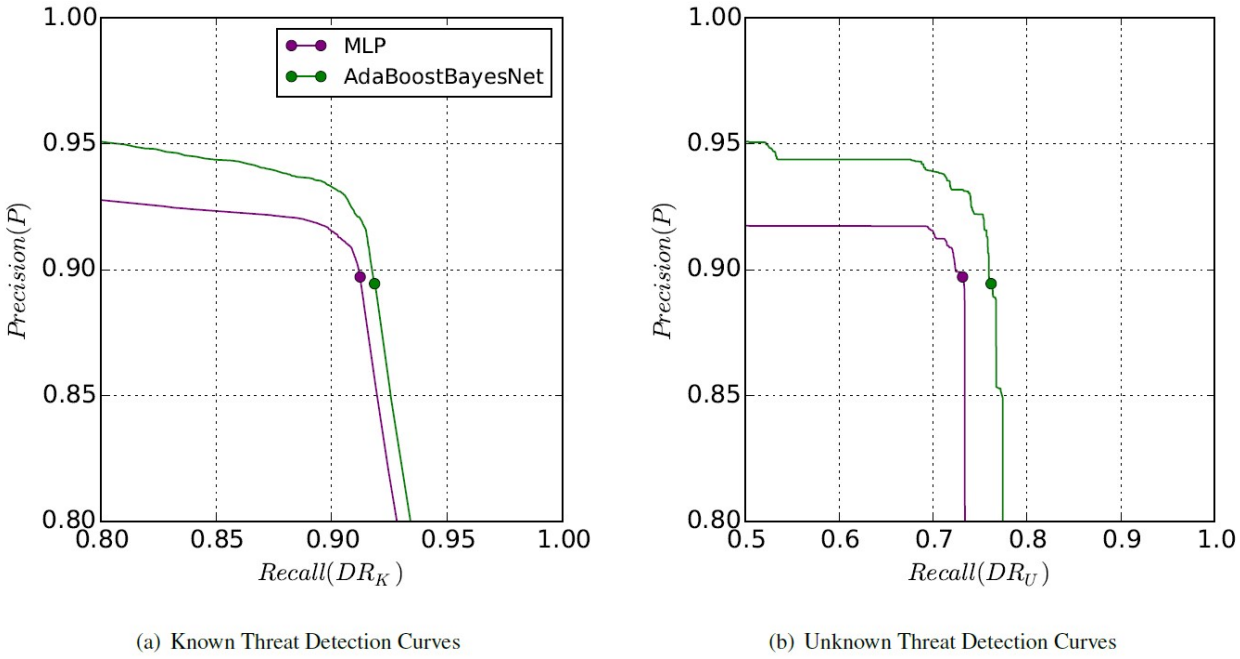
4.4.1. Balanced Weighting Scheme

A balanced weighting scheme is used as a baseline, with the expectation that the classifier selection ordering would mirror the traditional evaluation method. To achieve balanced weighting on detection and precision, DR_K and DR_U are weighted at 0.25 each, summing to 0.50, and P is weighted at 0.50. To balance the weight across the 16 alert classifications, 0.0625 (1/16) is used. The difference in classifier selection with this weighting scheme and the traditional evaluation method is BayesNet is ranked fifth and SMO is ranked sixth, which is attributed to the superior UTD rate from BayesNet. Otherwise, the rank ordering and overall classifier selection of AdaBoost J48 is consistent with the traditional evaluation method.

4.4.2. Alert Class Priority Weighting Scheme

The alert class priority weighting scheme places emphasis on the default alert class priorities used by Snort. High priority classes receive a weight of 0.14, medium priority classes receive a weight of 0.06, and low priority classes receive a weight of 0.01. The weighting of alert classes can be applied individually; the chosen weights are simply a notional application that is used for demonstration. This weighting scheme does not take full advantage of weighting known threat detection, unknown threat detection, and precision.

A significant finding in the classifier selection rank ordering from this weighting scheme is that MLP is ranked higher than AdaBoostBayesNet. When analyzing the single PR curves for these two classifiers, as shown in Figure 6, it is clear that AdaBoostBayesNet dominates the PR space. The points on each curve represent the optimal threshold selected by the value-focused evaluation. This demonstrates the fallacy that can occur when selecting classifiers without considering detection rates by alert classification. By examining the detection rates by alert class, the value-focused evaluation method determines that MLP is the superior classifier, whereas a traditional PR curve analysis of would select AdaBoostBayesNet.



(a) Known Threat Detection Curves

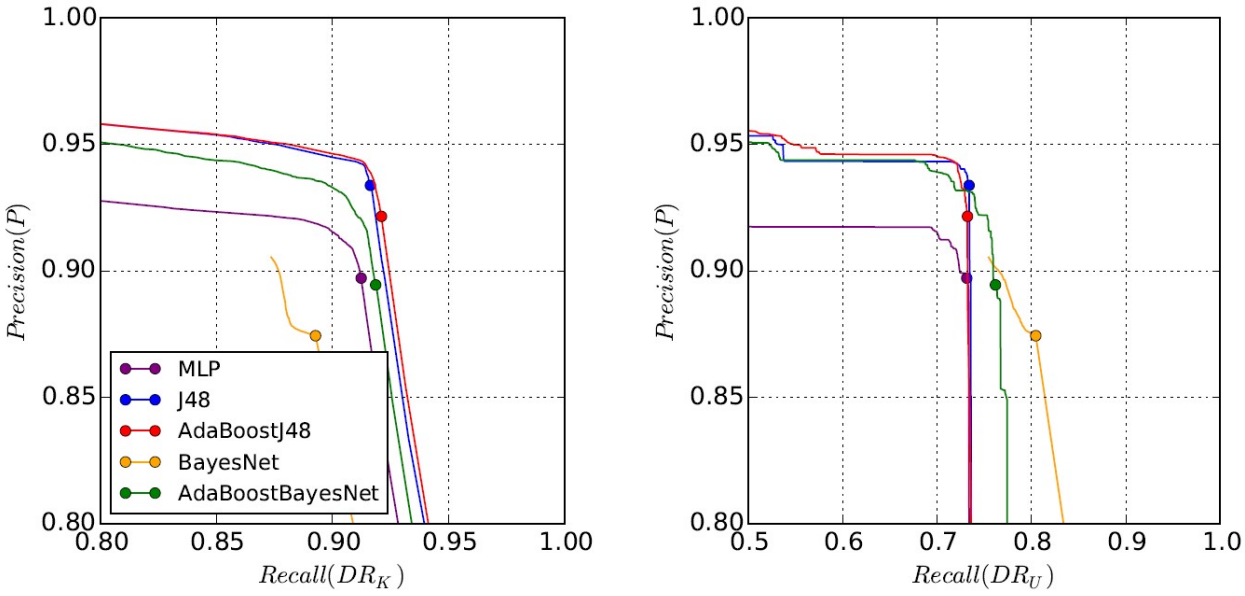
(b) Unknown Threat Detection Curves

Figure 6 - Value-Focused Threshold Selection for Alert Class Priority Weighting Scheme

4.4.3. Detect unknown weighting scheme

The detect unknown weighting scheme places emphasis on the detection of unknown threats by using a higher weight of 0.50 for DR_U and 0.00 for DR_K. The weight for P remains at 0.50 and the alert class weighting remains set to the alert class priority weighting. This weighting scheme would be plausible for a classifier intended solely for detecting unknown threats with a balance of precision. The classifier selection for this weighting scheme ranks MLP the highest, with J48 now outranking AdaBoostJ48, and BayesNet now outranking AdaBoostBayesNet.

This weighting scheme further demonstrates how the value-focused evaluation method can be used for classifier selection based on the intended use of the classifier, detecting unknown threats in this case. The single PR curves for the top 5 classifiers, with the value-focused threshold selections annotated, are shown in Figure 7. Similar to the findings with the alert class priority weighting scheme, we see that a seemingly inferior classifier using traditional PR curve analysis, MLP, is selected as the overall optimal classifier by the value-focused evaluation.



(a) Known Threat Detection Curves

(b) Unknown Threat Detection Curves

Figure 7 - Value-Focused Threshold Selection for Detect Unknown Weighting Scheme

Conclusion

In this article, an overview of machine learning and the challenges applying machine learning to DCO were discussed. While machine learning continues to be successful in some domains, we continue to struggle operationalizing machine learning for cyber defense. One of these challenges includes the evaluation of machine learning models. Cyberspace has proven to be different from other domains; therefore, we need innovative evaluation methods for machine learning models to be used in cyberspace operations. We provided a value-focused evaluation method suited for this domain and provided preliminary results validating the usefulness of the method.

Recommended future work is to apply the value-focused method to other cyber audit data (e.g., host-based IDS, malware detection), employ the value-focused score metric to tune classifier parameters to the evaluator-specified values, and to expand the default alert classifications to allow an evaluator to further understand the detection capabilities of the classifiers being evaluated. The dataset used in this experiment was very small compared to the realistic daily amount of network traffic experienced in an enterprise. Further experiments on larger datasets from an operational domain are needed for further validation of the value-focused methodology. The potential of this evaluation method is to provide insight of threats to operators in dynamic environments while allowing the operator to adjust weights of the model depending on current situations. Further experiments in an operational environment should also monitor the human-machine interactions to provide a better understanding of the capabilities and limitations of the algorithms and the human operator.

Disclaimer

The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense, or the United States Government

Bibliography

Bhuyan, Monowar H., Dhruba K. Bhattacharyya, and Jugal K. Kalita. 2014. "Network anomaly detection: methods, systems and tools." Communications Surveys & Tutorials, IEEE (IEEE) 16 (1): 303-336.

Davis, Jesse and Goadrich, Mark. 2006. "The relationship between Precision-Recall and ROC curves." Proceedings of the 23rd international conference on Machine learning. ACM. 233-240.

Dua, Sumeet, and Xian Du. 2011. Data mining and machine learning in cybersecurity. CRC press.

Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. "The WEKA data mining software: an update." ACM SIGKDD explorations newsletter (ACM) 11 (1): 10-18.

Kanter, James M., and Kalyan Veeramachaneni. 2015. "Deep Feature Synthesis: Towards Automating Data Science Endeavors." Data Science and Advanced Analytics (DSAA), 2015 IEEE International Conference on. IEEE.

Keeney, Ralph L. 1996. "Value-focused thinking: Identifying decision opportunities and creating alternatives." European Journal of operational research (Elsevier) 92 (3): 537-549.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep learning." Nature 521 (7553): 436-444.

Mills, R. F., Grimaila, M. R., Peterson, G. L., and Butts, J. W. 2011. "A scenario-based approach to mitigating the insider threat." Information Systems Security Association 9 (4): 12-19.

Mitchell, T. M. 1997. Machine Learning. 1st ed. New York, NY: McGraw-Hill, Inc.

Moore, Andrew and Zuev, Denis and Crogan, Michael. 2005. Discriminators for use in flow-based classification. Queen Mary and Westfield College, Department of Computer Science.

Mullins, Barry E., Tim H. Lacey, Robert F. Mills, Joseph M. Trechter, and Samuel D. Bass. 2007. "How the cyber defense exercise shaped an information-assurance curriculum." Security & Privacy, IEEE 5 (5): 40-49.

Sommer, Robin and Paxson, Vern. 2010. "Outside the closed world: On using machine learning for network intrusion detection." Security and Privacy (SP), 2010 IEEE Symposium on. IEEE. 305-316.

Symons, Christopher T. and Beaver, Justin M. 2012. "Nonparametric semi-supervised learning for network intrusion detection: combining performance improvements with realistic in-situ training." Proceedings of the 5th ACM workshop on Security and artificial intelligence. ACM. 49-58.

Witten, I. H., Frank, E., and Mark, A. 2011. Data Mining: Practical machine learning tools and techniques. 3rd ed. San Francisco: Morgan Kaufmann.

Wolpert, David H. and Macready, William G. 1997. "No free lunch theorems for optimization." Evolutionary Computation, IEEE Transactions on (IEEE) 1 (1): 67-82.

DOI: http://doi.org/10.5038/2378-0789.2.1.1005