# Reconstructing Classification to Enhance Machine-Learning Based Network Intrusion Detection by Embracing Ambiguity

Chungsik Song[‡*], Wenjun Fan[†*], Sang-Yoon Chang[†] and Younghee Park[‡]
[†]*Computer Science Department, College of Engineering and Applied Science*
*University of Colorado Colorado Springs,* Colorado Springs, United States, CO 80918
Email: {wfan, schang2}@uccs.edu
[‡]*Computer Engineering Department,*
*San Jose State University,* San Jose, United States, CA 95192
Email: {chungsik.song, younghee.park}@sjsu.edu

***Abstract*—Network intrusion detection systems (IDS) has efficiently identified the profiles of normal network activities, extracted intrusion patterns, and constructed generalized models to evaluate (un)known attacks using a wide range of machine learning approaches. In spite of the effectiveness of machine learning-based IDS, it has been still challenging to reduce high false alarms due to data misclassification. In this paper, by using multiple decision mechanisms, we propose a new classification method to identify misclassified data and then to classify them into three different classes, called a malicious, benign, and ambiguous dataset. In other words, the ambiguous dataset contains a majority of the misclassified dataset and is thus the most informative for improving the model and anomaly detection because of the lack of confidence for the data classification in the model. We evaluate our approach with the recent real-world network traffic data, Kyoto2006+ datasets, and show that the ambiguous dataset contains 77.2% of the previously misclassified data. Re-evaluating the ambiguous dataset effectively reduces the false prediction rate with minimal overhead and improves accuracy by 15%.**

***Index Terms*—Network Intrusion Detection, Machine Learning, , Ensemble Classifiers**

## I. Introduction

The interest to apply advanced machine learning techniques for the intrusion detection system has been steadily growing since the late 1990s [1]–[4]. Machine learning methods discover hidden, valid patterns and relationships among the attributes in a large dataset to find malicious actions through automatic model construction. These approaches allow for a more flexible and distributable way to identify network activity based on limited, incomplete, and nonlinear data sources by identifying adversarial activities [5]–[9]. Due to these promising capabilities, machine learning-based IDSs have been extensively studied by the research community for many years [10]–[16]. Network intrusion detection system (NIDS) endeavors to discover unauthorized access to network resources by analyzing the network traffic data and detect the signs of malicious activities which undermine the normal operation of a network [17], [18].

However, in real-world operational settings, only misuse detectors are predominantly used in the form of signature systems that scan the network traffic for characteristics byte sequences [19]. The imbalance between the extensive amount of research and the lack of operational deployments of such systems stems in large part from specifics of the problem domain. The high error rate is one of the primary reasons for the lack of success of machine learning-based intrusion detection systems in operational settings. Furthermore, to reduce misclassification errors and minimize false alerts is one of the major challenges in machine learning-based intrusion detection systems [20], [21].

To address the challenge of the misclassification errors in the machine learning-based intrusion detection system, this paper introduces a new category to embrace the ambiguity of the decision process and classify the dataset into three categories, benign (confident), malicious (confident), and ambiguous dataset. To categorize the confident and ambiguous datasets, we develop a confidence score system and utilize it for the decision of the classification. The ambiguous dataset is a collection of data that receive mixed decision predictions or unanimously indecisive predictions in the classification process of the model. The proposed system uses an ensemble model for classification and we evaluate our approach with recent real network traffic data. We extract the ambiguous dataset and show that it contains most of the misclassified data in the datasets. In other words, most of the misclassified results in the model, either false alerts or missed attacks, are included in the ambiguous dataset. We re-evaluate the ambiguous dataset to reduce the uncertainty in the decision process. Re-evaluating the ambiguous dataset effectively reduces false predictions with minimal overhead and improves the system performance for intrusion detection. Also, our scheme makes correction decisions on the dataset labeled as "Unknown" attacks which

This work is supported by NSF Award #1723663.

resulted in poor performances previously, and classify them as malicious with high confidence. Our contributions are summarized as follows.

1) The concept of an ambiguous dataset is first introduced to identify a subset of test data that contains the most uncertainty in the decision process of the model for the intrusion detection system.
2) We evaluate our approach with the recent real network traffic data and show that the ambiguous dataset contains 77.2% of misclassified data in our experiment.
3) We show that re-evaluating the ambiguous dataset effectively reduces false predictions and improves the accuracy of the anomaly detection by contributing to classify the full dataset completely into a specific class with almost no ambiguous data.

The rest of the paper is organized as follows. Section 2 describes the motivation of reconstructing classification by embracing the ambiguity in the intrusion detection system. Section 3 presents the structure of the classification and decision model used for intrusion detection. In section 4 we discuss the public datasets available for the evaluation of intrusion detection systems and our model. Section 5 and section 6 include the details of our evaluations and results, respectively. The details of the ambiguous dataset and its impact on our detection model are discussed. Section 7 summarizes related works. We conclude our work with discussion and future works in Section 8.

## II. Reconstructing Classification by Embracing Ambiguity

While the classification of variable sources and transient events can be obtained using real-time and archival data, the classification will be ambiguous in many cases. Additional data and analysis with other datasets would be needed for the confidence of classification results in the case of interesting events. This poses a challenge of the automated decision-making process for the optimal use of the available, finite computing resources, and limited time constraints. Most machine learning algorithms make the assumption that training data is a random sample drawn from a stationary distribution. A fixed set of features can represent the data when the underlying structure of the data is stationary. When we have more data and especially use optimized feature sets that differentiate malicious activities from benign ones, we can reduce the ambiguity in classification. However, continuous and rapid changes in network usage patterns, as well as the attack patterns, have added another level of challenges [20], [21].

Alternatively, instead of maximizing classification accuracy, we consider a scenario where the algorithm chooses a set of data that are most uncertain, determines the follow-up analyses, and improves the accuracy of classification in the subsequent analysis. The goal of the algorithm is to select a set of data that is the most interesting. The subsequent analysis then provides information on how interesting the dataset is. We follow the second approach and select a set of data that are most uncertain, called an ambiguous dataset. The ambiguous dataset is a dataset that is expected to be most informative and can most effectively improve the model in subsequent analysis. The source of the ambiguous dataset is not the intrinsic characteristics of the data but the deficiency of the model used. The selection of the ambiguous dataset thus depends on the balance between efficiency and accuracy of the detection model. There are two parts to this challenge. First, the selected dataset should be the most uncertain sample that is expected to be the most informative and is capable of most improving the model in subsequent analysis. And second, what type of follow-up measurement and/or analysis, given the available set of resources, would yield the maximum information gain in a situation? The subsequent section describes the way to classify the dataset into three classes, malicious, benign, and ambiguous. The definition of the ambiguous dataset and its relation to the misclassified dataset are followed.

### A. Classify into Three Classes

Algorithm 1 shows the pseudo-code of three-class classification. Firstly, predicted probability is evaluated for each base classifier used in the ensemble model (see lines 1-2). These values are used to evaluate the predicted probability of the ensemble classifier (see lines 3-4). Based on the evaluated predicted probability system classifies data into three classes, malicious (lines 5-6), benign (lines 7-8), and ambiguous (lines 9-10). Base classifiers and ambiguous threshold $\theta_A$ are provided as inputs. For the confident (either benign and malicious) dataset, we take actions based on the type of a decision, benign and malicious, as other traditional intrusion detection systems do. But we take further analysis and evaluation of the ambiguous dataset since most of the false alerts or missed attacks belong to this category and even novel (unknown) attacks can be included in this dataset. Since the ambiguous dataset is a small portion of the full dataset ($\sim$ 10% of the full dataset), the re-evaluation process is light in resource usage and fast in the computing process.

**Algorithm 1: Three-Class Classification**

**Inputs:**
classifiers: $\{C_i\}$
ambiguous threshold $\theta_A$
**Output :**
prediction result
1: Calculate the predicted probability from base classifier $C_i$:
2: $P_i$ = Predicted Probability ($C_i$)
3: Take the average of individual predicted probabilities to get the ensemble predicted probability:
4: $P_{ens} = Avg(P_i)$
5: **if** $P_{ens} > 1 - \theta_A$
6: return malicious
7: **else if** $P_{ens} < \theta_A$
8: return benign
9: **else** $(\theta_A < P_{ens} < 1 - \theta_A)$
10: return ambiguous

### B. Ambiguous Dataset

We define the ambiguous dataset as those received mixed decision predictions or unanimously indecisive predictions in

the multiple classifier systems. For a system of binary classes, either 1 (malicious) or 0 (benign), each classifier estimates the prediction probability for a specific category (benign or malicious) indicating its decision about the class of the object. The system chooses the class with the highest values of the prediction probability for the object. If most of the classifiers vote for malicious, the prediction probability for the class of the object will be close to 1. Otherwise, the prediction probability will be close to 0 when most of them vote for benign. The ambiguous dataset consists of the data that receive the value near 0.5. When using a probabilistic model for the binary classification, the ambiguous dataset simply queries the instance whose posterior probability of being malicious is near 0.5.

We introduce an ambiguous threshold value, $\theta_A$ where $0 < \theta_A < 0.5$, to determine the range of *prediction probability* for an ambiguous dataset. The threshold is the value our model uses for a classification decision to be confident. For binary classification (e.g., malicious and benign), our model classifies to confident "malicious" when *prediction probability* $> 1-\theta_A$ and confident "benign" when *prediction probability* $< \theta_A$. The ambiguous dataset is defined as those with $\theta_A \leq$ *prediction probability* $\leq 1 - \theta_A$.

*a) Misclassified Dataset:* One of the challenges in applying machine learning techniques to intrusion detection systems is the high rate of false alerts. Even though the error rate is very low ($1 \sim 2\%$), the number of misclassified data is not negligible with the growing number of data that need to be processed and analyzed. Our analysis in the subsequence section shows that most of the misclassified data – either false alerts or missed attacks – are included in the ambiguous dataset. Figure 5 shows the distribution of the predicted probability in the misclassified dataset. More than 77% of the misclassified dataset are distributed within the window for the ambiguous dataset, $\theta_A \leq$ *prediction probability* $\leq 1 - \theta_A$. This result indicates that most of the misclassified samples distribute near the boundary of two target classes and define the ambiguous dataset.

*b) Evaluation of Ambiguous Dataset:* The most uncertain samples are expected to be the most informative and are capable of most improving the model. As we show, the ambiguous dataset contains the majority of the misclassified dataset and is the most uncertain subset in our decision model. The measured performance metrics for the ambiguous dataset show low values compared to those of the confident dataset which excludes the ambiguous dataset from the full test dataset. We re-evaluate the ambiguous dataset to reduce the uncertainty in the decision process. This process concerns not only the uncertainty but also the outcomes and risks of this uncertainty. We build a model for the ambiguous dataset and re-evaluate the performance. The accuracy of detecting malicious activities increases to 90%. The importance of features is re-evaluated for the ambiguous dataset and compared to that of the full dataset.

## III. Model Build

In this section, we describe the classification model used in our intrusion detection system. We use an ensemble of classifiers that combines multiple base classification algorithms (Figure 1). Ensemble classifier systems (also called multiple classifier systems) exploit the mutually complementary decision boundaries produced from individual classifiers to improve the performance of the whole. The goal of an ensemble method is to combine different classifiers into a meta-classifier that has a better generalization performance. Ensemble classifier systems have shown to produce favorable results compared to those of single-expert systems for a broad range of applications and under a variety of scenarios [22]. Many studies have applied the diversity of ensemble methods to the intrusion detection problem [23].

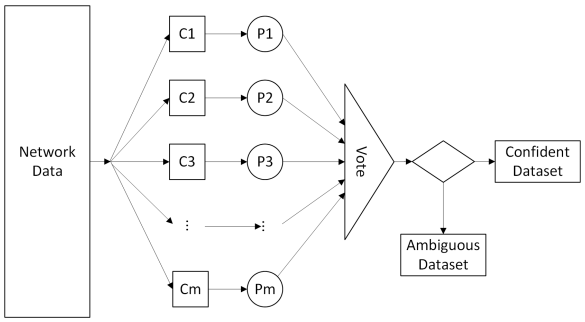


Fig. 1. Design for ensemble classifier system: $C_1, C_2, \cdots C_m$ are base classifiers and $P_1, P_2, \cdots P_m$ are for predicted probability of each individual classifier

We build our ensemble classifier model using base classifiers with different machine learning techniques; Decision Tree [24], [25], Gradient Boosted Tree [26], [27], Random Forest [28], [29] and Multi-layer Perceptron model [30]. We evaluate individual classifier using Receiver Operator Characteristic (ROC) graphs [31].

### A. Classification Algorithms

Classification is one of the most frequently encountered decision making processes [32] and used when an object needs to be assigned into a predefined group or class based on a number of observed attributes related to that object. For a classification problem, a set of $N$ training examples of the form $(X, Y)$ is given, where $Y$ is a discrete class label and $X$ is a vector of $n$ attributes. From these examples a model $Y = f(X)$ is inferred and used to predict the class $Y$ of future examples $X$ with high accuracy. Intrusion detection can be approached as a classification problem that classifies audit events as belonging to a benign or malicious class. The learned model labels or predicts new unseen audit data as belonging to one of them [33]–[35]. An ideal application in intrusion detection would gather sufficient "normal" and "abnormal" audit data for a user or a program to correctly train the classifier and then apply a classification algorithm to make decisions for the (future) audit data as belonging to the normal or abnormal class.

### B. Confidence of Decision

The general formulation of the design problem of an ensemble system is to generate several individual classifiers and then employ some fusion functions (e.g., majority voting) to combine classifier outputs to achieve high performance. In simple majority voting, a class label is selected when it has been predicted by the majority classifiers, that is, received more than 50% of the votes. For multi-class settings, it can be popularity voting where the class label that received the most votes is selected. In this paper, we use the probability of a predicted class label to make a decision in an ensemble of the classifiers. Each classification algorithm returns the probability of a predicted class label. We calculate the average value of the probability of a predicted class label instead of simply counting the number of votes for a class label. The modified version of the majority vote for predicting class labels from probabilities can be written as follows:

$$\hat{Y} =_i \sum_{j=1}^{N} P_{ij} \tag{1}$$

Here, $P_{ij}$ is the predicted probability for a class $i$ by a classifier $j$.

We are interested in not only the final decision from the ensemble classifiers but also the confidence level of the decision. To formalize the confidence level of the final decision in ensemble classifier, we introduce the "Confidence Score" $P$ which is defined as the ratio of a number of the decision vote ($N_D$) to the total number of a vote ($N$) for class label counting:

$$P = \frac{N_D}{N} \tag{2}$$

When the final decision of the classification for a sample is malicious, the number of decision votes is the number of votes for the malicious class. The confidence score for a class label $i$ in the predicted probability approach can be calculated simply by taking the average of the predicted probability of each base classifier:

$$P_i = \frac{1}{N} \sum_{j=1}^{N} P_{ij} \tag{3}$$

where $N$ is the number of base classifiers.

Let's consider simple cases of the decision process for a binary classification problem (e.g., 1 is malicious and 0 is benign) in an ensemble classifier to see the difference from the simple majority voting approach. Five base classifiers are used and each classifier predicts the probability for the class label to be malicious internally as shown in table I and votes to either malicious or benign based on the predicted probability with equal weights. For example, every base classifier predicts a "malicious" label with a predicted probability of 0.6 for a sample data $D_2$. Majority vote method will predict malicious with 100% confidence because all classifiers unanimously vote to "malicious". For sample data $D_3$, the majority vote method will predict malicious with 60% confidence since 3 out of 5 base classifiers vote to "malicious". However, the modified version of the majority vote from predicted probabilities shows different results. For sample data $D_2$, the modified version predicts the class label as malicious as the same as that of the majority vote method but with lower confidence (60%). More different results come for the sample data $D_3$ and $D_4$. Even though more classifiers vote for malicious (benign), final predictions are reversed in the modified version since two of them predict the very low(high) probability for $D_3$ ($D_4$).

| | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $C_5$ | $P_1(\%)$ | $P_2(\%)$ |
|---|---|---|---|---|---|---|---|
| $D_1$ | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 100 | 90 |
| $D_2$ | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 100 | 60 |
| $D_3$ | 0.6 | 0.6 | 0.6 | 0.1 | 0.1 | 60 | 40 |
| $D_4$ | 0.4 | 0.4 | 0.9 | 0.9 | 0.4 | 40 | 60 |
| $D_5$ | 0.4 | 0.4 | 0.4 | 0.4 | 0.4 | 0 | 40 |
| $D_6$ | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0 | 20 |

TABLE I

SAMPLE DATASET TO DEMONSTRATE THE DIFFERENCE IN THE CALCULATION OF CONFIDENCE SCORE BETWEEN MAJORITY VOTING AND THE PREDICTION USING THE AVERAGE VALUE OF THE PROBABILITY OF A PREDICTED CLASS LABEL. $D_1, D_2, \cdots, D_6$ ARE DIFFERENT SAMPLES. $C_1, C_2, \cdots, C_5$ ARE DIFFERENT BASE CLASSIFIERS. VALUES IN EACH CELL ARE THE PREDICTED PROBABILITY BY EACH CLASSIFIER FOR THE "MALICIOUS" LABEL. $P_1$ AND $P_2$ ARE CONFIDENCE SCORES IN % FOR EACH CASE USING MAJORITY VOTING AND AVERAGING PREDICTED PROBABILITY, RESPECTIVELY.

By using the predicted class probabilities instead of counting the class labels for majority voting, we have more fine-grained confidence levels for the prediction. The same 100% confident prediction can be distributed $50 \sim 100\%$ confidence. Some 0% confidences can have higher confidence scores. Since not only the final decision but also the confidence level of the decision of the individual classifier is taken into account, this approach is accurate when each classifier in the ensemble is well calibrated.

## IV. EVALUATION

We use the recent Kyoto2006+ dataset, which has been accumulated for the year 2015. Kyoto 2006+ datasets have built on the real traffic data which are obtained from diverse types of honeypots [36]. During the observation period, there were 6,581,188 normal sessions and 130,135,437 known attack sessions. Among the attack sessions, 2,770 sessions were related to unknown attacks. All traffic data on honeypots were thoroughly inspected using security software since all traffic data captured from honeypots have been collected as attack data [36]. Among traffic data captured from honeypots, however, there are some sessions that did not trigger any alerts but contained shellcodes. These sessions were labeled as "Unknown" attacks.

We analyze the dataset for the year 2015 month by month and present the result for the January 2015 dataset which has 11,218,206 known attack sessions, 1,186,780 normal sessions, and 553 "Unknown" attack sessions. There are more attack data than normal data, which makes the whole dataset imbalanced. We randomly select from the attack dataset and make the both normal and attack dataset balanced. The data labeled as "Unknown" attacks are collected separately and used own

analysis but not used in either training or testing process as shown in table II. Results for each month are similar even though there are small differences in details.

| | |
|---|---|
| Total | 2,374,113 |
| Normal | 1,186,780 |
| Known Attack | 1,186,780 |
| Unknown Attack | 553 |

TABLE II
SUMMARY OF THE TEST DATASET USED IN THE EVALUATION

### A. Feature Sets

The Kyoto 2006+ dataset consists of twenty-four statistical features; fourteen features based on features of the KDD Cup 99 dataset and ten additional features [36]. Among the original 41 features of the KDD Cup 99 dataset, insignificant features and content features were excluded in Kyoto 2006+ dataset, because they are not suitable for network-based intrusion detection systems. Fourteen statistical features, which are significant and essential features, were extracted from honeypot data. In addition to the 14 statistical features, additional 10 features were extracted, which allow investigating more effectively what kinds of attacks happened on networks. They also can be utilized for IDS evaluation with the 14 conventional features, and users are able to extract more features using the additional 10 features. In additional ten features, label feature is included. Label feature indicates whether the session was attacked or not; '1' means the session was normal, '-1' means a known attack was observed in the session, and '-2' means an unknown attack was observed in the session.

### B. Preprocessing

In our evaluation, we don't include IDS, Malware, Ashula detection, and IP address and port number for both source and destination features. For Start Time (indicates when the session was started) data, we divide it into day parting period, "Overnight", "Morning", "Midday", "Afternoon" and "Evenings". There are four categorical features: Service (the connection's service type, e.g., DNS, SSH, etc.) Start Time, Flags (the state of the connection at the time the connection was written), Protocols (TCP, ICMP, etc.). Each categorical feature expressing $m$ possible categorical values is transformed to a value in $\mathrm{R}^m$ using a function $e$ that maps the $j^{th}$ value of the feature to the $j^{th}$ component of an $m$-dimensional vector:

$$e(x_i) = \underbrace{(0, \cdots, 1, \cdots, 0)}_{1\ at\ position\ j} \text{ if } x_i = j \tag{4}$$

The set of features presented in Kyoto 2006+ dataset contains categorical and numerical features of different source and scales. Both the numerical and the categorical features are scaled with respect to each feature's mean $\mu$ and standard deviation $\sigma$:

$$n(x_i) = \frac{x_i - \mu}{\sigma} \tag{5}$$

After encoding all categorical features and excluding irrelevant features, there are 47 feature sets used in the experiment.

The importance of the features in Kyoto 2006+ data sets are evaluated. Using the Random Forest algorithm, we can measure the importance of a feature as the averaged impurity decrease computed from all decision trees in the forest without making any assumptions whether data is linearly separable or not. For the evaluation of the importance of features, we train a forest of 1,000 trees on the Kyoto2006+ dataset and rank the features by their respective importance measures. In Figure 2 we plot the ranks of features in the Kyoto 2006+ dataset by their relative importance; note that the feature importance is normalized so that they sum up to 1.0. First, 10 features among 47 features take 81% of importance.

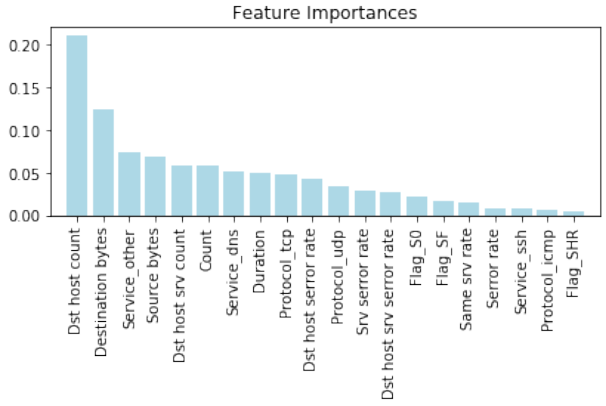


Fig. 2. Relative importance of feature in 2015 data set from Kyoto 2006+ dataset. The feature importances are normalized so that they sum up to 1.0.

In importance feature sets, there are features related to destination characteristics, as shown in the paper [36]. In table III, the first 10 features are listed in order of importance ("Full Dataset"). Four out of ten most important features are related to destination characteristics, such as Dst host count, Destination bytes, Dst host srv count, and Dst host serror rate. Two of them are service types, Service other and Service DNS. Source bytes, Count, Duration, and Protocol TCP are also listed in the ten most important features.

| | Full Dataset | Ambiguous Dataset |
|---|---|---|
| 1 | Dst host count | Destination bytes |
| 2 | Destination bytes | Duration |
| 3 | Service other | Dst host count |
| 4 | Source bytes | Source bytes |
| 5 | Dst host srv count | Count |
| 6 | Count | Dst host srv count |
| 7 | Service dns | Srv serror rate |
| 8 | Duration | Dst host srv serror rate |
| 9 | Protocol tcp | Period Midday |
| 10 | Dst host serror rate | Service dns |

TABLE III
THE IMPORTANCE OF FEATURES FOR THE WHOLE AND AMBIGUOUS DATASET. THE FIRST 10 IMPORTANT FEATURES ARE LISTED IN ORDER OF IMPORTANCE.

### C. Results: Performance of Base Classifiers

The ROC graphs for individual classification algorithms are evaluated on the data and results are shown in figure 3. The ensemble classifiers with Random Forest and Gradient Boost Tree classifier show the best results which attain 99% and 89% True Positive rate, respectively, at 2% False Positive rate. Decision Tree algorithm and MLP show relatively low performance.

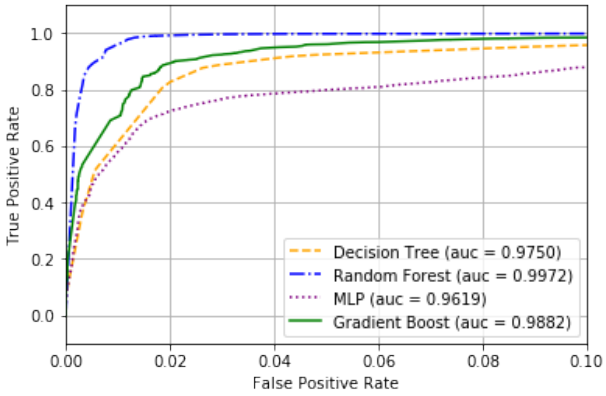


Fig. 3. ROC-curves for classifiers evaluated

Based on the ROC curve, we compute the area under the curve (AUC) to characterize the performance of a classification model. The 10-fold cross-validations were used to calculate ROC-AUC in the training phase and average values (mean) are obtained with standard deviation values (std). Test datasets are used to evaluate the model in the separated dataset to measure its capability to generalize the predictions to unseen data. Table IV shows the summary of results. Results for the test dataset are consistent with those for train datasets within errors and show that there is no indication of over-fittings in the trained models.

| Classifier | mean | std | test |
|---|---|---|---|
| Decision Tree | 0.9749 | $\pm$ 0.0006 | 0.9753 |
| Random Forest | 0.9972 | $\pm$ 0.0001 | 0.9973 |
| MLP | 0.9624 | $\pm$ 0.0003 | 0.9622 |
| Gradient Boost | 0.9880 | $\pm$ 0.0009 | 0.9873 |

TABLE IV
ROC-AUC FOR CLASSIFIERS. COLUMNS "MEAN" AND "STD" ARE RESULTS CALCULATED FOR THE TRAIN DATASET WITH 10-FOLD CROSS-VALIDATIONS AND COLUMN "TEST" ARE FOR THE TEST DATASET

### D. Results: Performance of Ensemble Classifier

We evaluate our model using performance metrics, such as accuracy(ACC), precision(PRE), recall (REC), and F1 score (F1). Accuracy provides general information about how many samples are classified correctly and is calculated as the sum of correct predictions divided by the total number of predictions. Precision and recall are performance metrics that are related to a true positive rate that is especially useful for imbalanced class problems. Precision is the ratio of the true positives to the total predicted positives and recall are the ratio of the true positives to the actual positives that are the sum of true positive and false negative predictions. In practice, often F1 score which is a combination of precision and recall is used. These performance metrics are used to quantify the performance of a model in general.

Results are summarized in table V (Full) with 96.7% accuracy, 96.2% precision, 97.1% recall and 96.6% F1 score. Relatively low value of precision compared to that of recall indicates that there are more false positives (false alerts) than false negatives (missed attacks).

| Dataset | ACC | PRE | REC | F1 |
|---|---|---|---|---|
| Full | 0.9665 | 0.9618 | 0.9711 | 0.9664 |
| Confidence | 0.9915 | 0.9958 | 0.9871 | 0.9914 |
| Ambiguous | 0.7477 | 0.7030 | 0.8267 | 0.7598 |

TABLE V
COMPARISON OF PERFORMANCE METRICS AMONG DATASET WITH ALL (WHOLE), DATASET EXCLUDING AMBIGUOUS DATASET (CONFIDENCE), AND ONLY AMBIGUOUS (AMBIGUOUS) DATASET

For the decision process, we use the probability of a predicted class label, predicted probability $P$, in our model. Figure 4 shows the distribution of the probability for predicted malicious in the full dataset. 1 means confident malicious (attack) and 0 means confident benign (normal) class. Most of the sessions are classified either as confident malicious (1) or as confident benign (0).

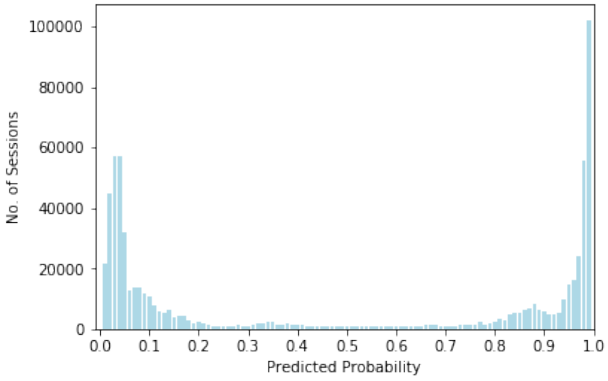


Fig. 4. Distribution of the probability of predicted values for malicious in the full dataset : 1 means malicious (attack) and 0 means benign (normal).

### E. Results: Misclassified Dataset

In our evaluation, 3.4% of sessions in the test dataset are incorrectly classified. We collect those misclassified sessions, called misclassified dataset, and analyze them. Figure 5 shows the prediction probability distribution of sessions in the misclassified dataset. There are two bands where predicted probabilities of misclassified datasets are distributed. One narrow band is at predicted probability $\sim 1.0$ and the other band is rather broad and distributed between $0.2 \sim 0.8$. The misclassified dataset with predicted probability $\sim 1.0$ can be false alerts in our model or missed attacks in the test

dataset. We need more investigation for these misclassified sessions and reserve it for future work. Other misclassified dataset is distributed around the boundary of the malicious and benign dataset and rather broadly distributed with predicted probabilities $0.2 \sim 0.8$.

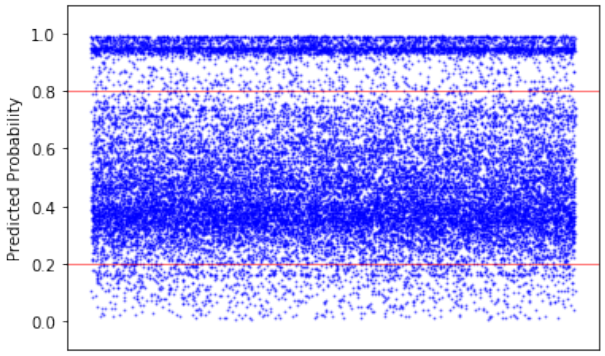


Fig. 5. Distribution of predicted values for misclassified datasets: 1 means attack and 0 means normal. Two horizontal lines (0.2 and 0.8) indicate the window for the ambiguous dataset

We define the ambiguous dataset as those have predicted probability in a range close to the boundary between malicious and benign dataset ($P \approx 0.5$). Since misclassified datasets are distributed around the boundary, most of them will belong to ambiguous datasets. When we select a wider range of ambiguous datasets, more datasets are categorized as ambiguous and more misclassified datasets belong to. Figure 6 presents the ratio of the ambiguous dataset to the total dataset and in the misclassified dataset with different selections of the predicted probability range. The ratio of the ambiguous dataset to the total dataset (red graph) increases rapidly when the predicted probability range is after $[0.2 \sim 0.8]$. But the increase of the ratio in misclassified datasets is saturated after $[0.25 \sim 0.75]$.

We use the ambiguous threshold, $\theta_A$ where $|\theta_A| < 0.5$, to determine the range of *prediction probability* for the ambiguous dataset. The ambiguous threshold value, $\theta_A$, is optimized to maximize the ratio of the ambiguous dataset to the misclassified dataset and to minimize the portion of the ambiguous dataset in the entire dataset. Let the ratio of the ambiguous dataset to the misclassified dataset as a function of the ambiguous threshold $f_1(\theta_A)$ and the portion of the ambiguous dataset in the entire dataset as $f_2(\theta_A)$. The ambiguous threshold $\theta_A$ is determined by maximizing

$$f(\theta_A) = f_1(\theta_A) + (1 - f_2(\theta_A)) \tag{6}$$

Figure 7 presents the $f_1(\theta_A) - f_2(\theta_A)$ obtained in our experiments as a function of $\theta_A$. It shows the maximum is reached at $\theta_A = 0.2$. We set $\theta_A = 0.2$ and the range for ambiguous dataset as $0.2 < P < 0.8$ based on this observation. With this selection, the ambiguous dataset contains 77.2% of the misclassified dataset but is only 10.24% of the total dataset.

We regroup the test dataset into "Confident" (either malicious or benign) and "Ambiguous" and re-evaluate our model

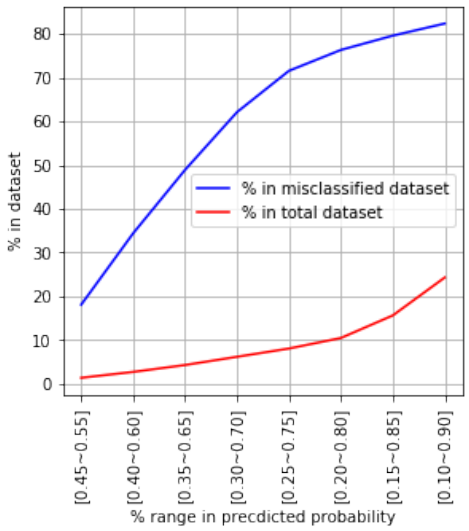


Fig. 6. The portion of ambiguous dataset in total dataset and in misclassified dataset with different selections of the predicted probability

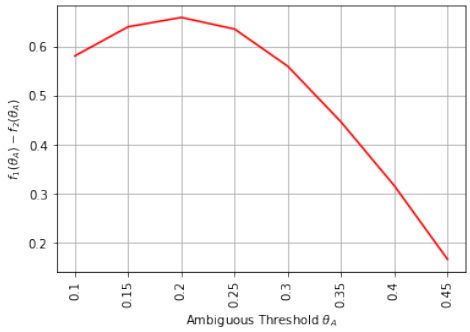


Fig. 7. The dependency of $f_1(\theta_A) - f_2(\theta_A)$ on ambiguous threshold $\theta_A$.

with the confident and ambiguous dataset separately. Results are summarized in table V as Confident and Ambiguous, respectively. The accuracy of the confident dataset is improved by 2.5% to 99.2%. On the other hand, it is decreased for the ambiguous dataset to 74.8%. The same changes can be seen in other performance metrics. This result shows that the ambiguous dataset contains the most uncertain samples. In section IV-G, we re-train the ambiguous dataset alone and resolve these misclassified sessions.

### *F. Results: Unknown Datasets*

In our test dataset, there are 553 "Unknown" attacks which are not included in the training phase of the model. We use our model and classify these "Unknown" datasets. Figure 8 shows the distribution of the predicted probabilities. We can see two interesting results for the dataset labeled "Unknown". First, our results show that all of the unknown datasets are classified as attacks in our model (prediction probability $>$ 0.5). Second, most of them are categorized as a malicious dataset with high confidence prediction probability $> 0.8$). Thus, any novel attacks which are not filtered by the existing

intrusion detection systems used in Kyoto 2006 + dataset are classified as malicious and detected in our model.

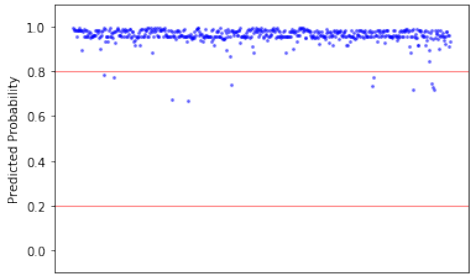


Fig. 8. Distribution of predicted values for unknown datasets: 1 means attack and 0 means normal. Two horizontal lines (0.2 and 0.8) indicate the window for ambiguous dataset

### G. Results: Re-Evaluation of Ambiguous Dataset

In this section, we show that the performance of intrusion detection can be improved by re-evaluating ambiguous datasets with minimum overhead. Since the number of ambiguous datasets is much less than that of the full datasets (10% of the full dataset in our case), the re-evaluation process is light in resource usage and fast in the computing process. We first review the importance of features for the ambiguous dataset. Table III compares the first 10 important features in two datasets, full and ambiguous dataset. Three different features are included in the 10 important features for the ambiguous dataset and the order in the importance of features also gets changed. This difference indicates that a separate model has to be learned (trained) for the optimal classification of the ambiguous dataset.

We re-train the ambiguous dataset and rebuild a model. Table VI shows the performance metrics of retraining results for the ambiguous dataset and compare to previous values. In re-evaluation, the accuracy (ACC), precision (PRE), recall (REC), and F1 scores (F1) are improved by 15%, 20%, 9%, and 15% respectively. We can improve performance and resolve many misclassified samples in the re-training process.

| | ACC | PRE | REC | F1 |
|---|---|---|---|---|
| Before | 0.7477 | 0.7030 | 0.8267 | 0.7598 |
| After | 0.9029 | 0.9077 | 0.9194 | 0.9135 |

TABLE VI
COMPARISON OF PERFORMANCE METRICS FOR AMBIGUOUS DATASETS BETWEEN BEFORE AND AFTER RE-EVALUATION.

## V. Related Works

Machine learning and data mining approaches for intrusion detection derive associative rules from available sample data and use statistical techniques to discover subtle relationships between data items and to find consistent and useful patterns that describe programs and user behavior. There are many survey papers [11]–[13], [16], [37] which summarize achievements, current trends, challenges, even limitations in various machine learning-based approaches. Because of simplicity, high detection accuracy, and fast adaptation, many supervised learning algorithms have been adopted in intrusion detection systems. Decision Trees are one of the most commonly used supervised learning algorithms in IDS [38]. The advantages of a decision tree are intuitive knowledge representation, high classification accuracy, and simple implementation. However, the larger the tree, the less intuitive the knowledge representation is because it is difficult to extract the rules for deeper and wider trees. Large trees often have high classification accuracy, but not a high generalization. Kruegel and Toth [33] used decision trees in Snort's misuse detection engine [9]. Zhang et al. [35] applied a Random Forest classifier to anomaly detection where an anomaly detector was used to feed the second intrusion classifier. Sahu et.al. used Decision Tree (J48) algorithm to classify the network packet in Kyoto 2006+ data set that can be used for NIDS [39].

The Naive Bayes classifier is a well-known machine learning technique and is also used for machine learning-based IDS. However, because Naive Bayes assumes conditional independence of data capabilities, the correlated features of network data for intrusion detection can degrade performance. Amor et al. [40] and Panda et al. [34] have used the Naive Bayes classifier and applied it to the KDD 1999 dataset for training and testing. In [41], authors present a method that automatically extracts only unknown attacks from anomaly-based intrusion detection system alerts. They modified the existing feature extraction method with new features; duration, source bytes, and destination bytes, and applied one-class SVM to them. Authors in [19] found, however, that machine learning-based intrusion detection system is rarely used in operational "real world" settings despite extensive academic research compared to other intrusion detection methods. This indicates that finding attacks might be fundamentally different from tasks in other applications, making it much more difficult for the intrusion detection community to adopt machine learning effectively. There are many types of ensembles proposed in the machine learning literature. With respect to architecture, individual classifiers can, in general, be structured in forms of parallel (e.g., bagging), sequential (e.g., boosting), or hybrid [42]. For making a decision, the composer of classifiers can apply various mechanisms such as majority voting, Bayesian combination, distribution summation, entropy weighting, and so on [23], [42].

## VI. Conclusion

In this paper, we propose a method to improve the performance of machine learning-based intrusion detection systems and reduce the rate of false alerts. Our focus is the ambiguity of the classification model which leads to misclassification and false alerts. We look for a practical approach to reduce these ambiguities in the domain-specific environment where usage patterns, as well as attack patterns, are continuously and rapidly changed. We take the strategy to choose a set of data that are the most uncertain and thus most informative and improve the accuracy of classification in the subsequent

analysis. Those uncertain data are collected as an ambiguous dataset. The goal is to extract the most uncertain dataset and re-evaluate them to reduce the ambiguity and improve the performance of the model. We show that the ambiguous dataset contains 77.2 % of misclassified data in our model. We can resolve many misclassified data in the re-training process. We also evaluate the "Unknown" attacks in collected Kyoto 2006+ network traffic data. Our model predicts those data as malicious with high confidence. In future work, we will continue to advance the proposed framework using advanced techniques, such as data stream mining for real-time processing, secure adversarial machine learning, and timely and intelligent response systems.

## ACKNOWLEDGMENT

This research was supported in part by Colorado State Bill 18-086.

## REFERENCES

[1] T. Lane and C. E. Brodley, "An application of machine learning to anomaly detection," in *Proceedings of the 20th National Information Systems Security Conference*, vol. 377. Baltimore, USA, 1997, pp. 366–380.

[2] A. K. Ghosh, J. Wanken, and F. Charron, "Detecting anomalous and unknown intrusions against programs," in *Computer Security Applications Conference, 1998. Proceedings. 14th Annual*. IEEE, 1998, pp. 259–267.

[3] J. Cannady, "Artificial neural networks for misuse detection," in *National information systems security conference*, 1998, pp. 368–81.

[4] C. Sinclair, L. Pierce, and S. Matzner, "An application of machine learning to network intrusion detection," in *Computer Security Applications Conference, 1999.(ACSAC'99) Proceedings. 15th Annual*. IEEE, 1999, pp. 371–377.

[5] S. Kumar and E. H. Spafford, "A software architecture to support misuse intrusion detection," 1995.

[6] K. Ilgun, R. A. Kemmerer, and P. A. Porras, "State transition analysis: A rule-based intrusion detection approach," *IEEE transactions on software engineering*, vol. 21, no. 3, pp. 181–199, 1995.

[7] T. F. Lunt, A. Tamaru, and F. Gillham, *A real-time intrusion-detection expert system (IDES)*. SRI International. Computer Science Laboratory, 1992.

[8] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer networks*, vol. 31, no. 23, pp. 2435–2463, 1999.

[9] M. Roesch *et al.*, "Snort: Lightweight intrusion detection for networks." in *Lisa*, vol. 99, no. 1, 1999, pp. 229–238.

[10] S. Mukkamala, A. Sung, and A. Abraham, "Cyber security challenges: Designing efficient intrusion detection systems and antivirus tools," *Vemuri, V. Rao, Enhancing Computer Security with Smart Technology.(Auerbach, 2006)*, pp. 125–163, 2005.

[11] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[12] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *computers & security*, vol. 28, no. 1, pp. 18–28, 2009.

[13] S. X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review," *Applied Soft Computing*, vol. 10, no. 1, pp. 1–35, 2010.

[14] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," *Ieee communications surveys & tutorials*, vol. 16, no. 1, pp. 303–336, 2014.

[15] S. Dua and X. Du, *Data mining and machine learning in cybersecurity*. CRC press, 2016.

[16] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[17] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on software engineering*, no. 2, pp. 222–232, 1987.

[18] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, "Network intrusion detection," *IEEE network*, vol. 8, no. 3, pp. 26–41, 1994.

[19] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 305–316.

[20] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Machine learning*, vol. 23, no. 1, pp. 69–101, 1996.

[21] T. Lane and C. E. Brodley, "Approaches to online learning and concept drift for user identification in computer security." in *KDD*, 1998, pp. 259–263.

[22] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits and systems magazine*, vol. 6, no. 3, pp. 21–45, 2006.

[23] G. Giacinto, F. Roli, and L. Didaci, "Fusion of multiple classifiers for intrusion detection in computer networks," *Pattern recognition letters*, vol. 24, no. 12, pp. 1795–1803, 2003.

[24] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[25] ——, *C4. 5: programs for machine learning*. Elsevier, 2014.

[26] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[27] ——, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[28] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[29] ——, "Out-of-bag estimation," 1996.

[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[31] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[32] T. M. Mitchell, "Machine learning and data mining," *Communications of the ACM*, vol. 42, no. 11, pp. 30–36, 1999.

[33] C. Kruegel and T. Toth, "Using decision trees to improve signature-based intrusion detection," in *Recent Advances in Intrusion Detection*. Springer, 2003, pp. 173–191.

[34] M. Panda and M. R. Patra, "Network intrusion detection using naive bayes," *International journal of computer science and network security*, vol. 7, no. 12, pp. 258–263, 2007.

[35] J. Zhang, M. Zulkernine, and A. Haque, "Random-forests-based network intrusion detection systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 5, pp. 649–659, 2008.

[36] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao, "Statistical analysis of honeypot data and building of kyoto 2006+ dataset for nids evaluation," in *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*. ACM, 2011, pp. 29–36.

[37] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, "An overview of ip flow-based intrusion detection." *IEEE Communications Surveys and Tutorials*, vol. 12, no. 3, pp. 343–356, 2010.

[38] J.-H. Lee, J.-H. Lee, S.-G. Sohn, J.-H. Ryu, and T.-M. Chung, "Effective value of decision tree with kdd 99 intrusion detection datasets for intrusion detection system," in *Advanced Communication Technology, 2008. ICACT 2008. 10th International Conference on*, vol. 2. IEEE, 2008, pp. 1170–1175.

[39] S. Sahu and B. M. Mehtre, "Network intrusion detection system using j48 decision tree," in *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*. IEEE, 2015, pp. 2023–2026.

[40] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive bayes vs decision trees in intrusion detection systems," in *Proceedings of the 2004 ACM symposium on Applied computing*. ACM, 2004, pp. 420–424.

[41] M. Sato, H. Yamaki, and H. Takakura, "Unknown attacks detection using feature extraction from anomaly-based ids alerts," in *Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on*. IEEE, 2012, pp. 273–277.

[42] L. Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1, pp. 1–39, 2010.