When a Tree Falls: Using Diversity in Ensemble Classifiers to Identify Evasion in Malware Detectors

Charles Smutz
George Mason University
csmutz@gmu.edu

Angelos Stavrou
George Mason University
astavrou@gmu.edu

***Abstract*—Machine learning classifiers are a vital component of modern malware and intrusion detection systems. However, past studies have shown that classifier based detection systems are susceptible to evasion attacks in practice. Improving the evasion resistance of learning based systems is an open problem. To address this, we introduce a novel method for identifying the observations on which an ensemble classifier performs poorly. During detection, when a sufficient number of votes from individual classifiers disagree, the ensemble classifier prediction is shown to be unreliable. The proposed method, ensemble classifier mutual agreement analysis, allows the detection of many forms of classifier evasion without additional external ground truth.**

We evaluate our approach using PDFrate, a PDF malware detector. Applying our method to data taken from a real network, we show that the vast majority of predictions can be made with high ensemble classifier agreement. However, most classifier evasion attempts, including nine targeted mimicry scenarios from two recent studies, are given an outcome of uncertain indicating that these observations cannot be given a reliable prediction by the classifier. To show the general applicability of our approach, we tested it against the Drebin Android malware detector where an uncertain prediction was correctly given to the majority of novel attacks. Our evaluation includes over 100,000 PDF documents and 100,000 Android applications. Furthermore, we show that our approach can be generalized to weaken the effectiveness of the Gradient Descent and Kernel Density Estimation attacks against Support Vector Machines. We discovered that feature bagging is the most important property for enabling ensemble classifier diversity based evasion detection.

I. Introduction

The use of machine learning has emerged as one of the primary techniques employed to address a wide range of malfeasance and malicious activities. Applications of machine learning include clustering of malware families [7], [20], detection of malicious downloads [12], [34], detection of account misuse in social networks [14], [44], and detection of commonly exploited file formats such as Java archives [36] and documents [24], [25], [39]. Moreover, statistical or machine learning techniques have been used successfully for years to identify SPAM [11], [21], [35].

NDSS '16, 21-24 February 2016, San Diego, CA, USA
 ISBN 1-891562-41-X
http://dx.doi.org/10.14722/ndss.2016.23078

One of the main weaknesses of systems that employ machine learning classification in adversarial environments is their susceptibility to evasion attacks. With evasion attacks, we refer to the classes of attacks that take advantage of knowledge of how the machine learning system operates, and in many cases utilize access to the training set and features, to evade detection passively or actively [8], [9], [15], [33], [45].

A common technique used in evasion attacks against machine learners is mimicry. Mimicry attacks thwart detection by making the attack data appear benign according to the model used by the intrusion detection system. Often this is achieved by hiding overtly malicious content through encoding or encryption [28], [42] or minimizing the footprint of malicious content through data misuse or code re-use attacks [17], [37]. For instance, content aligning with a benign observation is added to cover up or drown out the malicious content. Many detection systems are evaded by exploiting differences in the detection system and the systems being protected [16], [19]. Even if operational details of defense systems are kept secret, enough knowledge to conduct evasion can often be obtained solely from external testing [18]. With all of these potential evasion vectors, preventing detection evasion remains an open problem.

Our approach is not to prevent all possible evasion attacks, but to introduce a mechanism that provides detection of poor classifier performance. We analyze the use of introspection in an ensemble classifier to detect when the classifier provides unreliable results at classification time. The use of ensemble classifier mutual agreement analysis relies on the intuition that when individual classifiers in an ensemble vote for the same prediction, the prediction is likely to be accurate. When a sufficient number of the votes are in opposition, then the classifier prediction is not trustworthy. In this state of internal classifier disagreement, the detector returns the outcome of uncertain instead of a prediction of benign or malicious. In operation, confidence in the predictions of the classifier is improved at the cost of a small portion of the samples being labeled as uncertain, indicating that the classifier is not fit to provide an accurate response. This separation of accurate predictions from uncertain predictions is possible because the majority of the misclassifications, including evasion attempts, have a classifier voting score distribution distinct from the accurate predictions.

To evaluate our technique, we applied mutual agreement analysis to two well-studied malware detection systems: PDFrate [40] and Drebin [4]. PDFrate uses features derived from document structure and metadata fed into a Random

Forest classifier to detect Trojan PDFs. PDFrate is used in real world intrusion detection systems and can be evaluated by the public through submissions to pdfrate.com. PDFrate was selected because it is publicly accessible, well documented, uses an ensemble classifier which returns the raw voting score, and has been subjected to multiple recently published mimicry attacks [26], [27], [43]. Our evaluation includes over 100,000 documents sourced from an operational environment and hundreds of malicious documents in nine unique evasion scenarios from two independent evasion studies. To demonstrate the general applicability of our approach, we apply mutual agreement analysis to the Drebin Android malware detector using over 100,000 applications, including over 5,000 malicious applications in over 20 labeled malware families. We find that mutual agreement analysis enables the identification of novel malware that would otherwise not be detected reliably.

In building an evasion resistant ensemble using Support Vector Machines (SVM) as base classifiers, we find that feature bagging, or constructing many individual classifiers with randomized subsets of the whole feature set, is crucial to providing this discriminatory power. Using this method, we counter the Gradient Descent and Kernel Density Estimation (GD-KDE) attack, which is highly successful against a traditional SVM classifier.

II. Related Work

Adversarial learning is an active research topic [18]. Some studies have proposed methods for creating effective classifier based intrusion detection systems [6], [15], [41]. Many studies have addressed the importance of data sanitization or adversarial influence at training time [5], [13], [23], [30]. Yet others focus on evasion of the deployed classifier [8], [27], [43]. We also focus on evasion of a classifier during operation, but instead of focusing on strategies for evasion, we propose a means of detecting these evasion attempts.

Estimation of confidence based on knowledge of a population has long been foundational to statical methods [31]. Contemporary research has demonstrated how these confidence estimates can be applied to a machine learning based classifier deployed in an online setting [38]. However, since these approaches rely on new observations matching the distributions of training samples for which ground truth is known, they are not applicable to intrusion detection systems which face novel observations and mimicry attacks. Rather than seeking to quantify the overall accuracy of a classifier, we identify the individual observations for which a classifier cannot provide a reliable response. Our approach makes use of data already provided by the classifier, without additional appeal to ground truth or independent outlier analysis.

Recent work has demonstrated that the diversity in ensemble classifiers can improve malware detection rates [22], [29], [46], [47]. Few studies, however, advance practical strategies for detection of evasion attempts against these ensemble classifiers. Chinvale et al. proposed the use of mutual agreement between a small number of independent SPAM filters to optimize individual classifier re-training necessary due to drift in input data [11]. We extend this approach to introspection of ensemble classifiers in order to provide a per observation confidence estimate at test time. We differ fundamentally from Chinvale et al. in that they use the majority result of their ensembles as ground truth for re-training of individual classifiers while we focus on identifying the specific examples where the ensemble prediction is not trustworthy. In short, Chinvale et al. use diversity in ensembles to improve classifier performance. We use diversity to identify when resorting to external ground truth is necessary. We study the factors that enable diversity based confidence estimates in ensembles using variations of bagging. Going beyond natural drift or novel attacks, we apply mutual agreement analysis to targeted evasion attempts consisting of attacks against feature extractors, training data, and specific classifiers.

Our empirical evaluation relies on current research in machine learning based malware detectors [4], [40] and mimicry attacks [26], [27], [43]. We seek to mitigate evasion in these malware detectors.

III. Background

We apply mutual agreement analysis to two malware detectors: PDFrate and Drebin. Our study of PDFrate includes two mimicry attacks against PDFrate: Mimicus and Reverse Mimicry.

A. PDFrate

PDFrate is a machine learning based malware detector operating on PDF documents. The pdfrate.com website allows user submissions and returns ratings for these submitted files. PDFrate is useful for this study because the underlying mechanisms are well documented [40], it is openly available for online attack, and it provides considerable information about each submitted PDF. Because of this transparency, PDFrate has been the target of practical mimicry attack studies [26], [27], [43].

PDFrate classifies PDF documents based on analysis of their structural and metadata attributes. Risk factors for a malicious document include items such as existence of Javascript objects or improperly formatted timestamps. On the other hand, benign documents contain inert content such as text content or font objects. Basic structural and metadata information is extracted using regular expressions applied to the raw document. This small subset of structural information taken from the document is presented to the user in the document scan report. From this base information, features are extracted. Examples of features include the number of Javascript objects and the relative position of the end of file marker in the document. All told, 202 features are used.

Random Forests is used as the classifier in PDFrate. A Random Forest is constructed of hundreds or thousands of individual classification trees. For each tree, a subset of the training set is used for construction. At each node in the tree, a subset of features is tried to determine which feature and threshold best divides the classes. This process is repeated until each leaf node contains a single class. Hence, in a Random Forest, each tree is based on both a randomly selected subset of the training data and the features. New observations are run through each tree, the leaf node dictating the vote for that tree. A discriminating characteristic of PDFrate is that it provides a score or rating instead of a simple benign/malicious

determination. The score provided by PDFrate is the portion of trees that voted for the positive (malicious) class.

The PDFrate website provides scores from classifiers based on multiple training sets. The Contagio data set is taken from a widely available data set designated for researchers [32]. It contains 10,000 documents, evenly split between benign and malicious. The list of documents in this data set is published openly. The second data set was composed by researchers at George Mason University and is called the University data set. It contains a much larger number of documents, over 100,000, but the exact composition of the training set is not published. We use both of these training sets, and the classifiers derived from them, in this study.

B. Mimicus

Mimicus [1] is a framework for performing mimicry attacks against PDFrate. It is the implementation of what is described by Šrndić and Laskov as "the first empirical security evaluation of a deployed learning-based system" [43]. It is an independent, comprehensive, and openly available framework for attacks against the online implementation of PDFrate.

Mimicus implements mimicry attacks by modifying existing malicious documents to appear more like benign documents. Mimicus adds markers for additional structural and metadata items to documents. These additions do not involve adding actual content that is interpreted by a standards-conforming PDF reader, but rather these additions exploit a weakness in the feature extractor of PDFrate. The extraneous PDF attributes are added in slack, or unused space, immediately preceding the document trailer (structure at the end of the document), which is not prohibited by the PDF specification. This approach provides considerable flexibility in the evasion attack as the additional elements do not have to be valid. Mimicus enables a simple process for the attacker. The attacker constructs a malicious document without concern for PDFrate evasion. Mimicus then adds the necessary decoy structural elements. This mimicry attack only adds fake elements to the document file–no existing elements are removed or modified.

Mimicus constructs these decoy elements by comparing a malicious document to multiple different benign documents. The feature vectors for the malicious documents are adjusted to mirror the feature vectors for the benign documents. These adjustments are bounded by the modification approach Mimicus uses. The candidate mimicry feature vectors are run through a local PDFrate replica to determine the scores. The best feature vector is selected. That feature vector is used as the goal in modifying the original malicious document by adding decoy structural and metadata elements. Due to interrelated features and other complications, it is not feasible to construct a final mimicry malicious document that exactly matches the target mimicry feature vector. The resulting malicious document has a feature vector that is somewhere between that of the original Trojan document and that of a benign document. After the mimicry document is created, it is submitted to pdfrate.com for evaluation.

An important observation of the Mimicus study is that the interdependency of PDFrate's features make mimicry attacks more difficult because modifying one feature necessarily affects other features. It is generally accepted that irrelevant or redundant features are not desirable for machine learning methods. However, in the case of PDFrate, redundant features appear to make evasion attacks, like those implemented by Mimicus, more difficult by making construction of a PDF matching a target feature vector more difficult.

The Mimicus attack model requires knowledge of the feature set used by PDFrate. The premise is that for a mimicry attack to be successful, at least knowledge of the type of features is necessary. Also, since this attack leverages a difference between normal PDF readers and the PDFrate feature extractor, knowledge of how to exploit this difference is also necessary. Hence, all Mimicus attack scenarios are labeled with an "F", indicating that the attacker used knowledge of the feature set.

Relying on the common basis of the feature extraction, the Mimicus attacks demonstrate various levels of knowledge used by the attacker. In situations where the training data and classifier are known by the attacker, replicas that are very close to the original are used. When an attacker with limited system knowledge is modeled, reasonable substitutes are employed. The labels "T" and "C" are used to denote attacker knowledge of training data and classifier, respectively. Hence, an attack scenario with the label "FTC" denotes attacker knowledge of all three major facets of PDFrate.

The training set used by the Contagio classifier of PDFrate is publicly documented and is readily available to researchers. Hence, in attack scenarios where the training data is known by the attacker, the same data set is used by PDFrate and Mimicus. For scenarios where the attacker has no knowledge of the training set, Šrndić and Laskov compiled a surrogate training set with malicious documents sourced from VirusTotal and benign documents sourced from the Internet. In addition, they selected 100 malicious documents from within the Contagio training set for the baseline attack documents. To allow reproduction of results, all of the data sets used by Šrndić and Laskov are documented.

Lastly, to complete the offline PDFrate replica, Šrndić and Laskov used a Random Forests classifier when knowledge of the classifier was known, and a Support Vector Machine classifier to simulate the case of the naive attacker. The Mimicus study shows that when all three particulars of PDFrate are spoofed, the result is nearly identical scores from the PDFrate online and the Mimicus offline classifier, despite various implementation differences. Mimicus also implements a GD-KDE attack which seeks to attack the SVM surrogate classifier directly. This attack does not apply to Random Forests classifiers, and therefore does not directly apply to PDFrate.

C. Reverse Mimicry

Maiorca et al. also study evasion against PDFrate and other PDF document classifiers [26], [27]. They advance the Reverse Mimicry technique. Instead of adding content to a malicious document to make it appear benign (as Mimicus does), they embed malicious content into a benign PDF, taking care to modify as little as possible. The Reverse Mimicry attack implements an independent evasion approach against PDFrate.

Three different evasion scenarios are advanced by Maiorca et al. In the EXEembed scenario, a malicious executable is

implanted in an existing benign PDF document. The malware is executed when the document is opened. These documents utilize CVE-2010-1240. In the PDFembed scenario, a malicious PDF is embedded into a benign PDF. These embedded documents are rendered automatically when the document is opened. For evaluation, Maiorca et al. embedded a document leveraging CVE-2009-0927 into existing benign PDF documents. Lastly, in the JSinject scenario, malicious Javascript, the same used in the PDFembed embedded document, is injected directly into the root benign document.

In order to evade detection, the Reverse Mimicry attacks focus on changing the document structure as little as possible. For example, in the EXEembed attack, a new logical version of the PDF is constructed with few new structural elements, but all the content from the original PDF is left in the file. A compliant reader will not display the content associated with the previous version of the document, but the artifacts will be analyzed by the feature extractor of PDFrate and similar detectors.

In addition to minimizing the structural artifacts of the malcode injection, Maiorca et al. make use of PDF encoding, especially stream compression, to hide the inserted content. For example, in the PDFembed attack, the malicious document is embedded in a compressed PDF stream. Detection tools, such as PDFrate, that do not decompress the PDF streams are not able to extract features from the embedded malicious PDF.

The Mimicus and Reverse Mimicry attacks use two separate paths to evade PDFrate. Mimicus uses addition of decoy objects that would not be processed by a normal PDF reader but are parsed by the simple regular expression based processing of PDFrate. The Reverse Mimicry attacks, on the other hand, use valid PDF constructs to minimize and hide malicious indicators. Mimicus operates by adding camouflage while the Reverse Mimicry attack seeks to make the malicious elements stealthy. Mimicus leverages extensive knowledge of PDFrate while the Reverse Mimicry approach uses data hiding techniques peculiar to the PDF file format.

D. Drebin Android Malware Detector

Ensemble classifier mutual agreement analysis should be applicable to all situations where evasion is possible, including other malware classifiers. We evaluated the utility of mutual agreement analysis on the Drebin Android malware detector [4]. Drebin complements PDFrate because it operates on a software package instead of a document and utilizes many string based features instead of numerical features. Since the data used in the original Drebin study is available to researchers, we use this data for our evaluation.

Drebin operates by performing a quick scan to extract features from the Android application manifest and disassembled code. These features are formatted as strings. Features extracted from the manifest include the names/values of hardware components, requested permissions, application components, and intents (message framework). The values of API calls, used permissions, and network addresses/URLs are taken from the disassembled code. The string values are mapped into a binary feature vector containing over 500,000 unique values.

A linear SVM is trained offline and used to provide weights (distance from hyperplane) for each feature observed during classification. This per predictor weight is combined to provide an overall score and compared to a threshold to determine the outcome. Due to this scheme, Drebin provides a maliciousness score and can identify variables that contribute to this score. Drebin is evaluated with over 100,000 benign and 5,000 malicious samples, providing a false positive rate of 1% and a malware detection rate of nearly 94%.

IV. Approach

An ensemble classifier is constructed from many base classifiers. To provide meaningful diversity in the ensemble, each individual classifier is constructed using mechanisms such as random sampling (bagging) of training data and features. Typically, the result is combined by voting, where each independent classifier gets an equal vote. The count of votes are summed to generate a score. If the score is over 50%, then the observation is labeled malicious. Otherwise, the result is benign.

Ensembles have been shown to improve accuracy in many use cases, including malware detection. However, we have found the primary advantage of ensemble classifiers to be that they can provide a measure of internal coherence which serves as an estimate of the classifier's confidence of individual predictions.

In a well preforming ensemble, the majority of individual classifiers provide the same vote. If the base classifiers provide conflicting votes, then the ensemble is in a state of disagreement and the prediction is less trustworthy. The agreement or disagreement in voting of individual contributors in the ensemble provides an estimate of the confidence of the prediction of the ensemble.

A classifier may not be able to provide an accurate response for some observations. For example, when a 50/50 vote split occurs in traditional ensembles, a prediction is provided using a method such as random selection. Most applications will treat a randomly selected prediction when the classifier is in total disagreement the same as one where all contributors vote for the same class. However, in the case of complete disagreement, the only reasonable interpretation is that the classifier cannot make a competent prediction.

Diversity in ensemble classifiers is the core attribute that facilitates mutual agreement based confidence estimates. This diversity is caused by extrapolation in individual classifiers. Barring limitations of the classifier scheme and quality of features, when an observation is close to samples in the training set, the classification is well supported and should be accurate. However, as new observations diverge farther from training samples, the classifier is forced to extrapolate. For ensemble classifiers which employ bagging effectively, the farther new observations are from classifier training, the more disagreement there will be in the ensemble.

This diversity in extrapolation is observed in the Random Forest based classifiers used in PDFrate. Table I shows the classification performance of the first 25 trees (out of 1000) in the Contagio classifier applied to various mimicry attacks. Performance is reported relative to the forest average number of votes for the correct class, dividing at $\pm$ 0.5 standard deviations. It is observed that the vast majority of the trees have all

TABLE I. RELATIVE PERFORMANCE OF INDIVIDUAL TREES IN CONTAGIO CLASSIFIER INDICATED AS ABOVE (+), BELOW (-), OR WITHIN (0) 0.5 STANDARD DEVIATIONS OF FOREST AVERAGE

Evasion Scenario	Individual Tree Performance																								
F_mimicry	0	+	+	-	0	0	-	+	0	+	-	0	+	-	+	0	0	0	-	+	+	+	+	-	-
FC_mimicry	+	+	+	-	+	0	-	+	0	+	-	-	+	0	0	0	0	0	0	+	+	+	0	-	-
FT_mimicry	0	+	+	-	-	0	0	+	0	0	-	0	0	0	+	-	-	0	0	+	+	0	0	-	0
FTC_mimicry	-	+	+	-	0	+	0	-	-	+	0	-	+	0	+	+	-	+	0	+	-	+	0	-	0
F_gdkde	-	+	+	+	+	+	-	-	+	+	0	0	+	-	+	-	-	+	-	+	0	-	-	-	0
FT_gdkde	+	+	+	+	0	+	-	-	+	+	+	-	+	+	-	-	-	0	-	0	-	+	+	-	0
JSinject	+	-	-	0	+	+	-	0	+	+	+	0	0	+	0	0	0	+	-	0	+	-	0	+	-
PDFembed	0	-	-	+	0	0	0	-	-	-	-	+	+	-	-	-	-	-	0	-	-	-	-	+	-
EXEembed	-	0	0	-	-	-	+	0	+	0	-	-	-	+	0	+	-	-	+	+	-	0	0	-	-

TABLE II. ENSEMBLE CLASSIFIER OUTCOMES

Voting Score	Outcome		Evasion Type
[0,25]	Benign		Strong Evasion
(25,50)	Uncertain	(Benign)	Weak Evasion
[50,75)		(Malicious)	
[75, 100]	Malicious		No Evasion

three outcomes depending upon the evasion scenario: average (0), below average (-), and above average (+). Hence, when applied to data distant from the training data, the accuracy of each tree varies widely between observations. There are no universally strong or weak trees. The random noise present when extrapolating far from the training data is what enables mutual agreement analysis.

To apply mutual agreement analysis generally, we propose a new outcome, uncertain, in addition to the predictions of benign and malicious. For example, instead of splitting the vote region in half, we split it into 4 quadrants. In the 0% to 25% region, the majority of the votes agree that the result is negative (benign). Similarly, in the 75% to 100% region, the majority of the votes agree that the result is positive (malicious). However, if the score is between 25% and 75%, the individual classifiers disagree and the outcome is uncertain. To support comparison with simple ensemble voting predictions, this area can be split into the other two quadrants: uncertain (benign) from 25% - 50% and uncertain (malicious) from 50% - 75%. These classification outcomes are demonstrated in Table II. The uncertain rate (UR) is the portion of observations that fall within the uncertain range.

To be more precise about this concept, we introduce a metric to quantify the agreement between individual votes in an ensemble classifier:

$$A = |v - 0.5| * 2$$

Where A is the ensemble classifier mutual agreement rate and v is the portion of votes for either of the classes. This function is demonstrated in Figure 1, which also shows the classifier outcomes resulting from a 50% mutual agreement threshold. The end and middle points drive the general shape of this function. If the classifier vote ratio is either 0 or 1, then the classifier has full agreement on the result and the mutual agreement should be 1 (or 100%). If the classifier is split with 0.5 of the votes for each class, then the mutual agreement should be at the minimum of 0 (or 0%). As long as a single threshold is used, it is not important what shape is used for the lines between these end and middle points– any continuous curve would allow the selection of a given

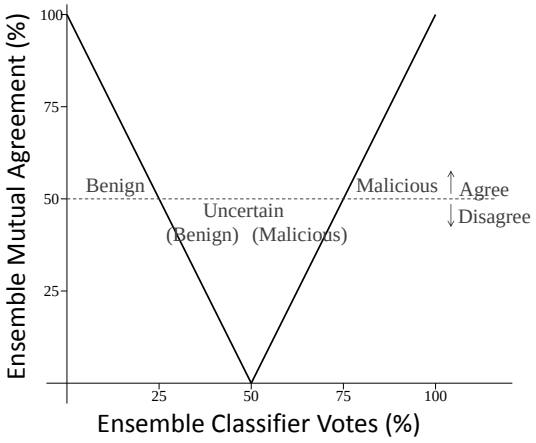

Fig. 1. Mutual Agreement Based on Ensemble Vote Result

threshold on the classifier vote scores. The function need not follow the distribution of scores, for example. We choose a linear function because it is straightforward.

The threshold for mutual agreement is the boundary above which the classifier is said to be in a state of ensemble agreement, and the resulting classification should be considered valid. Below this mutual agreement rating, the classification is specious. We use the boundary of 50% throughout most of this paper. However, this value should be adjusted by the operator. Decreasing this threshold decreases the number of observations in the disagreement or uncertain classification zone. Tuning of this threshold is discussed in detail in Section VI.

Mutual agreement analysis is effective at identifying the specific samples on which the classifier performs poorly. In the context of evasion attacks, ensemble mutual agreement serves as criteria for separating novel attacks and weak mimicry attacks from effective mimicry attacks. For novel attacks, it is common for the voting result to be distributed around 50%, indicating that the observations under consideration map consistently close to neither the benign or malicious samples in the training set. Since these attacks fall in the relatively rare uncertain range, they are easily discerned and are considered weak evasions. Strong mimicry attacks are those where the distribution of the attack votes is close to that of the benign observations. Hence, typical novel attacks are identified by mutual agreement analysis, but strong mimicry attacks cannot be. Since uncertain observations are supported poorly by the training set, these observations are the most effective to add to the training set in order to improve classifier accuracy.

TABLE III. PDFRATE OUTCOMES FOR BENIGN DOCUMENTS FROM OPERATIONAL EVALUATION SET

	Benign		Malicious	
Classifier		Uncertain		
Contagio	98076	1408	203	40
University	99217	360	95	55

TABLE IV. PDFRATE OUTCOMES FOR MALICIOUS DOCUMENTS FROM OPERATIONAL EVALUATION SET

	Benign		Malicious	
Classifier		Uncertain		
Contagio	0	0	19	254
University	0	0	0	273

In operation, mutual agreement analysis is employed to prevent evasion of an intrusion detection system. The mutual agreement rate is trivially derived from the result provided by an ensemble classifier at the time that detection occurs. Ensemble classifier agreement can be used in many ways by the operator, including adjusting the vote threshold to prevent false positives or false negatives, filtering observations for quarantine or more expensive analysis, and prioritizing alerts. The strength of mutual agreement analysis is that it can be used to identify probable intrusion detection evasion at the time of evasion attempts.

V. EVALUATION

To evaluate our approach, we apply mutual agreement analysis to PDFrate using an operational data set taken from a real world sensor and mimicry attack data taken from the Mimicus and Reverse Mimicry attacks. We study the degree to which mutual agreement analysis separates observations on which the classifier is reliable from classifier evasions. We also evaluate the utility of mutual agreement analysis in detecting novel malware families using the Drebin Android malware detector.

A. PDFrate Operational Data Set

We applied mutual agreement analysis to PDFrate scores for documents taken from a network monitor processing files transfered through web and email. This data set includes 110,000 PDF documents, which we randomly partitioned into two data sets. The operational evaluation set contains 100,000 documents and operational training set contains 10,000 documents. Ground truth for the documents was determined by scanning with many antivirus engines months after collection. These data sets included 273 and 24 malicious documents respectively. Table III and Table IV show the scores for the operational evaluation data set using both the Contagio and the University classifiers of PDFrate. The distribution of the PDFrate Contagio classifier scores for the benign and malicious samples of this operational evaluation data set are shown in Figure 2 and Figure 3.

It is important to note that the scores for the benign and malicious examples are weighted heavily to the far end of their respective score range, with the distribution falling off quickly. In a typical system deployment, the number of observations in the uncertain range is very small and the majority of misclassifications fall within the uncertain region. Hence, mutual agreement analysis can be used to make an estimate of

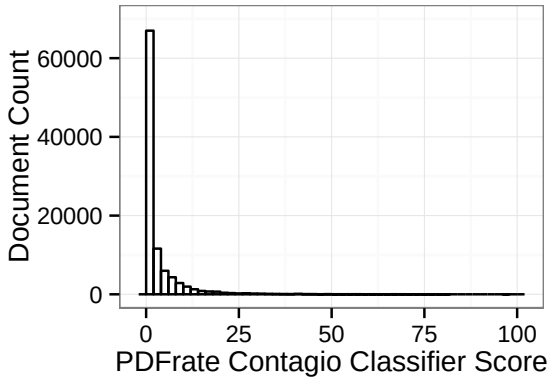

Fig. 2. Scores for Benign Documents From Operational Evaluation Set

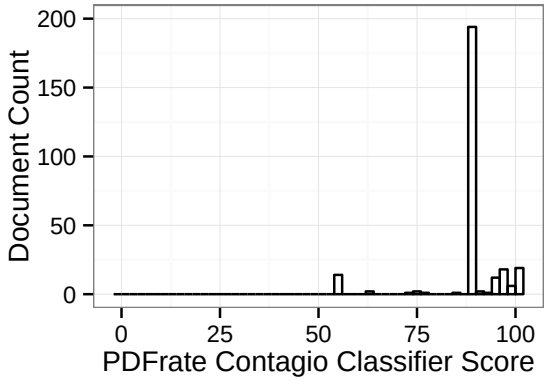

Fig. 3. Scores for Malicious Documents From Operational Evaluation Set

the upper bound on the number of misclassifications, at least in the absence of strong evasion attacks.

Not only is ensemble classifier mutual agreement analysis useful for identifying when the classifier is performing poorly, it is also effective for identifying specific examples which will provide the most needed support to improve the classifier. To demonstrate this, we sought to replicate improvements to the classification scores that would occur in the operational evaluation data set as additional samples are added to the classifier training set. We started with the Contagio classifier and added samples from the operational training set.

Using the original Contagio training data set, we determined the rating of all the observations in the operational training set. In an operational setting, all observations above the uncertain threshold (scores greater than 25) would typically require additional investigation, whether the outcome is uncertain or malicious. There were 200 documents in the operational training set matching this criteria. Of these 200 samples, 43 would be false positives and 14 would be false negatives using a traditional threshold. We added these 200 observations to the Contagio training set with the correct ground truth and created another classifier.

For comparison, we also created additional classifiers with varying sized randomly selected subsets of the operational training set to simulate randomly selected additions to the Con-

TABLE V. SCORES OF BENIGN DOCUMENTS FROM OPERATIONAL EVALUATION SET USING CONTAGIO CLASSIFIER SUPPLEMENTED WITH OPERATIONAL TRAINING DATA

		Benign		Malicious	
Additional Training Data	Training Set Size		Uncertain		
None (original Contagio)	10000	98076	1408	203	40
Random subset 2500	12500	99332	265	98	32
Random subset 5000	15000	99444	200	71	12
Random subset 7500	17500	99502	169	49	7
Uncertain and Malicious	10200	99506	183	26	12
Full training partition	20000	99540	134	48	5

TABLE VI. SCORES OF MALICIOUS DOCUMENTS FROM OPERATIONAL EVALUATION SET USING CONTAGIO CLASSIFIER SUPPLEMENTED WITH OPERATIONAL TRAINING DATA

		Benign		Malicious	
Additional Training Data	Training Set Size		Uncertain		
None (original Contagio)	10000	0	0	19	254
Random subset 2500	12500	0	14	4	255
Random subset 5000	15000	0	14	4	255
Random subset 7500	17500	0	14	4	255
Uncertain and Malicious	10200	0	14	7	252
Full training partition	20000	0	14	4	255

tagio classifier. The performance of these classifiers applied to the operational evaluation set is demonstrated in Table V and Table VI.

These results indicate that local tuning of the classifier has a great effect on improving the accuracy of the classifier. Note that shifting a few samples across the score midpoint in the wrong direction, as occurs with the malicious observations, is not considered harmful as these samples are already deep in the uncertain range (very close to the 50% vote mark) as shown in Figure 3. The ratio of observations in the benign region (certain true negatives) rises from 98.3% to 99.8% for either of the top two re-training strategies, even surpassing the accuracy of the generally superior University classifier (99.5%). The corresponding drop in false positives is important because it coincides with a drop in uncertain observations. In this case, if an operator responds to all uncertain or malicious observations, the majority of alerts will be true positives.

The random subset training additions have the outcome anticipated by intuition. As the number of random samples added from the training set increases, the classification results on the partitioned evaluation data improve. Adding the samples above the uncertain threshold from the training partition results in a classifier that is very close in accuracy to that constructed with the complete training partition. It follows that mutual agreement analysis is effective at identifying the observations on which the classifier performs poorly. It also follows that adding these samples to the training set does indeed improve the classifier by providing support in the region near these samples. On the other hand, adding the observations for which there is high mutual agreement improves the classifier very little. The result of adding the whole training set and adding the uncertain samples is similar, but the effort invested is drastically different. The difference in obtaining ground truth and adding 10,000 vs. 200 observations to the training set is monumental.

B. Mimicus

To demonstrate the utility of mutual agreement analysis in identifying observations that evade detection, we reproduced the work of Šrndić and Laskov [43] and applied mutual agreement analysis to these evasion attempts. We used the Mimicus framework to generate PDF documents that implement various evasion attack scenarios. We used the same data sets as the Šrndić and Laskov publication and submitted the resulting documents to pdfrate.com to obtain scores. Because we used the same attack data, our results are limited to 100 samples per attack type. We were able to achieve results that closely mirrored those documented in the Mimicus study.

TABLE VII. PDFRATE CONTAGIO CLASSIFIER OUTCOMES FOR MIMICUS EVASION ATTACKS

	Benign		Malicious	
Scenario		Uncertain		
Baseline Attack	0	0	0	100
F_mimicry	2	70	26	2
FC_mimicry	7	78	15	0
FT_mimicry	10	64	26	0
FTC_mimicry	33	62	5	0
F_gdkde	7	92	1	0
FT_gdkde	4	95	0	1

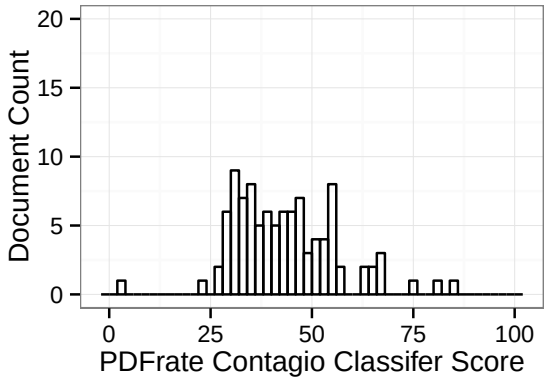

Fig. 4. Score Distribution for F_Mimicry Attack

We present the results of classification using mutual agreement from the various attack scenarios in Table VII. Note that since all of these documents are malicious, the correct classification is malicious. A rating of benign indicates successful evasion.

The distribution of PDFrate voting scores for the documents in each non-GD-KDE scenario is demonstrated in Figures 4 through 7. The GD-KDE attacks will be addressed specifically in Section VII. The vote score distribution of these attacks is largely disjoint of that seen in typical benign or malicious observations. Using an ensemble classifier diversity based approach, the majority of these attacks can be separated from benign observations. Hence, these attacks should be considered weak mimicry attempts.

When all attributes of the classifier are known, 33% of the attacks are effective. However, when either the details of the classifier or the training set are withheld, the attack success rate drops to 10% or lower. In addition to evaluation against the Contagio data set, the mimicry attack data was tested against the classifier trained with the University data set. This results in an alternate FC attack scenario because the training set is unknown to the attacker. Figure 8 shows the distribution of scores from applying the malware from the FTC attack sce-

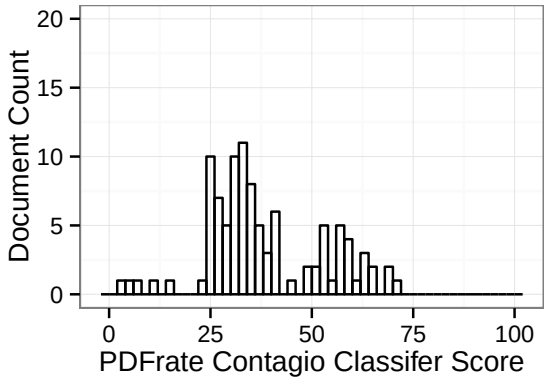

Fig. 5. Score Distribution for FT_Mimicry Attack

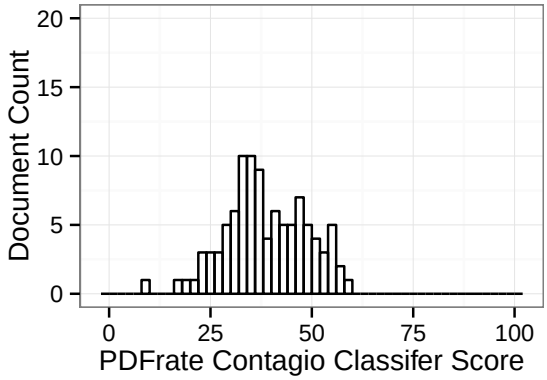

Fig. 6. Score Distribution for FC_Mimicry Attack

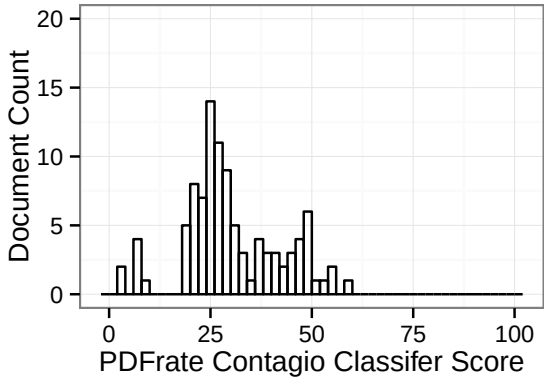

Fig. 7. Score Distribution for FTC_Mimicry Attack

nario against the Contagio classifier to the University classifier. The results are very similar between the two classifiers. In both cases, only 7 of the 100 evasion attempts are classified as benign. Carefully comparing Figure 6 and Figure 8 yields the observation that the University classifier provides a tighter cluster of scores near the center of the disagreement region. The results from the Contagio classifier are similar to that of the University classifier because the Mimicus evasion attempts use Contagio data for both baseline benign and attack data.

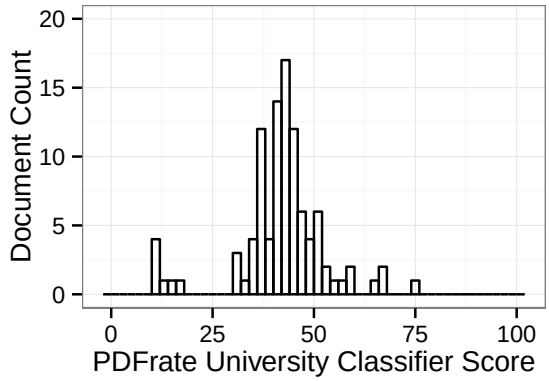

Fig. 8. Scores for FC_Mimicry Attack Using University Classifier

TABLE VIII. PDFRATE OUTCOMES FOR REVERSE MIMICRY ATTACKS

Contagio Classifier

	Benign		Malicious	
Scenario		Uncertain		
EXEembed	77	22	1	0
PDFembed	93	7	0	0
JSinject	30	67	3	0

University Classifier

	Benign		Malicious	
Scenario		Uncertain		
EXEembed	0	4	16	80
PDFembed	81	19	0	0
JSinject	0	22	55	23

When mutual agreement is utilized, the majority of mimicus attacks are labeled as uncertain, indicating known classifier failure and possible evasion. In the best mimicry attack scenario, where all attributes of PDFrate are known, only 33% of the mimicry attempts are successfully classified as benign. If some details of the classifier, such as the exact training set, are not known by the attacker, then the mimicry success rate is below 10%.

C. Reverse Mimicry

We also applied mutual agreement analysis to the Reverse Mimicry attack proposed by Maiorca et al. [26], [27]. The exact procedures required to replicate these attacks are not publicly documented. However, Maiorca et al. provided us with the documents used in their studies. Their most recent attacks involved 500 documents in each evasion scenario. To remain consistent with the Mimicus attack evaluation, we took a 100 sample random subset of each scenario for our evaluation.

In Table VIII, we present the results of applying mutual agreement analysis to the Reverse Mimicry attacks against both the Contagio and University classifiers. The score distributions for these attacks against the University classifier are shown in Figures 9, 10, and 11. In spite of mutual agreement analysis, 67% of the Reverse Mimicry attacks are successful evasions (considered benign) against the Contagio classifier.

The University classifier fares much better than the Contagio classifier. The only evasions against the University classifier are achieved by the PDFembed attack. This attack is so successful because a complete malicious PDF is embedded

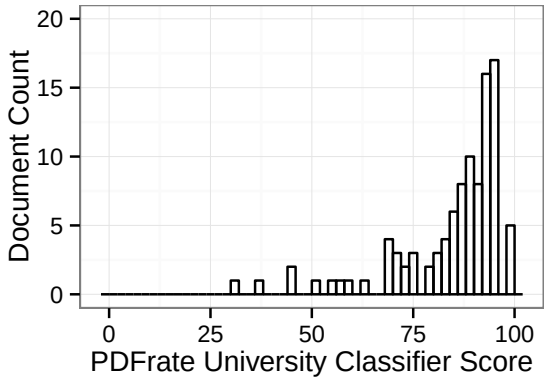

Fig. 9. Score Distribution for EXEembed Attack

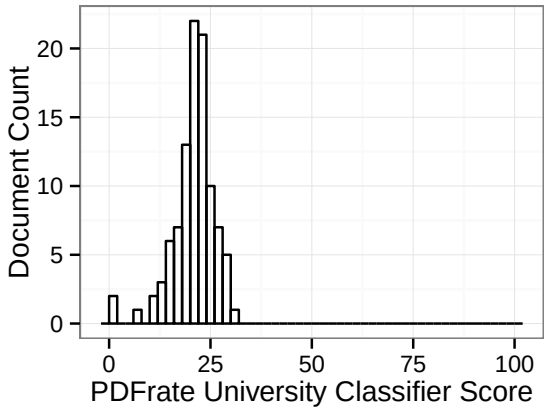

Fig. 10. Score Distribution for PDFembed Attack

in an otherwise benign document. This embedded document resides in a compressed data stream, which means that the structural features cannot be observed by PDFrate's feature extractor. This is in contrast to the other scenarios, EXEembed and JSinject, where despite efforts at minimization, some indicators of malfeasance remain exposed.

The PDFembed scenario is effective against the detector at pdfrate.com because it does not perform recursive decoding and analysis as would be necessary in an operational system. This failure is similar to malware analysis systems that assume an input of an unpacked executable and fail when presented with a packed executable or a Trojan document. When PDFrate is deployed in operational detection systems, it is usually done within a framework that provides both decoding of PDF streams and extraction of PDFs from other containers such as emails or zip files [3]. In all the PDFembed attacks, the embedded document was identical. The Contagio and University classifiers both easily detect this document with high confidence once it is extracted, returning scores of 97.6% and 100% respectively.

For the isolated PDFrate implementation, the PDFembed scenario represents a strong evasion scenario, where classifier introspection provides little benefit because the feature extractor is evaded so well. Even though the Contagio based

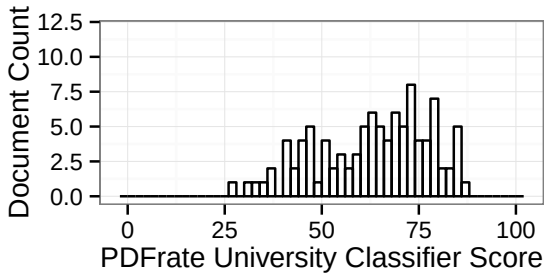

Fig. 11. Score Distribution for JSinject Attack

classifier is a poor fit for the malware used in the EXEembed and JSinject, many of these samples still fall in the uncertain outcome vote range. When the stronger University classifier is used, mutual agreement analysis flags these evasion scenarios that would otherwise be successful.

D. Drebin

To apply mutual agreement analysis to the Drebin Android malware detector, we constructed an ensemble classifier. We employed a Random Forest classifier, which required adapting the features to ensure computational efficiency and to ensure results comparable to the original linear SVM. Instead of using all string values as features, we used a subset of 891 features that comprise the most durable features. We used all of the features for constrained categories such as API calls and permissions. For arbitrarily named attributes, such as components and intents, we utilized the most prolific values, selecting those which occur over 100 times in the training set. Lastly, we ignored specific values for highly volatile items such as URLs and network addresses, which compose over half the features used by Drebin. Lastly, we summed the occurrences of each category of features and used these counts as features. As an optimization, we de-duplicated any equivalent feature vectors during classifier training (not during evaluation). This de-duplication, using our narrow feature set, resulted in a reduction from 123,453 to 63,379 unique benign and 5,560 to 2,185 unique malicious samples. Barring these transformations of data, we used the published Drebin data sets including data set partitions in our evaluation.

We tuned our Random Forest based classifier to provide classification performance comparable to the linear SVM classifier of Drebin. The primary item we tuned was the ratio of benign to malicious samples used in training each tree. This was necessary because there is an extreme imbalance in the benign to malicious ratio of the various training sets. We tuned the ratio for individual tree training to 2.5 benign to 1 malicious in order to match the desired false positive rate of 1% chosen by Arp et al. We set the other tunable parameters for Random Forest to standard values: each Random Forest contained 1000 trees and the number of variables tried at each split was set to the square root of the number of features. Our Random Forest classifier provided an average false positive rate of 1.06% and a malware detection rate of 92.3% on the published data set partitions using a traditional threshold without an uncertain region. The Random Forest based classifier performance is very similar, albeit slightly inferior to that provided by Drebin's linear SVM. Figure 12 shows

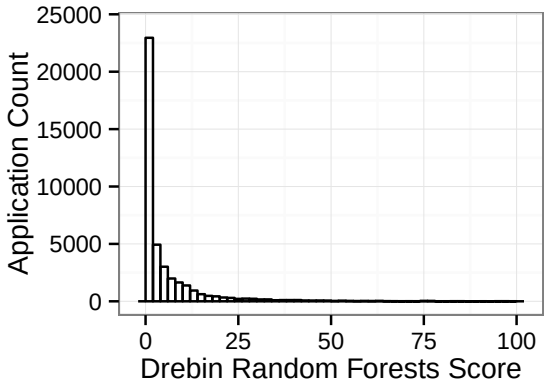

Fig. 12. Score Distribution for Benign Samples

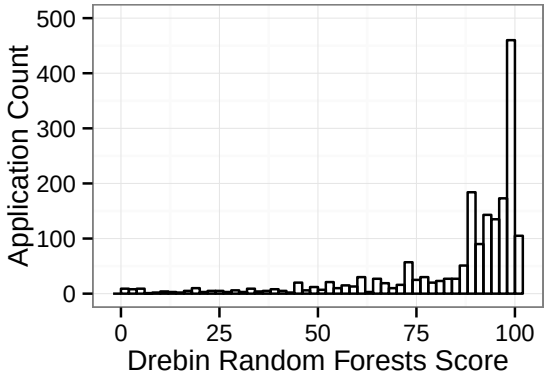

Fig. 13. Score Distribution for Malicious Samples

TABLE IX. DREBIN RANDOM FOREST CLASSIFIER OUTCOMES AS MUTUAL AGREEMENT THRESHOLD IS ADJUSTED

Benign Samples

	Benign (%)		Malicious (%)	
Mutual Agreement Threshold (%)		Uncertain		
30	97.46	1.49	0.54	0.52
40	96.49	2.45	0.63	0.43
50	95.12	3.82	0.71	0.35

Malicious Samples

30	4.44	3.27	5.44	86.85
40	3.77	3.93	7.30	84.99
50	3.16	4.56	10.34	81.95

the distribution of scores for the benign samples using one of the published data set partitions. Figure 13 shows the same for the malicious samples. As expected, the score distributions are shaped similar to that of PDFrate, but since the classifier accuracy is lower, the samples are distributed farther from the respective ends of the score continuum. Table IX shows the classifier outcomes for typical mutual agreement thresholds.

An important facet of the original Drebin study is the division of the malware by family and evaluation of the classifier on previously unknown malware families. This was achieved by withholding the family to be evaluated from the training set and then applying the resultant classifier to the malware samples in that family. It is noted by Arp et al. [4] that Drebin provides relatively poor classification of previously

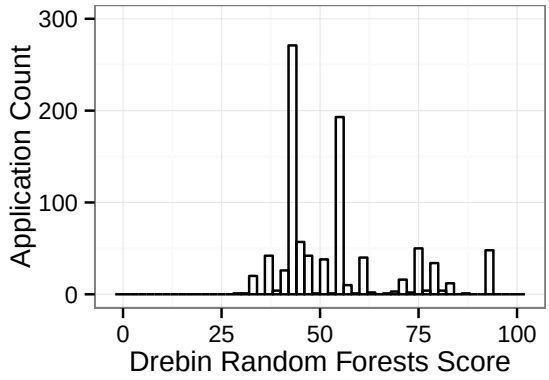

Fig. 14. Score Distribution for Unknown Family A

unknown malware. We applied our Random Forest based classifier and uncertain score region to this same problem. Figure 15 compares the detection rates of the linear SVM classifier and our Random Forest based classifier using mutual agreement analysis.

As expected, the vast majority of unknown malware families have the score distribution of a weak evasion attack, indicating that the classifier considers these observations neither similar to the benign or malicious samples seen in the training set. As an example, the scores of malware family A are shown in Figure 14. On average, 75.2% of every family is labeled as uncertain and an additional 8.2% are labeled as malicious using our Random Forest based classifier, while 50.6% of every family is labeled as malicious by the Drebin linear SVM. Families Q and R represent strong evasion. Arp et al. note that Family R cannot be reliably detected with the feature set used by Drebin. While the features used by Drebin are sufficient for the detection of Family Q when included in the training set, it is too different from other families in Drebin's feature space to be flagged as an evasion. On the other hand, Family P is so similar to other malware families in Drebin's feature space, that it is not necessary to have samples of this family in the training set. Removing these 3 families, an average 89.7% of the samples in the remaining 17 families are identified as malicious or uncertain by the Random Forest classifier, while 53.2% are detected by the linear SVM classifier. It should be considered advantageous to label these previously unknown samples as uncertain so that the operator can take action to improve the classifier. While the linear SVM classifier provides the average classification accuracy of a coin toss in these scenarios, the mutual agreement conscious ensemble is able to flag the majority of the novel attacks as possible evasions.

Mutual agreement analysis is effective at identifying possible evasions in the PDFrate and Drebin malware detection systems caused by both novel attacks and targeted mimicries.

VI. MUTUAL AGREEMENT THRESHOLD TUNING

For most of our evaluations, we used a 50% mutual agreement threshold, which splits the classifier voting score region into four equal sized quadrants. It is possible to choose an arbitrary mutual agreement threshold. Table IX contains Drebin predictions for three mutual agreement threshold levels.

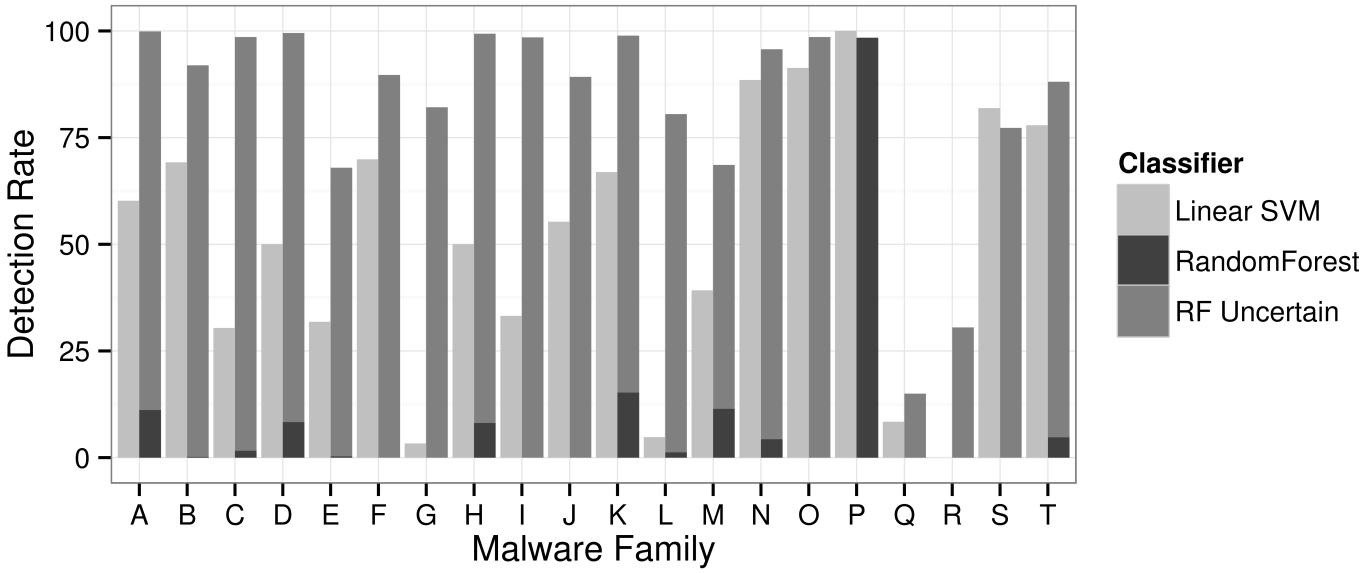

Fig. 15. Comparison of Detection Rate for Previously Unknown (excluded from training set) Malware Families

In Table X, we present the PDFrate University classifier outcomes applied to the operational evaluation data set and the FC Mimicus attacks across the full mutual agreement range.

The exact mutual agreement threshold chosen strikes a balance between improvement in classification failure detection and the number of classifier predictions thrown out as uncertain. Operators who wish to have a lower amount of uncertain outcomes may choose a lower threshold. Taking the PDFrate performance in Table X as an example, if 30% is selected as a threshold, the uncertain region comprises ensemble classifier voting scores between 35% and 65% instead of 25% and 75% with a 50% threshold. For the operational data set, the uncertain rate for benign samples drops from 0.456% to 0.256%. However, the number of successful evasion attempts rises from 7% to 12%. The optimal setting for this threshold depends on the preferences of the operator. The sensitivity of uncertain detection is adjusted by tuning the mutual agreement threshold, setting the boundaries for the uncertain range.

VII. GD-KDE and Ensemble SVMs

Mutual agreement analysis should apply to all ensemble classifiers that provide sufficient diversity in individual classifiers. To validate this, we studied the feasibility of countering evasion against SVMs by applying mutual agreement analysis to SVMs using an ensemble approach.

The Mimicus attack framework implements a Gradient Descent and Kernel Density Estimation (GD-KDE) attack against their PDFrate replica utilizing an SVM classifier. This attack operates by exploiting the known decision boundary of a differentiable classifier [8].

We reproduced the GD-KDE evasion attacks of Mimicus and confirm that they are indeed extremely effective. Using the e1071 package of R [2], which relies on libSVM [10], we calculated the average probability of 8.9% malicious (or 91.1% benign) for both GD-KDE scenarios, putting these attacks squarely within the evasion region. Šrndić and Laskov use the

TABLE X. PDFrate University Classifier Performance as Mutual Agreement Threshold is Adjusted

Benign Operational Evaluation

Mutual Agreement Threshold	Uncertain Score Range	True Negative Rate (TNR)	False Positive Rate (FPR)	Uncertain Rate (UR)
0%	-	99.8%	0.150%	0%
10%	(45,55)	99.8%	0.128%	0.0592%
20%	(40,60)	99.7%	0.103%	0.147%
30%	(35,65)	99.7%	0.0832%	0.256%
40%	(30,70)	99.6%	0.0712%	0.342%
50%	(25,75)	99.5%	0.0552%	0.456%
60%	(20,80)	99.3%	0.0331%	0.618%
70%	(15,85)	99.1%	0.0291%	0.825%
80%	(10,90)	98.7%	0.0261%	1.27%
90%	(5,95)	97.0%	0.0120%	3.01%
100%	(0,100)	53.6%	0%	46.4%

Malicious Operational Evaluation

Threshold	Range	FNR	TPR	UR
0%	-	0%	100%	0%
10%	(45,55)	0%	100%	0%
20%	(40,60)	0%	100%	0%
30%	(35,65)	0%	100%	0%
40%	(30,70)	0%	100%	0%
50%	(25,75)	0%	100%	0%
60%	(20,80)	0%	99.6%	0.366%
70%	(15,85)	0%	99.6%	0.366%
80%	(10,90)	0%	99.6%	0.366%
90%	(5,95)	0%	99.6%	0.366%
100%	(0,100)	0%	95.6%	4.40%

Mimicus FC Attack

Threshold	Range	FNR	TPR	UR
0%	-	84%	16%	0%
10%	(45,55)	69%	8%	23%
20%	(40,60)	31%	4%	65%
30%	(35,65)	12%	4%	84%
40%	(30,70)	7%	1%	92%
50%	(25,75)	7%	1%	92%
60%	(20,80)	7%	0%	93%
70%	(15,85)	6%	0%	94%
80%	(10,90)	0%	0%	100%
90%	(5,95)	0%	0%	100%
100%	(0,100)	0%	0%	100%

TABLE XI. NUMBER OF DOCUMENTS PER GD-KDE ATTACK WHERE ENSEMBLE SVM CLASSIFIER PROVIDES CORRECT PREDICTION AS PORTION OF FEATURES USED IS VARIED.

	Feature Subset			
Attack	5%	7.5%	10%	12.5%
Baseline Malicious	100	99	98	98
Baseline Benign	2	41	93	94
F_gdkde	100	100	99	5
FT_gdkde	99	100	92	1

TABLE XII. NUMBER OF DOCUMENTS PER GD-KDE ATTACK WHERE ENSEMBLE SVM CLASSIFIER PROVIDES CORRECT PREDICTION AS PORTION OF TRAINING DATA USED IS VARIED.

	Training Data Subset			
Attack	12.5%	25%	50%	100%
Baseline Malicious	86	87	92	98
Baseline Benign	100	100	100	100
F_gdkde	0	0	0	0
FT_gdkde	0	0	0	0

scaled distance from the SVM decision boundary of a different SVM implementation to provide a similar result. The GD-KDE attacks demonstrate that introspection of a single classifier such as SVM cannot be relied upon to detect evasions.

While effective against an SVM classifier, the results on PDFrate's Random Forest classifier using the GD-KDE attack are roughly comparable to the conventional counterparts (see Table VII). It is is not practical to wage a similar type of attack against Random Forests because Random Forests have extremely complex and stochastic decision boundaries.

We sought to determine the extent to which we could make an SVM classifier identify probable evasions through diversity enabled introspection. We implemented a simple SVM based ensemble classifier using 100 independent SVM classifiers with the score being the simple sum of the votes of individual classifiers. To determine the attributes important to building diversity in ensembles, we varied the subset of features and training data used in constructing each of the individual SVMs. We performed a full grid search. The most salient results are reported in Table XI, which shows feature bagging using the full training data set, and Table XII, which shows bagging on training data using the full feature set. These tables demonstrate the portion of classifier outcomes that match the correct result (malicious or uncertain).

It appears that bagging of training data is not particularly important in building an ensemble classifier where mutual agreement analysis is useful. To our amazement, we found no situation where anything but the full training set provided the best results.

However, bagging of features is critical to constructing a classifier where mutual agreement analysis is able to identify uncertain predictions. This bagging of features for an SVM classifier provides the necessary diversity in extrapolation that makes mutual agreement analysis meaningful. It seems that the individual classifiers based on subsets of the complete feature set are much harder to evade collectively than a single classifier using all the features. While a single classifier can be evaded by successfully mimicking a subset of the features, it appears that a combination of multiple classifiers based on a small number of features requires a more complete mimicry across the full feature set. The application of feature bagging to the many independent SVMs makes a GD-KDE style attack infeasible as there is no longer a single predictable decision boundary to attack.

TABLE XIII. PDFRATE SVM ENSEMBLE CLASSIFIER OUTCOME FOR GD-KDE EVASION ATTACKS.

	Benign		Malicious	
Attack		Uncertain		
Baseline Malicious	0	0	2	98
Baseline Benign	93	7	0	0
F_gdkde	3	97	0	0
FT_gdkde	8	91	1	0

The results also indicate that careful tuning of the portion of features used in bagging is critical when using an SVM based ensemble. There seems to be a trade-off between the ability to correctly classify malicious observations (including evasion attempts) by using fewer features in each classifier, and benign observations by using more features. The use of fewer features results in a more complex classifier with smaller divisions while more features moves closer to a standard SVM, which has a single hyperplane divider. These result suggest that the features used in PDFrate provide better extrapolation for benign samples. It appears that the malicious samples have higher variation in PDFrate's features, requiring more similar training samples for successful classification.

Table XIII shows the outcomes of the SVM ensemble classifier applied to the Mimicus GD-KDE attacks and baseline benign and malicious samples. The outcome shows that while the evasion attempts are successful in dropping the scores out of the malicious range, the vast majority of the evasion attempts fall in the uncertain range. Only 8% of the evasion attempts are fully successful in the best scenario while only 4.5% of the known data is in the uncertain region. These results are comparable to results obtained using PDFrate's Random Forest classifier where GD-KDE attacks are not possible. Hence, mutual agreement analysis applies not only to Random Forests, but seems to apply generally to all ensembles which have adequate diversity. Bagging of features appears central to this capability.

VIII. DISCUSSION AND FUTURE WORK

Mutual agreement analysis in ensemble classifiers provides an estimate of confidence that the classifier prediction is accurate, without external validation. Many classifiers can provide a score continuum, such as the distance from the decision boundary used in SVM, but these metrics are not accurate in the face of mimicry attacks. Furthermore, conventional measures of confidence are not applicable to data which diverges from the population for which ground truth is known.

Mutual agreement reflects the internal consistency of the classifier. This internal consistency is a proxy for the confidence of the classifier, assuming adequate strength of the features. The attacks against PDFrate demonstrate that mimicry resistant features are critical to identification of novel attacks. The sole strong evasion attack against PDFrate was successful because it fully evaded PDFrate's features through embedding a malicious PDF in another, making the malicious PDF invisible to the feature extractor. Other attacks, while seeking to fool the feature extractor, were insufficient because some features were still operative. If the feature extractor is

resistant to tampering and the features are proper indicators of malfeasance, then novel attacks will either be detected as malicious or be rated as uncertain. However, if the feature set (or feature extraction mechanism) is weak, then evasion will still be possible. Operators must be vigilant to prevent evasion during the feature extraction phase of malware detection.

In building an ensemble using base SVM classifiers, we found feature bagging to be critical to generating the diversity necessary to make mutual agreement measurements meaningful. Unqualified, bagging refers to the utilization of random subsets of training data. This method is used extensively in machine learning techniques. In our study, bagging of training data was not shown to be important for mutual agreement analysis. This may have been due to a lack of diversity in that training set. Further studies might show under what conditions training data bagging provides diversity useful for facilitating mutual agreement analysis. We also observed that tuning the portion of features used in our SVM ensemble was important. We observed no similar need to tune the parameters of Random Forest based classifiers, but this should be an area of future study. The number of features tried at each node (mtry) and the depth of the trees might impact the useful diversity in a Random Forest. It appears that many features, even if they are interdependent or have low classification value, contribute to make evasion more difficult.

If the features are strong, then the relevance of the training set will dictate the mutual agreement rating for individual observations. If the test observations are similar to samples in the training set, then high certainty predictions will occur. The test observations that differ from the training set in feature space will be given classifications considered uncertain by the classifier. In some instances in our evaluation, the quality of the training set was shown to be important to detection of evasion attacks. For example, the superior PDFrate University classifier had considerably fewer evasions than the Contagio classifier for the Reverse Mimicry attacks. For the Drebin evaluation, Family R represented strong evasion due to weak features, but Family Q was detectable if it was included in the training set. Therefore, the effectiveness of mutual agreement analysis is also dependent upon adequate coverage in the training set. However, the effectiveness of the training set is directly dependent upon the strength of the features. A weak feature set will require a more expansive training set than a feature set that more closely models fundamental malicious attributes. Operators should ensure that features used for malware detection are not only resistant to spoofing but that they are based on artifacts caused by malware and not merely coincidental with current attacks.

As was shown in Section V-B, keeping the training set of a classifier secret helps improves resiliency against targeted evasion attempts. It might be advisable for operational systems to hide the exact scores returned from their classifiers as these scores assist attackers in knowing if changes they make hurt or help their evasion attempts. This information could weaken the benefit provided by a secret training set [18].

The GD-KDE attacks of Mimicus demonstrate that some classifiers can make machine learning based detectors susceptible to evasion attacks. Stochastically generated ensemble classifiers have not been shown to be vulnerable to similar attacks, but new approaches might be found. The ability to measure mutual agreement in ensemble classifiers comes at little cost, but provides for detection of practical classifier evasion. This capability is a strong reason to use ensembles in situations where classifier evasion is a concern, such as in malware detectors. If mutual agreement is used to optimize classifier training, then an attacker may have more knowledge of additions to the training set than if random selection is used. However, it is not obvious how this knowledge could be exploited by an attacker. Any effective attacks that use knowledge of mutual agreement based training optimization to poison the classifier would be important.

Some advocate the use of simple, monolithic classifiers, because the result is perceived as easier to interpret. For example, the ability of Drebin to identify the features that contribute to the classification is lauded. It is not clear, however, if this information is really useful to end users. Users are already given the opportunity to review permissions and often choose incorrectly when prompted. Given that URLs and API calls can be socially engineered and that users are generally not aware of these elements, it is not likely that providing these items as context to a user will help them make a correct decision. For security professionals, ensemble classifiers provide mechanisms that aid in analysis such as similarity to existing known malicious or benign samples. Most importantly, the feature set will be useful to a trained analyst.

Mutual agreement analysis gives operators greater confidence in the accuracy of the classifier and the ability to prioritize response to alerts. Some operators will use ensemble classifier introspection simply to adjust the voting threshold. Environments that seek to avoid false negatives (evasion attacks) will use a low threshold and increase the number of false positives. On the other hand, some environments might use a higher than normal voting threshold to achieve a low false positive but potentially higher false negative rate, such as that achieved by antivirus engines.

The operator gets the most benefit from mutual agreement analysis when uncertain observations are subjected to focused analysis. These samples must necessarily be subjected to different and complementary analysis or detections. Since the number of uncertain observations is low for a well performing classifier, this second opinion can be relatively expensive, possibly manually driven or involving dynamic analysis. Ensemble diversity based confidence estimates are useful for organizations that desire to identify novel attacks to perform additional analysis. While possibly unconventional for the machine learning field, the addition of the uncertain outcome is intuitive for the security field where many systems provide only adjudications for known observations, whether benign or malicious. For example, it is common for SPAM filters to utilize a quarantine for samples that cannot be classified reliably. Very often, high fidelity alerts are preferred over a response for every observation.

Mutual agreement analysis is very effective at identifying those samples which are not similar to the already known samples in the training set. Since adding uncertain samples to the classifier dramatically improves the classifier accuracy, analysis of uncertain observations is likely to motivate rather than desensitize operators. Operators are empowered to improve the classifier in a manner much more effective than random additions to the training set.

Evaluation of machine learning based detectors might be improved though application of mutual agreement analysis. A concise metric is the Uncertain Rate, or portion of observations for which a classifier is poorly suited to provide a prediction. The effectiveness of classifier evaluation using the mutual agreement score distribution and variance could be a topic of future studies. The classifier score distributions shown in Figure 12 and Figure 13 seem to indicate that regression could be used to predict the amount of successful evasions. The difficulty in this type of analysis, however, is separating the arcs for the benign and malicious data when external ground truth is not provided.

Most importantly, monitoring mutual agreement in ensemble classifiers raises the bar for evasion, for both previously unseen attacks and targeted mimicry attacks. Contemporary evasion attacks, which have called into question the resiliency of learning based detectors, are shown to be weaker than previously supposed. Merely obfuscating attacks such that they no longer appear as known attacks is not enough. Successful mimicries must very closely mirror benign samples. Of course, future research into the degree to which mutual agreement analysis can improve attack quality seems worthwhile.

IX. Conclusions

We introduce a new technique to detect malware classifier performance degradation on individual observations. Mutual agreement analysis relies on diversity in ensemble classifiers to produce an estimate of classifier confidence. We evaluate our approach using over 100,000 PDF documents and 100,000 Android applications. Applying PDFrate to documents taken from a real network, we find that the number of uncertain outcomes is small–only 0.2%. If these uncertain examples are excluded, the true positive rate rises from 95% to 100% and the false positive rate drops from 0.053% to 0.0050%. Furthermore, mutual agreement analysis is effective at identifying the samples to be added to the training set, resulting in considerably more rapid improvements in classifier performance than random sampling. In the direct evasion attacks against PDFrate and novel attacks against Drebin, the majority of the observations are assigned the outcome of uncertain, notifying the operator of the detector failure. While evasion attacks are still possible, they require more complete mimicry across the whole feature set.

We believe that mutual agreement analysis can be applied generally to ensemble classifiers. We find that feature bagging is critical to diversity based evasion detection. The GD-KDE attack, employed with great success against Support Vector Machines, can be foiled by an SVM ensemble. Ensemble classifier mutual agreement analysis provides a critical mechanism to evaluate the accuracy of machine learning based detectors without using external validation.

Acknowledgments

The authors would like to thank all of the reviewers for their valuable comments and suggestions. This work is supported by the National Science Foundation Grant No. CNS 1421747 and II-NEW 1205453. Opinions, findings, conclusions, and recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or US Government.

References

[1] "Mimicus," http://github.com/srndic/mimicus.

[2] "R Project," http://www.r-project.org/.

[3] M. Arnao, C. Smutz, A. Zollman, A. Richardson, and E. Hutchins, "Laika BOSS: Scalable File-Centric Malware Analysis and Intrusion Detection System," https://github.com/lmco/laikaboss, Jul. 2015.

[4] D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, and K. Rieck, "Drebin: Effective and explainable detection of android malware in your pocket," in *21th Annual Network and Distributed System Security Symposium (NDSS)*, Feb. 2014.

[5] D. Barbara, C. Domeniconi, and J. P. Rogers, "Detecting outliers using transduction and statistical testing," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 55–64.

[6] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar, "Can Machine Learning Be Secure?" in *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS '06. New York, NY, USA: ACM, 2006, pp. 16–25.

[7] U. Bayer, P. M. Comparetti, C. Hlauschek, C. Kruegel, and E. Kirda, "Scalable, Behavior-Based Malware Clustering," in *Network and Distributed System Security Symposium (NDSS) 2009*, 2009.

[8] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Srndic, P. Laskov, G. Giacinto, and F. Roli, "Evasion Attacks against Machine Learning at Test Time," in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science, H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, Eds. Springer Berlin Heidelberg, 2013, no. 8190, pp. 387–402.

[9] B. Biggio and P. Laskov, "Poisoning attacks against Support Vector Machines," in *In International Conference on Machine Learning (ICML) 2012*, 2012.

[10] C.-C. Chang and C.-J. Lin, "LIBSVM: A Library for Support Vector Machines," http://www.csie.ntu.edu.tw/˜cjlin/libsvm/, May 2011.

[11] D. Chinavle, P. Kolari, T. Oates, and T. Finin, "Ensembles in Adversarial Classification for Spam," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, ser. CIKM '09. New York, NY, USA: ACM, 2009, pp. 2015–2018.

[12] M. Cova, C. Kruegel, and G. Vigna, "Detection and Analysis of Drive-by-download Attacks and Malicious JavaScript Code," in *Proceedings of the 19th International Conference on World Wide Web*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 281–290.

[13] G. Cretu, A. Stavrou, M. Locasto, S. Stolfo, and A. Keromytis, "Casting out Demons: Sanitizing Training Data for Anomaly Sensors," in *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, 2008, pp. 81–95.

[14] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna, "COMPA: Detecting Compromised Accounts on Social Networks." in *NDSS, 2013*, 2013.

[15] P. Fogla and W. Lee, "Evading network anomaly detection systems: formal reasoning and practical techniques," in *Proceedings of the 13th ACM conference on Computer and communications security*. Alexandria, Virginia, USA: ACM, 2006, pp. 59–68.

[16] M. Handley, V. Paxson, and C. Kreibich, "Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics." in *2001 USENIX Security Symposium*, 2001, pp. 115–131.

[17] M. Heiderich, M. Niemietz, F. Schuster, T. Holz, and J. Schwenk, "Scriptless Attacks: Stealing the Pie Without Touching the Sill," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, ser. CCS '12. New York, NY, USA: ACM, 2012, pp. 760–771.

[18] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar, "Adversarial machine learning," in *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, ser. AISec '11. New York, NY, USA: ACM, 2011, pp. 43–58.

[19] S. Jana and V. Shmatikov, "Abusing File Processing in Malware Detectors for Fun and Profit," in *2012 IEEE Symposium on Security and Privacy (SP)*, May 2012, pp. 80–94.

[20] J. Jang, D. Brumley, and S. Venkataraman, "BitShred: feature hashing malware for scalable triage and semantic analysis," in *Proceedings of the 18th ACM conference on Computer and communications security*, ser. CCS '11. New York, NY, USA: ACM, 2011, pp. 309–320.

[21] G. Kakavelakis, R. Beverly, and J. Young, "Auto-learning of SMTP TCP Transport-Layer Features for Spam and Abusive Message Detection," in *LISA 2011, 25th Large Installation System Administration Conference*. Boston, MA, USA: USENIX Association, Dec. 2011.

[22] L. I. Kuncheva and C. J. Whitaker, "Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy," in *Machine Learning*, vol. 51, May 2003, pp. 181–207.

[23] P. Laskov and R. Lippmann, "Machine learning in adversarial environments," in *Machine Learning*, vol. 81, Aug. 2010, pp. 115–119.

[24] P. Laskov and N. Srndic, "Static detection of malicious JavaScript-bearing PDF documents," in *Proceedings of the 27th Annual Computer Security Applications Conference*, ser. ACSAC '11. New York, NY, USA: ACM, 2011, pp. 373–382.

[25] W.-J. Li, S. Stolfo, A. Stavrou, E. Androulaki, and A. D. Keromytis, "A Study of Malcode-Bearing Documents," in *Detection of Intrusions and Malware, and Vulnerability Assessment 2007*, B. Hämmerli and R. Sommer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, vol. 4579, pp. 231–250.

[26] D. Maiorca, D. Ariu, I. Corona, and G. Giacinto, "A Structural and Content-Based Approach for a Precise and Robust Detection of Malicious PDF Files," in *Proceedings of the 1st International Conference on Information Systems Security and Privacy*. ScitePress Digital Library, 2015, pp. 27–36.

[27] D. Maiorca, I. Corona, and G. Giacinto, "Looking at the bag is not enough to find the bomb: an evasion of structural methods for malicious PDF files detection," in *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*, ser. ASIA CCS '13. New York, NY, USA: ACM, 2013, pp. 119–130.

[28] J. Mason, S. Small, F. Monrose, and G. MacManus, "English Shellcode," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, ser. CCS '09. New York, NY, USA: ACM, 2009, pp. 524–533.

[29] E. Menahem, A. Shabtai, L. Rokach, and Y. Elovici, "Improving malware detection by applying multi-inducer ensemble," in *Computational Statistics & Data Analysis*, vol. 53, Feb. 2009, pp. 1483–1494.

[30] B. Nelson, M. Barreno, F. J. Chi, A. D. Joseph, B. I. P. Rubinstein, U. Saini, C. Sutton, J. D. Tygar, and K. Xia, "Exploiting machine learning to subvert your spam filter," in *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats 2008*. Berkeley, CA, USA: USENIX Association, 2008, pp. 7:1–7:9.

[31] J. Neyman, "Outline of a Theory of Statistical Estimation Based on the Classical Theory of Probability," *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, vol. 236, no. 767, pp. 333–380, 1937.

[32] M. Parkour, "11,355+ Malicious documents - archive for signature testing and research," http://contagiodump.blogspot.com/2010/08/malicious-documents-archive-for.html, Apr. 2011.

[33] R. Perdisci, D. Dagon, W. Lee, P. Fogla, and M. Sharif, "Misleading worm signature generators using deliberate noise injection," in *Security and Privacy, 2006 IEEE Symposium on*, 2006, pp. 15 pp.–31.

[34] M. A. Rajab, L. Ballard, N. Lutz, P. Mavrommatis, and N. Provos, "CAMP: Content-Agnostic Malware Protection." in *NDSS 2013*, 2013.

[35] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, "A Bayesian Approach to Filtering Junk E-Mail," in *AAAI 98 Workshop on Text Categorization*, Jul. 1998.

[36] J. Schlumberger, C. Kruegel, and G. Vigna, "Jarhead analysis and detection of malicious Java applets," in *Proceedings of the 28th Annual Computer Security Applications Conference*, ser. ACSAC '12. New York, NY, USA: ACM, 2012, pp. 249–257.

[37] H. Shacham, "The Geometry of Innocent Flesh on the Bone: Return-into-libc Without Function Calls (on the x86)," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 552–561.

[38] G. Shafer and V. Vovk, "A Tutorial on Conformal Prediction," *Journal of Machine Learning Research*, vol. 9, pp. 371–421, Jun. 2008.

[39] M. Z. Shafiq, S. A. Khayam, and M. Farooq, "Embedded Malware Detection Using Markov n-Grams," in *Proceedings of the 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, ser. DIMVA '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 88–107.

[40] C. Smutz and A. Stavrou, "Malicious PDF Detection Using Metadata and Structural Features," in *Proceedings of the 28th Annual Computer Security Applications Conference*, ser. ACSAC '12. New York, NY, USA: ACM, 2012, pp. 239–248.

[41] 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*, May 2010, pp. 305 –316.

[42] Y. Song, M. E. Locasto, A. Stavrou, A. D. Keromytis, and S. J. Stolfo, "On the Infeasibility of Modeling Polymorphic Shellcode," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 541–551.

[43] N. Srndic and P. Laskov, "Practical Evasion of a Learning-Based Classifier: A Case Study," in *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, ser. SP '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 197–211.

[44] G. Stringhini, C. Kruegel, and G. Vigna, "Shady Paths: Leveraging Surfing Crowds to Detect Malicious Web Pages," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, ser. CCS '13. New York, NY, USA: ACM, 2013, pp. 133–144.

[45] G. Varghese, J. A. Fingerhut, and F. Bonomi, "Detecting evasion attacks at high speeds without reassembly," in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*. Pisa, Italy: ACM, 2006, pp. 327–338.

[46] B. Waske, S. van der Linden, J. Benediktsson, A. Rabe, and P. Hostert, "Sensitivity of Support Vector Machines to Random Feature Selection in Classification of Hyperspectral Data," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, Jul. 2010, pp. 2880–2889.

[47] Y. Ye, L. Chen, D. Wang, T. Li, Q. Jiang, and M. Zhao, "SBMDS: an interpretable string based malware detection system using SVM ensemble with bagging," in *Journal in Computer Virology*, vol. 5, Nov. 2008, pp. 283–293.