Inferring Malware Detector Metrics in the Absence of Ground Truth

–

by

JOHN E CHARLTON III, Department of Computer Science

DISSERTATION
Presented to the Graduate Faculty of
The University of Texas at San Antonio
In Partial Fulfillment
Of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

COMMITTEE MEMBERS:
Ravi Sandhu, Ph.D., Co-Chair
Shouhuai Xu, Ph.D., Co-Chair
Murtuza Jadliwala, Ph.D.
Xiaoyin Wang, Ph.D.
Greg B. White, Ph.D.

THE UNIVERSITY OF TEXAS AT SAN ANTONIO
May
College of Sciences
2021

DEDICATION

This work is dedicated to the triumvirate that helped me along my expedition and through my quest.

Michael Ayers: you helped to launch the ship of discovery through the gift of freedom and time; without your help, I never would have been able to start the journey.

Dr. Ross McConnel, you helped me chart the seas and set my sights on what lay over the distant horizon; without your guidance, I never would have known what I could make of myself.

My wife, Allison, you helped fill my sails with wind, and kept me persevering through every storm; without the fire you kept lit, I never would have made it to these distant ports and come safely to this sturdy dock.

ACKNOWLEDGEMENTS

We thank VirusTotal for providing us the dataset. This work was supported in part by NSF CREST Grant #1736209, SaTC Grant #1814825 (#2122631), DMS Grant #1620945 and ARO Grant #W911NF-17-1-0566.

This Masters Thesis/Recital Document or Doctoral Dissertation was produced in accordance with guidelines which permit the inclusion as part of the Masters Thesis/Recital Document or Doctoral Dissertation the text of an original paper, or papers, submitted for publication. The Masters Thesis/Recital Document or Doctoral Dissertation must still conform to all other requirements explained in the Guide for the Preparation of a Masters Thesis/Recital Document or Doctoral Dissertation at The University of Texas at San Antonio. It must include a comprehensive abstract, a full introduction and literature review, and a final overall conclusion. Additional material (procedural and design data as well as descriptions of equipment) must be provided in sufficient detail to allow a clear and precise judgment to be made of the importance and originality of the research reported.

It is acceptable for this Masters Thesis/Recital Document or Doctoral Dissertation to include as chapters authentic copies of papers already published, provided these meet type size, margin, and legibility requirements. In such cases, connecting texts, which provide logical bridges between different manuscripts, are mandatory. Where the student is not the sole author of a manuscript, the student is required to make an explicit statement in the introductory material to that manuscript describing the students contribution to the work and acknowledging the contribution of the other author(s). The signatures of the Supervising Committee which precede all other material in the Masters Thesis/Recital Document or Doctoral Dissertation attest to the accuracy of this statement.

2021

Abstract

Measuring malware detector metrics is an important problem, which is extremely intriguing. This is so because the problem would be straightforward to solve if malware ground-truth labels are available. However, malware ground-truth labels are extremely costly to obtain in the real world, especially so when a large number of malware examples are involved, which is certainly true in practice. Therefore, the real-world problem is: How can we measure malware detector metrics in the absence of ground-truth labels? This problem has recently started to receive attention from the research community, but our understanding is still at a superficial level because there are many unaddressed questions. The present dissertation makes a solid step towards ultimately tackling this important problem, by making three contributions.

The first contribution is to introduce and investigate the notion of *relative accuracy* of malware detectors in the absence of ground truth, where relative accuracy is measured in an ordinal scale. We propose an algorithm to estimate the relative accuracy of malware detectors. We characterize when the algorithm leads to accurate relative accuracy using synthetic data with known ground-truth. We then apply the algorithm to measure the relative accuracy of real-world malware detectors based on a real dataset consisting of 10.7 million files and 62 malware detectors.

The second contribution is to use the relative accuracy rankings to infer the accuracy of malware detectors, including true positive rate and true negative rate. To this end, we introduce the novel idea of Bellwether detector. We validate the algorithm via synthetic data with known ground truth. We then apply the algorithm to the real-world dataset.

The third contribution is to enable the detection of metrics in regards to individual attributes of files that are scanned by the detectors. We use this data to analyze the specific ability of various detectors to measure these file attributes, and analyze when individual detectors can provide detailed data in regards to certain traits, assigning accuracy, true positive rates, and true negative rates for each detector in regards to each attribute.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

The ranking and evaluation of metrics in regards to malware detectors is an open problem in the field of cybersecurity that has a direct and immediate impact to industry, and is a critical concern in related research. Two aspects are of primary concern; the question of what metrics are of importance and how those metrics, whether new or established, should be measured.

In this work, we work to first measure the metric of detection accuracy when no ground truth values for the underlying state exist for the malware detector, or for the class (i.e., malicious or benign) of the the files the detector analyzes. This measurement provides us a ranking for the detectors, but this value is relative. The relative accuracy of the detectors is a new measurement, proposed here, for which we provide motivation and analysis in Chapter 3. We then use this method to evaluate a dataset based on synthetic data that has known ground truth values so that our method can be evaluated, and then apply the methodology to a corpus of real world malware dataset containing results from detectors and files provided by VirusTotal.

In Chapter 4 we break down the Relative Accuracy algorithm and transform it while keeping it mathematically equivalent; contrast it with the well known process of Principal Component Analysis and the correlation matrix; improve upon it by introducing the concept of a Bellwether detector; and use these improved ratings to motivate a new algorithm that allows for recovery of a variety of useful metrics for the individual detectors, including absolute accuracy, true negative detection rates, and true positive detection rates using further synthetic and real world data. This evaluation leads to a deeper understanding of the participating detectors, and highlights the dangers of treating all detectors equally.

Chapter 5 then concludes our work by extending our method to include analyzing file types detected by the various detectors, and applying the algorithm to recover metrics in regards to how well these detectors are able to classify these files individually versus in a conglomerate.

CHAPTER 2: RELATED WORK

In many contexts of cybersecurity practice, obtaining ground truth is challenging and often infeasible due to factors that include cost, high noise, uncertain data, and/or imperfect estimation. For instance, when using machine learning to train cyber defense models, we need to know the ground truth labels of the training examples. For a small set of examples, we may use human experts to label them. However, even in this case, perfect evaluation may not be guaranteed because humans are also error-prone and can make mistakes. For a large set of examples, it is obviously unrealistic for human experts to label them. This explains why third-party datasets are not necessarily trustworthy (e.g., blacklisted websites [23, 25, 47, 48, 55, 56, 73]).

Several studies have been conducted to estimate the accuracy of malware detectors in the absence of the ground truth regarding which examples are malicious [11, 12, 24, 50]. These studies used varying assumptions to estimate the accuracy of the malware detectors. For instance, [24] uses the naïve Bayesian method and treats the unknown ground truth labels as hidden variables, while leveraging the Expectation-Maximization (EM) method [11, 50] to estimate the (absolute) accuracy of malware detectors. However, four assumptions are made in [24]: (i) homogeneity of false-positives in all detectors; (ii) independence between malware detectors in their prediction process; (iii) low false-positives; and (iv) high false-negatives of the malware detectors. These assumptions need be removed in order to reflect real world applications. Du et al. [12] use a *frequentist* approach to design statistical estimators to measure the (absolute) accuracy of malware detectors while making strictly weaker assumptions, with only two of the four that are made in [24]. Still, all these studies [11, 12, 24, 50] make some assumptions. In contrast, the present paper investigates recovering the accuracy metrics of malware detectors without making those assumptions, effectively leading to a new approach.

This dissertation falls into the study of cybersecurity metrics and measurements, which is certainly one of the most fundamental open problems in cybersecurity that have yet to be tackled [43]. Recently, a new approach to tackling this problem, dubbed Cybersecurity Dynamics, has been pro-

posed [61, 62, 65–67]. This approach has three pillars, with the core being cybersecurity metrics. Recent advancement in security metrics includes [4–6, 8–10, 17, 20, 22, 35, 37, 39, 41, 43, 49, 52, 53, 74]. The two thrusts that are orthogonal to cybersecurity metrics are first-principle cybersecurity dynamics modeling [4, 5, 18, 21, 29, 30, 34, 36, 42, 51, 58, 59, 63, 64, 69, 70, 75, 76] and cybersecurity data analytics [1, 7, 13–16, 18, 19, 26, 27, 27, 28, 31–33, 44, 45, 54–57, 60, 68, 71, 72, 77, 78]. It would be fair to say that measuring and quantifying cybersecurity metrics is becoming a center of cybersecurity research.

CHAPTER 3: MEASURING RELATIVE ACCURACY OF MALWARE DETECTORS IN THE ABSENCE OF GROUND TRUTH

3.1 Chapter Introduction

Measuring security metrics is a vital but challenging open research problem that has not been well addressed. The major problems in measuring security metrics are two-fold: (1) what to measure, which questions how to define new, useful security metrics; and (2) how to measure, which asks how to devise new methods to measure security metrics. In this work, we are interested in answering the latter question, how to measure a security metric where the ground truth does not exist for the detection accuracy of a malware detector as well as for the class (i.e., malicious or benign) of the files.

When measuring the quality of malware detectors, many methods have been used based on certain heuristics such as using the labels of a few malware detectors as ground truth [38, 40, 46]. These heuristic-based approaches are troublesome because of the well-known fact that each malware detector has a different quality of detection accuracy. Although some methods have been proposed to answer *how to measure* security metrics [12, 24], measuring the relative accuracy of malware detectors has not been addressed in existing works. In particular, this work is inspired by our prior work [12] which measured the quality of malware detectors assuming that a voting-based estimation of detection accuracy is true. Unlike [12], this work aims to estimate the *relative* accuracy of malware detectors, which are obtained without making the assumptions that were made in [12]. Although the relative accuracy of malware detectors is weaker than the absolute accuracy, it is still useful in regards to comparing malware detectors. Therefore, this paper aims at answering the following research question: “How can we rank the accuracy of malware detectors in the absence of ground truth?"

Chapter contributions. This chapter makes the following **key contributions**: (i) This study offers a method to formulate how to estimate the *relative accuracy* of malware detectors. This method

can be used when one needs to choose one malware detector over others; and (ii) The proposed algorithm measuring the relative detection accuracy of a malware detector is validated based on a real world malware dataset consisting of 62 detectors, given synthetic data with known ground truth values.

The rest of the paper is organized as follows. Chapter 2 provides the overview of the related state-of-the-art approaches. Section 3.2 presents a problem statement and our proposed methodology to solve the given problem. Section 3.3 describes the experimental setup and results, with the discussion of key findings. Section 3.4 summarizes the present chapter.

3.2 Problem Statement and Methodology

3.2.1 Definitions of Relative Accuracy of Malware Detectors

Suppose that there are m files, denoted by $F_1, \ldots, F_j, \ldots, F_m$ and n malware detectors, denoted by $D_1, \ldots, D_i, \ldots, D_n$. The input dataset is represented by a matrix $V = (V_{ij})_{1 \leq i \leq n, 1 \leq j \leq m}$, which is defined where

$$V_{ij} = \begin{cases} 1 & \text{if } D_i \text{ detects } F_j \text{ as malicious,} \\ 0 & \text{if } D_i \text{ detects } F_j \text{ as benign,} \\ -1 & \text{if } D_i \text{ did not scan } F_j. \end{cases}$$

Vector $V_i = (V_{i1}, \ldots, V_{ij}, \ldots, V_{im})$, where $1 \leq i \leq n$, represents the outcome of using detector D_i to label m files.

With respect to a set of n detectors, the relative accuracy of detector i, denoted by T_i for $1 \leq i \leq n$, is defined over interval $[0,1]$, where 0 means the minimum degree (i.e., a zero degree) of *relative accuracy* while 1 indicates the maximum degree of *relative accuracy*.

Definition 1. (Properties of relative accuracy) *For a given set of files and a fixed set of n detectors, the relative accuracy of detector i, denoted by T_i for $1 \leq i \leq n$, is defined based on the labels assigned by the n detectors (including detector i itself). For simplicity, we define T_i as a real*

number in the range $[0,1]$.

We stress that the relative accuracy metric does not measure the true accuracy or trustworthiness of detectors. For example, consider three detectors D_1, D_2, and D_3, with respective true accuracy, 90%, 80%, or 70%. In this work, our goal is not to measure the accuracy of each detector. Instead, based on the labels of files assigned by these three detectors, we are more interested in knowing which detector is more accurate than others, leading to generating the ranks of examined detectors. Based on our proposed methodology, we obtain their respective relative accuracy as $T_1 = 100\%$, $T_2 = 90\%$, and $T_3 = 70\%$, which gives the performance of relative accuracy of the detectors: $D_1 > D_2 > D_3$. However, the relative accuracy does not approximate the true accuracy. Moreover, when we consider a set of files scanned by detectors, D_1, D_2, D_3, and D_4, letting the true accuracy of D_1-D_3 remain the same while the true accuracy of D_4 is 95%, then the resulting relative accuracy may be $T_1 = 80\%$, $T_2 = 60\%$, $T_3 = 50\%$, and $T_4 = 100\%$. This leads to the relative accuracy of the detectors being $D_4 > D_1 > D_2 > D_3$. This is because the detector with the highest relative accuracy is always normalized to have a relative accuracy of 100% and the relative accuracy is always measured over a set of detectors.

3.2.2 Methodology

The basic underlying idea of estimating the relative accuracy of malware detectors is to measure the similarity between each pair of detectors. To do so, we iteratively create the relative accuracy of the malware detectors, given that the initial relative accuracy of each detector is set to 1, assuming that each detector is equally accurate.

Similarity Matrix

To measure the relative accuracy of detectors, the concept of a *similarity matrix* is introduced to collectively represent the similarity between malware detectors according to their decisions in labeling files as benign or malicious. In this matrix, denoted by $\mathbf{S} = (S_{ik})$, the i-th row corresponds to detector D_i and the k-th column corresponds to detector D_k, where $1 \leq i,k \leq n$. Element S_{ik}

denotes the similarity between detectors D_i and D_k in terms of their capabilities in detecting malware. Naturally, we require (i) $S_{ik} = S_{ki}$ because the similarity should be symmetric; and (ii) $S_{ii} = 1$ for any $1 \leq i \leq n$. Intuitively, the similarity between D_i and D_k, S_{ik}, is defined by the ratio of the number of decisions where D_i and D_k agree with each other over the total number of files scanned by both detectors, D_i and D_k.

To clearly define S_{ik} in a modular fashion, two auxiliary matrices are defined: the *agreement matrix*, denoted by $\mathbf{A} = (A_{ik})_{1 \leq i,k \leq n}$, and the *count matrix*, denoted by $\mathbf{C} = (C_{ij})_{1 \leq i,k \leq n}$. Intuitively, A_{ik} is the number of files upon which detectors D_i and D_k give the same labels, namely

$$A_{ik} = A_{ki} = \sum_{\ell=1}^{m} \begin{cases} 1 & \text{if } V_{i\ell} = V_{k\ell} \wedge V_{i\ell} \neq -1 \wedge V_{k\ell} \neq -1 \\ 0 & \text{if } V_{i\ell} \neq V_{k\ell} \vee V_{i\ell} = -1 \vee V_{k\ell} = -1. \end{cases}$$

and C_{ik} is the number of files scanned by both detectors, D_i and D_k, namely

$$C_{ik} = C_{ki} = \sum_{\ell=1}^{m} \begin{cases} 1 & \text{if } V_{i\ell} \neq -1 \wedge V_{k\ell} \neq -1, \\ 0 & \text{if } V_{i\ell} = -1 \vee V_{k\ell} = -1. \end{cases}$$

Note that both $\mathbf{A}$ and $\mathbf{C}$ are symmetric. Given matrices $\mathbf{A}$ and $\mathbf{C}$, a similarity matrix $\mathbf{S}$ is defined as:

Definition 2. (Similarity matrix) *The similarity matrix* $\mathbf{S} = (S_{ik})_{1 \leq i,k \leq n}$ *is defined as the ratio of labels that detectors* D_i *and* D_k *agree, namely* $S_{ik} = \frac{A_{ik}}{C_{ik}}$*, implying that* S_{ik} *is symmetric.*

Algorithm for Computing Relative Accuracy

Definition 1 specifies the properties that a good relative accuracy definition should meet. Now we address a specific definition to measure the relative accuracy that satisfies those desired properties; the definition is shown in Algorithm 1.

The underlying idea of Algorithm 2 is as follows: The relative accuracy vector $\mathbf{T}$ is recursively calculated from the similarity matrix $\mathbf{S}$. The algorithm halts when the error δ is smaller than a

Algorithm 1 Computing relative accuracy

Input: Similarity matrix $\mathbf{S}_{n\times n}$; tolerable error threshold ε
Output: Relative accuracy vector $\mathbf{T} = [T_1, T_2, \ldots, T_n]^{\mathsf{T}}$

$\delta \leftarrow 2\varepsilon$
$\mathbf{T} \leftarrow ([1,1,\ldots,1]_{1\times n})^{\mathsf{T}}$
$\mathsf{NextT} \leftarrow ([0,0,\ldots,0]_{1\times n})^{\mathsf{T}}$
while $\delta > \varepsilon$ **do**
    $\mathsf{NextT} \leftarrow S \times T$
    $\mathsf{NextT} \leftarrow \mathsf{NextT} / \max(\mathsf{NextT})$
    $\delta \leftarrow \sum_{1\leq i\leq n} |T_i - \mathsf{NextT}_i|$
    $\mathbf{T} \leftarrow \mathsf{NextT}$
end while
Return T

threshold ε.

The similarity matrix $\mathbf{S}$ resembles a well-known correlation matrix consisting of correlation coefficients between a group of random variables. The major difference between these two kinds of matrices is that similarities are in the range of $[0,1]$ while correlations are in the range of $[-1,1]$. The sample version of a correlation matrix is the base for a statistical technique called *principal component analysis* where the *eigendecomposition* of the sample correlation matrix is used to find the dominating directions of variation in the data. In a similar sense, we use the similarity matrix to rank the relative accuracies of detectors. On the other hand, the recursive computation of the relative accuracy vector $\mathbf{T}$ may be reminiscent of a Markov Chain of n states. However, the similarity matrix $\mathbf{S}$ is not a probability transition matrix because the entries do not reflect probability.

3.3 Experiments and Results

In this section, we conduct experiments using a synthetic dataset with known ground truth to evaluate the approach and then use the approach to analyze a real dataset.

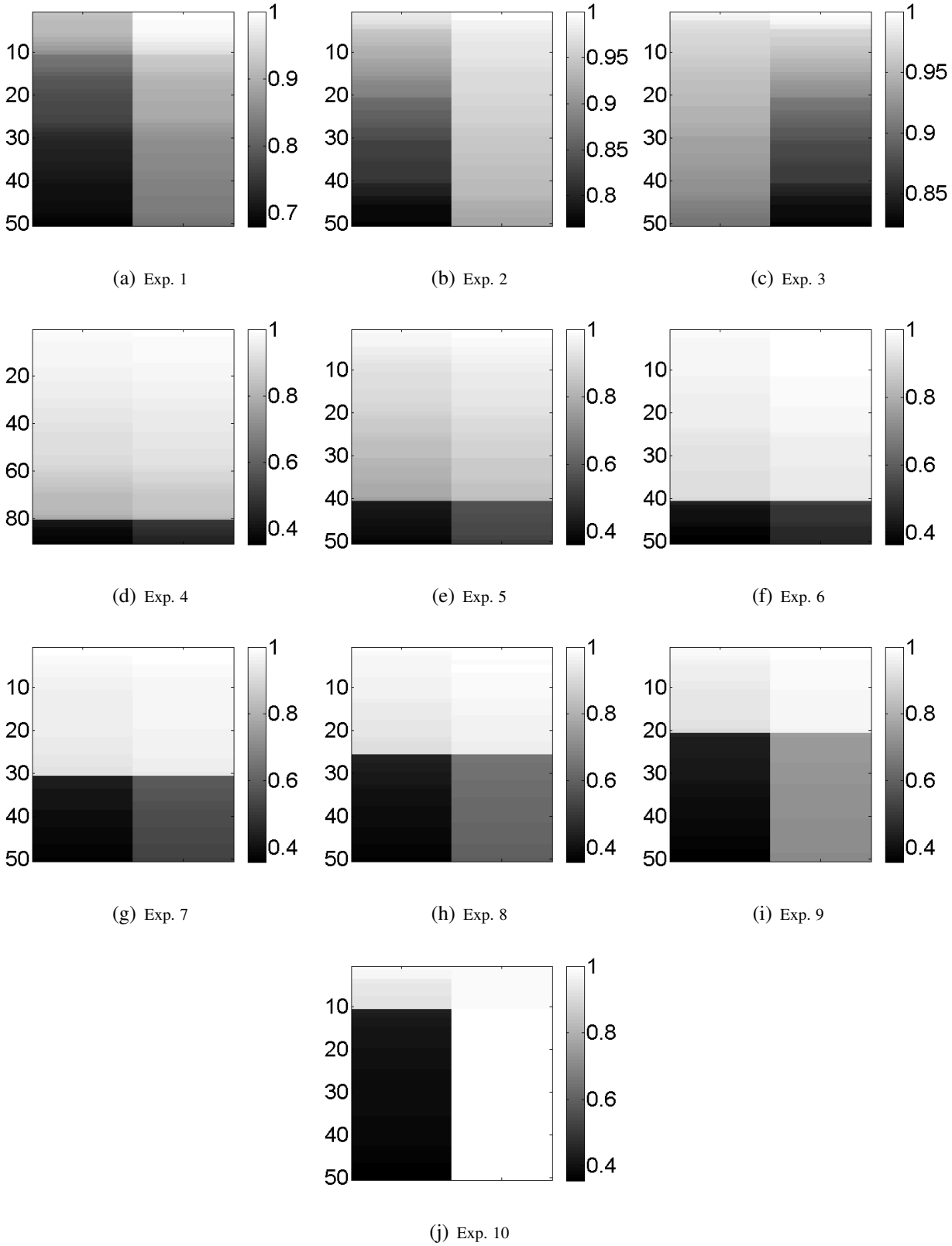
3.3.1 Experiments with Synthetic Data

Generating synthetic data. Three synthetic datasets of labels are generated for one million samples per dataset: **D1** contains 300,000 malicious files and 700,000 benign files; **D2** contains 500,000 malicious files and 500,000 benign files; and **D3** contains 700,000 malicious files and 300,000 benign files. Using these three datasets allows us to see the impact of the ratio between malicious and benign files.

Experimental setup. We consider 10 experiments where each experiment uses a number of detectors characterized by a true-positive rate (TP) and a true-negative rate (TN), while false-positive rates (FP) and false-negative rates (FN) are used to derive TP and TN, such as $TP = 1 - FN$ and $TN = 1 - FP$ [43]. Moreover, accuracy is defined as $\frac{\text{TP+TN}}{\text{TP+FP+TN+FN}}$ [43].

Experiments 1-5 aim to test a variety of situations with various distributions of accuracies of malware detectors.

- **Experiment 1** - Four sets of detectors of varying true accuracy rates are simulated:
  - 10 detectors with an accuracy range of 95% to 85%;
  - 10 detectors with an accuracy range of 85% to 75%;
  - 10 detectors with an accuracy range of 80% to 70%;
  - 20 detectors with an accuracy range of 75% to 65%.
- **Experiment 2** - Four sets of detectors of varying true accuracy rates are simulated:
  - 10 detectors with an accuracy range of 100% to 90%;
  - 10 detectors with an accuracy range of 95% to 85%;
  - 10 detectors with an accuracy range of 90% to 80%;
  - 20 detectors with an accuracy range of 85% to 75%.
- **Experiment 3** - The algorithm is tested with all 50 detectors that have a narrow range of accuracies (i.e., approximately the same detection capability):



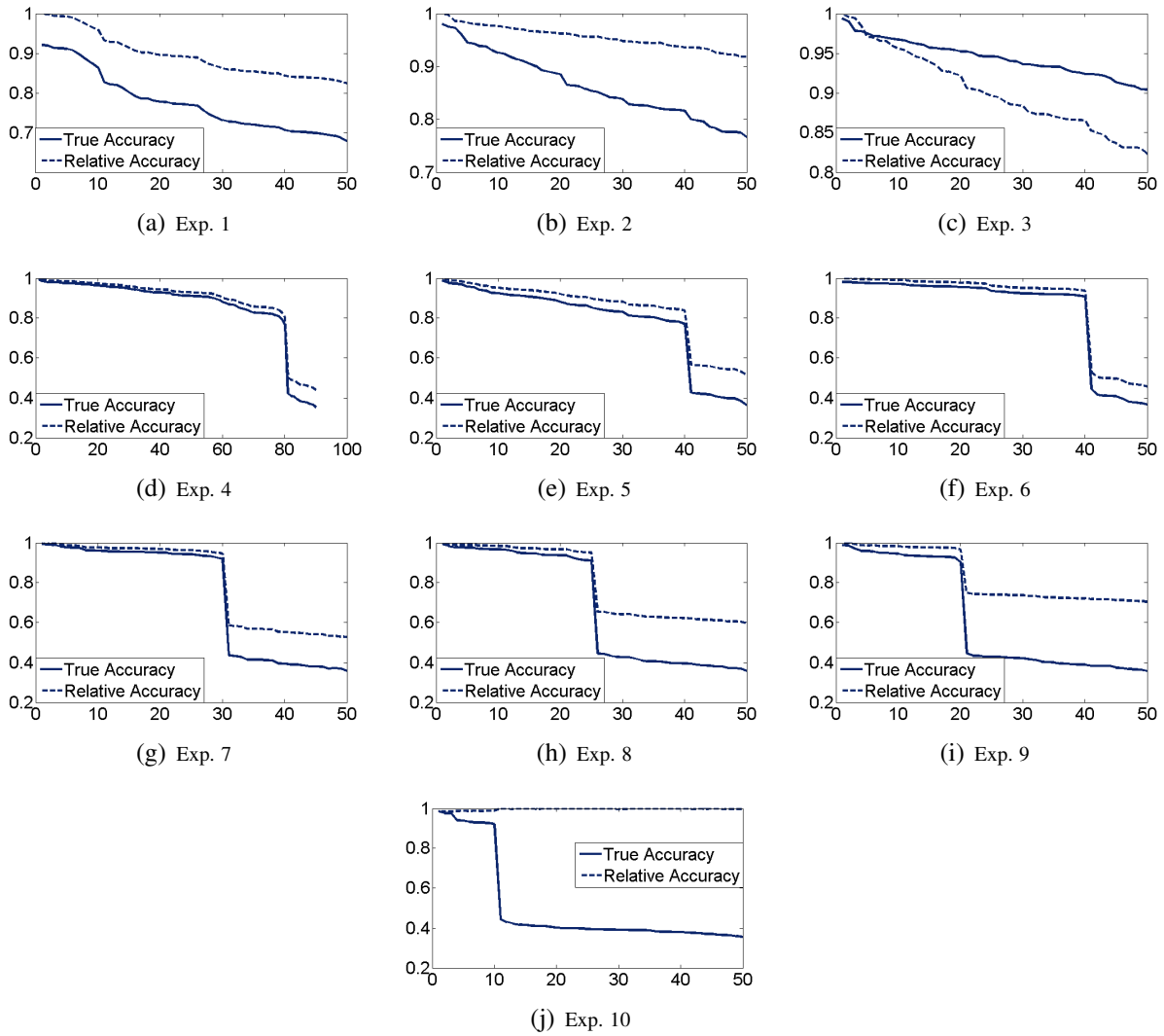
(a) Exp. 1 (b) Exp. 2 (c) Exp. 3

(d) Exp. 4 (e) Exp. 5 (f) Exp. 6

(g) Exp. 7 (h) Exp. 8 (i) Exp. 9

(j) Exp. 10

Figure 3.1: Experiment results with dataset **D1**: In each picture, the y-axis corresponds to the detectors in an experiment while the x-axis corresponds to the True Accuracy (left-hand half) and Relative Accuracy (right-hand half) of the detectors in color scale.



(a) Exp. 1 (b) Exp. 2 (c) Exp. 3

(d) Exp. 4 (e) Exp. 5 (f) Exp. 6

(g) Exp. 7 (h) Exp. 8 (i) Exp. 9

(j) Exp. 10

Figure 3.2: Experiment results with dataset **D1**: In each picture, the y-axis corresponds to the true accuracy and relative accuracy of each detector. The x-axis counts the individual detectors.

  - 50 detectors with an accuracy range of 100% to 90%.

- **Experiment 4** - The algorithm is tested with *poor* detectors that have their accuracies below 50%:
  - 50 detectors with an accuracy range of 100% to 90%;
  - 10 detectors with an accuracy range of 95% to 85%;
  - 10 detectors with an accuracy range of 90% to 80%;
  - 10 detectors with an accuracy range of 85% to 75%.
  - 10 detectors with an accuracy range of 45% to 35%.

- **Experiment 5** - The algorithm is tested with a higher ratio of *poor* detectors, as compared with Experiment 4:
  - 10 detectors with an accuracy range of 100% to 90%;
  - 10 detectors with an accuracy range of 95% to 85%;
  - 10 detectors with an accuracy range of 90% to 80%;
  - 10 detectors with an accuracy range of 85% to 75%;
  - 10 detectors with an accuracy range of 45% to 35%.

Experiments 6-10 are conducted to investigate the threshold where the algorithm is able to diagnose 'good' detectors over 'poor' detectors as the ratio of poor detectors increases. In all 5 experiments, the good detectors have accuracies that range from 100% to 90%, while the poor detectors have accuracies that range from 45% to 35%.

- **Experiment 6** - The algorithm is tested with 20% poor detectors:
  - 40 detectors with an accuracy range of 100% to 90%;
  - 10 detectors with an accuracy range of 45% to 35%.

- **Experiment 7** - The algorithm is tested with 40% poor detectors:

  - 30 detectors with an accuracy range of 100% to 90%;
  - 20 detectors with an accuracy range of 45% to 35%.

- **Experiment 8** - The algorithm is tested with 50% poor detectors:
  - 25 detectors with an accuracy range of 100% to 90%;
  - 25 detectors with an accuracy range of 45% to 35%.

- **Experiment 9** - The algorithm is tested with 60% poor detectors:
  - 20 detectors with an accuracy range of 100% to 90%;
  - 30 detectors with an accuracy range of 45% to 35%.

- **Experiment 10** - The algorithm is tested with 80% poor detectors:
  - 10 detectors with an accuracy range of 100% to 90%;
  - 40 detectors with an accuracy range of 45% to 35%.

Fig. 3.1 plots experimental results with **D1**. For each experiment, we look into the *true accuracy* and *relative accuracy* of each detector. We observe that except Exp. 10, the order of the accuracies (e.g., detector 2 is more accurate than detector 3) is preserved by the relative accuracy (i.e., detector 2 has a higher relative accuracy than detector 3). This does not hold for Exp. 10 because 40 (out of the 50) detectors are 'poor.'

Fig. 3.2 maps both the *true accuracy* and the *relative accuracy* of each detector for the experiments in **D1**. Each experiment shows that there is one detector whose relative accuracy is 100%, which is implied by Algorithm 1. We also observe that the relative accuracy of a detector is *not* the same as the accuracy of the detector; in contrast, it can deviate significantly. However, note that the slope of the *relative accuracy* matches with the changes observed in the slope of the *true accuracy*, with minor differences in the degree of the change. This implies that recovering the true accuracy of each detector should be possible, if a proper method is provided. Experiment 10 is the exception again, showing that the rating system was overwhelmed by the 'poor' detectors

which outranked the 'good' by a factor of 4 to 1. Furthermore, Experiments 6 through 9 show that with similar graphs for true accuracy, the difference in measurements between true accuracy and relative accuracy increases as the level of uncertainty increases.

For **D2** and **D3**, the results are almost identical to **D1**. The changes in the file distribution changed the initial ordering, but the results for comparing the true accuracy with the relative accuracy are trivial, so we didn't include them here.

Algorithm 1 provides useful and significant results whether the (true) accuracy of detectors is continuously distributed across a wide range (as in Experiments 1 and 2), distributed across a narrow range (as in Experiment 3), or distributed across a wide range in a discontinuous fashion (as in Experiments 4 and 5). Experiments 6 through 10 show that through all three datasets, Algorithm 1 provides reliable results up to the point where there are 4 poor detectors per good detector. At this threshold, the poor detectors begin to be rated above the good detectors due to the noise introduced by sheer numbers.

Summarizing the experiment results with synthetic datasets **D1**-**D3**, we obtain the following insight:

Insight 1. *Algorithm 1 is useful because it can compute the relative accuracy, or relative ranking, of malware detectors as long as the number of 'poor' detectors is not overwhelming.*

3.3.2 Applying the approach to evaluate a real dataset

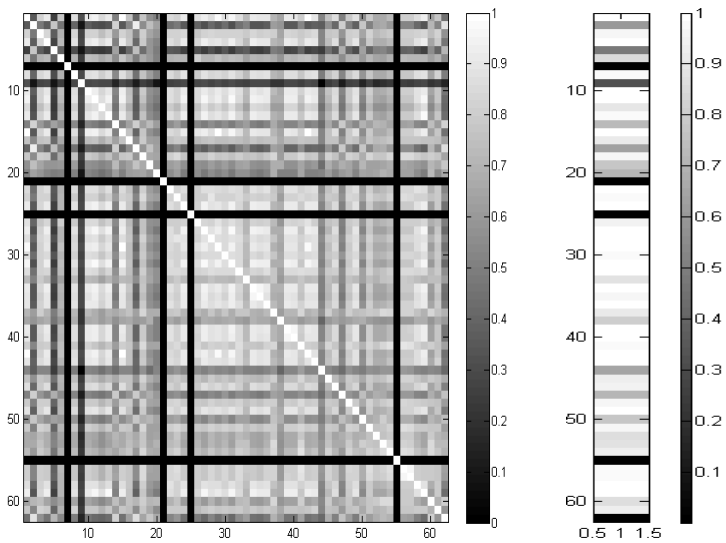
The dataset was collected from VirusTotal. It contains a corpus of $m \approx 10.7$ million files, each of which was scanned by up to $n = 62$ anti-malware detectors, but some files were not scanned by every detector. Each detector labels a file it scanned as malicious ("1") or benign ("0"). The dataset is transformed to matrix $\mathbf{V_{ij}}_{n \times m}$, from which we derive a similarity matrix $\mathbf{S}$ and a relative accuracy vector $\mathbf{T}$ according to Algorithm 1.

Table 3.1 summarizes the relative accuracy of the 62 detectors. We observe that the relative accuracy of 35 detectors is in $[1, 0.95]$, 11 detectors in $[0.85, 0.95]$, 7 detectors in $[0.7, 0.8]$, 3 detectors in $[0.6, 0.7]$, 1 detector in the 0.4 range, 1 detector in the 0.3 range, and 4 detectors at

Table 3.1: The detector name and trust value from the processed data from VirusTotal.

Detector Name	Trust	Detector Name	Trust
BitDefender	100%	Symantec	95.78%
Ad-Aware	99.85%	CAT-QuickHeal	95.46%
McAfee	99.63%	Panda	95.22%
GData	99.62%	Emsisoft	95.22%
Kaspersky	99.48%	Zillya	93.20%
AhnLab-V3	99.45%	TotalDefense	92.80%
VIPRE	99.39%	nProtect	92.64%
MicroWorld-eScan	99.38%	Kingsoft	91.74%
Avast	99.37%	Bkav	91.68%
AVG	99.34%	Avira	89.93%
F-Pro	99.32%	Jiangmin	88.99%
K7AntiVirus	99.30%	TheHacker	88.33%
NANO-Antivirus	99.22%	Tencent	86.53%
F-Secure	99.05%	ViRobot	86.53%
McAfee-GW-Edition	99.04%	ALYac	85.91%
DrWeb	98.98%	Malwarebytes	79.80%
ESET-NOD32	98.78%	SUPERAntiSpyware	77.63%
Sophos	98.63%	ClamAV	77.47%
VBA32	98.45%	Baidu-International	73.13%
Comodo	98.45%	Qihoo-360	71.68%
Ikarus	98.44%	CMC	70.72%
AVware	98.31%	Zoner	70.17%
Fortinet	97.79%	Norman	65.44%
Cyren	97.53%	ByteHero	64.01%
TrendMicro	97.43%	AegisLab	60.10%
Microsoft	97.27%	Alibaba	47.14%
TrendMicro-HouseCall	97.03%	Arcabit	32.09%
Antiy-AVL	96.87%	AntiVir	5E-04%
Agnitum	96.73%	Commtouch	5E-04%
K7GW	96.50%	DrWebSE	3E-04%
Rising	95.81%	TotalDefense2	2E-06%

the order of magnitude of 10^{-6}. The extremely low relative accuracy of the last four detectors can be attributed to the following: (i) these detectors match poorly with the decisions of the other detectors; (ii) these detectors provide monotonous detection, meaning that they label all files either as 1 or 0; and (iii) these detectors scanned fewer than 1% of the files. Therefore, these detectors are correctly labeled as inaccurate.



(a) S (b) T

Figure 3.3: Relative accuracy vector T and associated similarity matrix S.

Fig. 3.3 (a) shows the similarity matrix while Fig. 3.3 (b) shows the *relative* accuracy of the detectors. The similarity matrix provides a good visual intuition as to why several detectors were rated low. The inherent symmetry is also observed. In order to reach the steady state, we ran 8 iterations of the algorithm to reach a resolution of $\varepsilon = 10^{-9}$.

Fig. 3.4 shows the relative accuracy of the 62 detectors in the real-world data. Summarizing the preceding discussion, we obtain the following insight:

Insight 2. *A few detectors in the real-world dataset are not really useful. In traditional majority voting, these poor detectors would be given equal weight votes with the good detectors. With the ability to discern which detectors are more reliable, more appropriate voting weights can be*

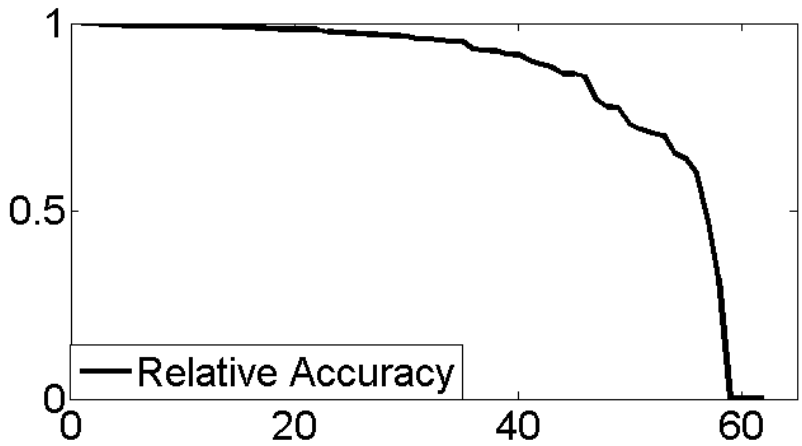


Figure 3.4: Relative accuracy (y-axis) of the 62 detectors (x-axis) in the real-world dataset. Note that true accuracies of the detectors are not known and therefore not plotted.

applied to the appropriate detectors.

3.4 Chapter Summary

We formulated the problem of estimating the *relative accuracy* of malware detectors in the absence of ground truth and presented an algorithm to derive the relative accuracy. We validated the proposed algorithm based on real-world datasets from VirusTotal, given synthetic data with ground truth. Through the extensive experimental study, we found that the proposed algorithm of estimating the relative accuracy of malware detectors is capable of ranking the relative accuracies of the 62 real-world detectors which scanned millions of files. In particular, we identified 4 detectors that not only are useless, but also may do more harm than good.

CHAPTER 4: LEVERAGING RELATIVE ACCURACY TO INFER METRICS IN THE ABSENCE OF GROUND TRUTH

4.1 Chapter Introduction

Cyber security metrics is one of the most notoriously difficult open problems, despite the numerous efforts that have been made by the research community [38, 40, 46]. Measuring cyber security is recognized as a vital but challenging open research problem that has not been well addressed. Major problems in measuring cyber security are two-fold: (1) *what to measure*, which is a question of how to define new and useful cyber security metrics; and (2) *how to measure*, which asks how to devise new methods to measure well-defined cyber security metrics . In this work, we cope with the latter question, *how to measure* well-defined cyber security metrics in the context of malware detection. In this setting, well-defined cyber metrics may be easy to measure given some relevant ground-truth data [12,24]. The challenge is to measure such metrics when the desired ground-truth data is not available, which is often encountered in practice because ground-truth is often hard to obtain.

For measuring the quality of malware detectors without knowing the ground truth, various kinds of heuristic methods have been proposed, such as using the labels of a few malware detectors as ground truth [38,40,46]. These heuristic-based approaches are troublesome because of the well-known fact that each malware detector has a different quality of detection accuracy. Towards the ultimate goal of principled measurement of malware detection metrics in the absence of ground-truth data, some methods have been proposed while making various kinds of assumptions [12, 24]. It remains to be understood what metrics can be measured under what weakest assumptions, given that no ground-truth data is given. This is indeed a broad problem domain because many research problem can be formulated, as demonstrated by the particular problem formulation we introduce in the present paper. One new approach is to measure the relative accuracy of malware detectors in the absence of ground-truth labels. Measuring relative accuracy is useful when ground-truth labels

are not known, for example in the process of decision-making in regards to choosing the more accurate malware detectors to conduct majority voting between malware detectors. Knowing the relative accuracy would represent a principled step beyond the pure heuristic of "blindly" using malware detectors to formulate an ensemble for majority voting.

Chapter contributions. In this chapter, we make three contributions. First, we propose a new algorithm for measuring the relative accuracy of malware detectors in the absence of ground truth. The innovation of the algorithm lies in the introduction of the concept of *bellwether detector*, which we introduce in this paper to define something as a leading indicator or indicator of trends. Intuitively speaking, a bellwether detector is a dumb classifier and always has an accuracy of $1/n$, where n is the number of classification classes in question and randomly assigns a value without considering any information from the underlying data. As a step towards understanding the properties of the relative accuracy algorithm, we contrast it with the Principal Component Analysis (PCA) and the correlation matrix in Statistics. Our finding is that the Similarity Matrix used in our approach is orthogonal to the Correlation Matrix used by PCA. This means that the ordered eigenvectors of the Similarity Matrix used in our approach represent the ordered directions according to which detectors most often make similar decisions. In other words, the detectors corresponding to the higher ordered eigenvectors are to be more trusted than the detectors corresponding to the lower ordered ones. The mathematical dimensionality induced by the eigenvectors shows why the underlying numbers indicate that those detectors should be more trusted. In other words, detectors corresponding to the higher ordered eigenvectors provide a mathematically more pivotal decision than the lower ordered ones.

Second, we leverage the relative accuracy derived from the algorithm to measure the ultimately desired *absolute* accuracy of malware detectors, such as their true-positive rate, false-positive rate, true-negative rate, and false-negative rate in the absence of ground truth. For this purpose, we propose an algorithm that treats each detector's relative accuracy as its voting weight. This allows us to aggregate the detectors' predictions together with their weights to derive a total score for each file, which can be interpreted as the degree of maliciousness of the file. Each detector is then

assigned an inferred accuracy score based on their predictions on the files with respect to the scores of the files inferred from the preceding process. Intuitively, this method works because a higher weighted detector is more trusted and provides a more valued vote.

Third, we evaluate the usefulness of both the relative accuracy algorithm and the resulting absolute accuracy algorithm by conducting experiments using both synthetic data and real-world data. For the synthetic data, we know the ground truth in terms of (i) which file is malicious or not and (ii) the malware detectors' absolute accuracy. Experimental results using the synthetic data show that every malware detector's absolute accuracy metrics inferred by our algorithms are 99.9% accurate with respect to the known ground truth of the synthetic data. This justifies that the proposed algorithms can be applied to the real-world data, for which the ground truth is not known. Our findings obtained by applying the algorithms to the real-world data show that for the most part, the algorithms provide consistent results. We also observe how an inconsistent number of measurements between detectors can give false readings, highlighting the need for proper data sanitation. Additionally, we observe that the removal of problematic detectors does not greatly influence the results of the remaining detectors.

Chapter outline. The rest of the chapter is organized as follows. Section 4.2 presents the problem statement and our methodology. Section 4.3 describes our experiments and results. Section 4.4 summarizes the present chapter.

4.2 Problem Statement and Methodology

4.2.1 From Relative Accuracy to Absolute Accuracy

Consider a set of m files, denoted by $F_1, \ldots, F_m$, and n malware detectors, denoted by $D_1, \ldots, D_n$. The input is a matrix $V = (V_{ij})_{1 \leq i \leq n, 1 \leq j \leq m}$, where

$$V_{ij} = \begin{cases} 1 & \text{if } D_i \text{ detects } F_j \text{ as malicious,} \\ 0 & \text{if } D_i \text{ detects } F_j \text{ as benign,} \\ -1 & \text{if } D_i \text{ did not scan } F_j. \end{cases}$$

The vector $V_i = (V_{i1}, \ldots, V_{im})$, where $1 \leq i \leq n$, represents the outcome of using detector D_i to label the m files.

Given this input, the research problem is to infer, *in the absence of ground truth*, the following standard malware detection metrics [43]. Let TP denote the number of true-positives, TN the number of true-negatives, FP the number of false-positives, and FN the number of false-negatives. The true-positive rate is defined as $TPR = \frac{TP}{TP+FN}$, the true-negative rate is defined as $TNR = \frac{TN}{TN+FP}$, and the accuracy is then defined as $\frac{TP+TN}{TP+TN+FP+FN}$.

In order to achieve this goal, we proposed an approach that leverages the innovative notion of *relative accuracy* [2].

Definition 3 (relative accuracy [2])**.** *The relative accuracy of detector D_i where $1 \leq i \leq n$, denoted by T_i, is defined by a number the interval $[0,1]$ such that a larger value indicates a higher accuracy.*

The relative accuracy is measured in an *ordinal* scale [43]. Recall that an ordinal scale is a discrete ordered set that permits comparisons between two measurements (e.g., "relatively more accurate" or "relatively less accuracy" in the context of the present paper). However, an ordinal scale has two drawbacks [43]: (i) The *equal-distance* property is not assured. This means that the distance between two adjacent measurements or points on the same scale is not necessarily constant and that it does not make sense to compare the difference between one pair of measurements and

the difference between another pair of measurements. (ii) The notion of *origin* or measurement value 0 is not defined. A scale assuring the preceding (i) and (ii) is called a *ratio* scale [43]. The metrics we aim to estimate, namely TPR, TNR and Accuracy, are measured in a ratio scale. This means that our method of leveraging relative accuracy to estimate absolute accuracy may not be perfect. Relative Accuracy calculated using an iterative algorithm has been previously explored [2] and shown to be effective at ranking malware detectors, but did not provide nor measure the ability to recover true metrics from those detectors. In this work, we show how the Relative Accuracy algorithm can be improved through refinement, how to translate the algorithm through the use of eigenvalues, why this translation makes sense, and how the addition of a random noise generator can improve our overall measurements. Further, we go a step further and show how the True Accuracy of the detectors can be recovered using the Relative Accuracy measurements, and why this is better than the default majority voting scheme when evaluating the ground truth classification of potential malware files.

4.2.2 Methodology

Similarity Matrix

Denote by $D = \{D_1, \ldots, D_n\}$ a universe of n detectors. Denote by $F = \{F_1, \ldots, F_m\}$ a universe of m files. Let matrix $V = (V_{ij})_{1 \leq i \leq n, 1 \leq j \leq m}$ be defined as

$$V_{ij} = \begin{cases} 1 & \text{if } D_i \text{ detects } F_j \text{ as malicious,} \\ 0 & \text{if } D_i \text{ detects } F_j \text{ as benign,} \\ -1 & \text{if } D_i \text{ did not scan } F_j. \end{cases}$$

Each vector $V_i = (V_{i1}, \ldots, V_{im})$, where $1 \leq i \leq n$, is detector D_i's labels each on the m files.

Given two detectors, D_i and D_k, where $1 \leq i, k \leq n$ and $i \neq k$, the following definitions aim to describe the similarity between two detectors in terms of how they label files.

Definition 4 (agreement matrix [2])**.** *The agreement matrix counts the number of files that are*

labeled by detectors D_i and D_k consistently, namely

$$A_{ik} = A_{ki} = \sum_{\ell=1}^{m} \begin{cases} 1 & \text{if } V_{i\ell} = V_{k\ell} \wedge V_{i\ell} \neq -1 \wedge V_{k\ell} \neq -1 \\ 0 & \text{if } V_{i\ell} \neq V_{k\ell} \vee V_{i\ell} = -1 \vee V_{k\ell} = -1. \end{cases}$$

Definition 5 (count matrix [2])**.** *The count matrix describes the number of files that have been labelled by both D_i and D_k, regardless of whether their labels are consistent or not. That is,*

$$C_{ik} = C_{ki} = \sum_{\ell=1}^{m} \begin{cases} 1 & \text{if } V_{i\ell} \neq -1 \wedge V_{k\ell} \neq -1, \\ 0 & \text{if } V_{i\ell} = -1 \vee V_{k\ell} = -1. \end{cases}$$

Definition 6 (similarity between detectors)**.** *The notion of similarity between two detectors D_i and D_k measures their degree of consistency in labeling files.*

Definition 6 is a general metric that can be instantiated in multiple ways to quantify the consistency between two detectors. It is general in the sense that it can be adapted to accommodate various kinds of information relevant to the detectors, such as: the features that are used by the detectors to predict labels of files, the labels that are relevant (e.g., binary vs. multi-class classification). Although Definition 6 can be instantiated in many ways, any meaningful instantiation should satisfy at least the following:

Definition 7 (similarity requirements [2])**.** *A quantitative measure of similarity, S_{ik} between two detectors D_i and D_k where $1 \leq i,k \leq n$, should satisfy two natural requirements:*

- *$S_{ik} = S_{ki}$, because similarity should be symmetric; and*
- *$S_{ii} = 1$ for any $1 \leq i \leq n$ to reflect self-similarity.*

A particular instantiation of Definition 6 was proposed in [2] and reviewed below.

Definition 8 (concrete similarity [2])**.** *Similarity is defined as the ratio of the number of labels where D_i and D_k agree, A_{ik}, over the total number of files that are processed or scanned by both,*

C_{ik}. *Namely*

$$S_{ik} = \frac{A_{ik}}{C_{ik}}$$

By considering all of the n detectors, Definition 8 naturally leads to the following notion of the *similarity matrix*:

Definition 9 (similarity matrix [2])**.** *Given n detectors and m files that are scanned by these detectors, the similarity matrix describing the consistency between the labeling behavior of the detectors and is denoted by* $\mathbf{S} = (S_{ik})_{1 \leq i,k \leq n}$*, where element* S_{ik} *is given by Definition 8.*

Algorithm for Computing Relative Accuracy

Algorithm 2 provides a method for computing the Relative Accuracy values of detectors. Recall that relative accuracy T_i of detector D_i where $1 \leq i \leq n$ is defined to be in the interval $[0,1]$ as an ordinal scale.

Algorithm 2 Computing relative accuracy [2]

Input: Similarity matrix $\mathbf{S}_{n \times n}$; tolerable error threshold ε
Output: Relative Accuracy vector $\mathbf{T} = [T_1, T_2, \ldots, T_n]^{\mathsf{T}}$

```
δ ← 2ε
T ← ([1, 1, ..., 1]_{1×n})^T
while δ > ε do
    NextT ← S × T
    NextT ← NextT / max(NextT)
    δ ← Σ_{1≤i≤n} |T_i − NextT_i|
    T ← NextT
end while
Return T
```

At a high level, the idea behind Algorithm 2 is described as follows: The relative accuracy vector $\mathbf{T}$ is recursively calculated from the similarity matrix $\mathbf{S}$, which is elaborated below. The algorithm halts when the error δ is smaller than a threshold ε.

The similarity matrix $\mathbf{S}$ resembles a well-known correlation matrix consisting of correlation coefficients between a group of random variables. The major difference between these two kinds

of matrices is that similarities are in the range of $[0,1]$ while correlations are in the range of $[-1,1]$. The sample version of a correlation matrix is the base for a statistical technique called *principal component analysis* where the *eigendecomposition* of the sample correlation matrix is used to find the dominating directions of variation in the data. In a similar sense, we use the similarity matrix to rank the relative accuracy of detectors. The detail on the relationship between relative accuracy and principal component analysis will be discussed in Subsection 4.2.2. On the other hand, the recursive computation of the relative accuracy vector $\mathbf{T}$ may be reminiscent of a Markov Chain of n states. However, the similarity matrix $\mathbf{S}$ is not a probability transition matrix because the entries do not reflect probability.

Algorithmic Modification to Allow Variable Step Size

Examining the main loop of 2, it can be observed that during each iteration the algorithm repeatedly multiplies the Relative Accuracy vector $\mathbf{T}$ by the Similarity Matrix $\mathbf{S}$ from the left hand side, scales that result by the largest value in the resulting matrix, and then substitutes that new vector for $\mathbf{T}$ in the next iteration.

Let $\mathbf{1}$ be the initial Relative Accuracy vector $([1,1,\ldots,1]_{1\times n})^{\mathsf{T}}$ and N_i be the normalization scalar of iteration i. Given the associative property of scalar multipliers, let $N_* = N_1 \times N_2 \times \ldots \times N_i$. Algorithm 2 can then be rewritten as

$$\begin{aligned}
\mathbf{T} &= \underbrace{S\ldots(S(S}_{i \text{ times}}(\mathbf{1})/N_1)/N_2)\ldots/N_i \\
&= \underbrace{S\ldots(S(S}_{i \text{ times}}(\mathbf{1})))/N_* \\
&= S^i\mathbf{1}/N_* \qquad (4.1)
\end{aligned}$$

Due to this simplification, the complexity of algorithm 2 can be reduced to allow a step-size greater than 1, and delay the scalar multiplication until the end of the cycle, as shown in Alogrithm 3.

This eliminates the need to track the Relative Accuracy vector T explicitly, as the values are not

Algorithm 3 Computing relative accuracy with step size

Input: Similarity matrix $\mathbf{S}_{n\times n}$; tolerable error threshold ε; step size x
Output: Relative Accuracy vector $\mathbf{T} = [T_1, T_2, \ldots, T_n]^\mathsf{T}$

```
δ ← 2ε
T ← ([1, 1, ..., 1]_{1×n})^T
c ← 1 (Initial Step)
x ∈ ℕ (Step Size)
CurrentT ← S^c × T / max(S^c × T)
while δ > ε do
    NextT ← S^{c+x} × CurrentT / max(S^{c+x} × CurrentT)
    δ ← Σ_{1≤i≤n} |NextT_i − CurrentT_i|
    c ← c + x
    CurrentT ← NextT
end while
Return T ← CurrentT
```

iteratively generated, but are instead calculated explicitly. The ability to utilize a step size greater than 1 produces a scaled reduction in the number of iterations needed.

Algorithmic Modification to Analyze Eigenvalues

Due to the fact that the similarity matrix **S** is symmetric, it is known that the eigenvalues of **S** are all real numbers. Furthermore, **S** is positive semi-definite if we assume the following condition.

- Condition 1: All detectors analyze all files. Therefore, $C_{ik} = m$ and V_{ik} consists of only 0 or 1 entries for all i, k, and $S_{ik} = \sum_{l=1}^{m} I(V_{il} = V_{kl})/m$.

Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0$ and $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n$ be respectively the eigenvalues and eigenvectors of the similarity matrix S. Then the spectral decomposition of S gives

$$S = UDU^\mathsf{T} = \sum_{i=1}^{n} \lambda_i \mathbf{e}_i \mathbf{e}_i^\mathsf{T}, \tag{4.2}$$

where $U = (\mathbf{e}_1, \ldots, \mathbf{e}_n)$ is an orthonormal matrix such that $U^\mathsf{T}U = UU^\mathsf{T} = I$, and $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$.

Therefore, combining (4.1) and (4.2), we have

$$
\begin{aligned}
N_*\mathbf{T} = S^i\mathbf{1} = (UDU^\top)^r\mathbf{1} &= UD^rU^\top\mathbf{1} \\
&= (\sum_{i=1}^{n} \lambda_i^r \mathbf{e}_i\mathbf{e}_i^\top)\mathbf{1} \\
&= \sum_{i=1}^{n} \lambda_i^r \mathbf{e}_i(\mathbf{e}_i^\top\mathbf{1})
\end{aligned} \tag{4.3}
$$

where $D^r = \text{diag}(\lambda_1^r, \ldots, \lambda_n^r)$.

Substituting $U \times D^i \times U^T$ for S^i in Algorithm 3 allows the algorithm to be rewritten as Algorithm 4.

From the standpoint of Relative Accuracy, we observe that Algorithm 4 is derived directly from Algorithm 2. Algorithm 4 is simply much more efficient.

Algorithm 4 Computing relative accuracy using eigenvalues

Input: Similarity matrix $\mathbf{S}_{n\times n}$; tolerable error threshold ε; step size x
Output: Relative Accuracy vector $\mathbf{T} = [T_1, T_2, \ldots, T_n]^\top$

```
δ ← 2ε
c ← 1 (Initial Step)
x ∈ ℕ (Step Size)
[U, D] ← eig(S)
CurrentT ← U × D^c × U^T / max(U × D^c × U^T)
while δ > ε do
    NextT ← (U × D^(c+x) × U^T) / max(U × D^(c+x) × U^T)
    δ ← Σ_{1≤i≤n} |NextT_i − CurrentT_i|
    c ← c + x
    CurrentT ← NextT
end while
Return T ← CurrentT
```

Algorithmic Convergence

Algorithm 4 provides a good form for examining convergence over multiple steps. The Similarity Matrix, $S_{n\times n}$, is by definition a real symmetric matrix. Thus, it can undergo eigendecomposition. Let $S_{n\times n} = UDU^T$ be that decomposition, where U is made up of the eigenvectors of S and D is a diagonal matrix whose values are the corresponding eigenvalues. Since S is a real symmetric

matrix, U is an orthogonal matrix and $U^T = U^{-1}$. Specifically, if a step size of 1 is assumed, the xth iteration of the loop provides a Relative Accuracy vector of

$$T_x \leftarrow U \times D^x \times U^T / max(U \times D^x \times U^T) \tag{4.4}$$

Utilizing the fact that $U \times U^T = U \times U^{-1} = I$,

$$\begin{aligned}
T_x &= \frac{U \times D^x \times U^T}{max(U \times D^x \times U^T)} \\
&= \frac{U \times D_1 \times D_2 \times \cdots \times D_x \times U^T}{max(U \times D_1 \times D_2 \times \cdots \times D_x \times U^T} \\
&= \frac{U \times D_1 \times (U^T \times U) \times D_2 \times (U^T \times U) \ldots D_x \times U^T}{max(U \times D_1 \times (U^T \times U) \times D_2 \times (U^T \times U) \ldots D_x \times U^T} \\
&= \frac{(U \times D_1 \times U^T) \times (U \times D_2 \times U^T) \ldots (U \times D_x \times U^T)}{max((U \times D_1 \times U^T) \times (U \times D_2 \times U^T) \ldots (U \times D_x \times U^T)} \\
&= \frac{(U \times D \times U^T)_1 \times (U \times D \times U^T)_2 \ldots (U \times D \times U^T)_x}{max((U \times D \times U^T)_1 \times (U \times D \times U^T)_2 \ldots (U \times D \times U^T)_x)} \\
&= \frac{(U \times D \times U^T)^x}{max((U \times D \times U^T)^x)} \\
&= \frac{(U \times D \times U^T)^x}{(max(U \times D \times U^T))^x}
\end{aligned} \tag{4.5}$$

Let V represent vector $U \times D \times U^T$, v an arbitrary element of V and $v^* = \max(V)$. Then $\forall v \in V$, v in range $(0 \ldots 1)$, $max(V^x) = max(V)^x$ and $v \leq v^*$.

$$\lim_{x \to +\infty} \frac{v^x}{v^{*x}} = \begin{cases} 1 & \text{if } v = \max(V) \\ 0 & \text{if } v < \max(V) \end{cases}$$

Thus, for every element of $v \in V$ where $v \neq v^*$, each iteration of the algorithm monotonically approaches 0. Since there is defined an arbitrarily small ε, there is guaranteed to be an iteration where the difference between this step and the next, represented by δ, finds $\delta \leq \varepsilon$, which guaran-

tees the algorithm converges and exits.

Since algorithm 4 is mathematically equivalent to algorithm 2, this guarantee also applies to algorithm 2.

Why Relative Accuracy Makes Sense

In this section we derive the true meaning of the relative accuracy through a careful examination of its close relationship to the principal component analysis (PCA), a commonly used statistical procedure for efficient representation of data. Consider an $m \times n$ data matrix $\mathbf{X}$ with entries $x_{ij}, 1 \leq i \leq m, 1 \leq j \leq n$. It represents m observations on n variables, say, $X_1, \ldots, X_n$. In particular, the ith row $(x_{i1}, \ldots, x_{in})$ of $\mathbf{X}$ represents the ith observation vector on all the n variables, and the jth column represents all the m observations made on variable X_j. The first step in PCA is to calculate the sample correlation matrix $\mathbf{R}$ between these n variables as follows. Let $\bar{x}_j = \sum_{i=1}^{m} x_{ij}/m$ and $\sigma_j^2 = \sum_{i=1}^{m}(x_{ij} - \bar{x}_j)^2/(m-1)$ be respectively the sample mean and sample variance of variable X_j. The sample correlation between variables X_j and X_k is $r_{jk} = \{(m-1)\sigma_j\sigma_k\}^{-1}\sum_{i=1}^{m}(x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)$. Then the sample correlation matrix $\mathbf{R}$ is obtained by assembling the entries r_{jk} into an $n \times n$ matrix. The next step in PCA is to perform a spectral decomposition of $\mathbf{R}$. Specifically, let $\gamma_1 \geq \gamma_2 \geq \ldots, \geq \gamma_n \geq 0$ and $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n$ with $\mathbf{f}_i = (f_{i1}, \ldots, f_{in})^\mathsf{T}$ be respectively the eigenvalues and eigenvectors of the correlation matrix $\mathbf{R}$. Then the spectral decomposition of $\mathbf{R}$ is

$$\mathbf{R} = \sum_{i=1}^{n} \gamma_i \mathbf{f}_i \mathbf{f}_i^\mathsf{T}. \tag{4.6}$$

And $\mathbf{f}_i$ is now the weight vector for the ith *principal component* defined as $Z_i = \sum_{j=1}^{n} f_{ij} X_j$, a linear combination of the original variables $X_1, \ldots, X_n$. Due to the orthogonality between $\mathbf{f}_i$, the principal components Z_is are uncorrelated with each other, that is, their pairwise correlations are all zeros. Recall that each row of the data matrix $\mathbf{X}$ represents an observation. If one envisions that the m observations are plotted in the n-dimensional space with each dimension representing one X_j, then the first principal component weight vector $\mathbf{f}_1$ represents the direction where these observations show the most variation (i.e., have the widest spread). The second principal component weight

vector $\mathbf{f}_2$ is orthogonal to $\mathbf{f}_1$ and represents the direction of the most variation among all the directions perpendicular to $\mathbf{f}_1$. The third principal component weight vector $\mathbf{f}_3$ is orthogonal to both $\mathbf{f}_1$ and $\mathbf{f}_2$, and represents the direction of the most variation among all the directions perpendicular to both $\mathbf{f}_1$ and $\mathbf{f}_2$.

Recall that our relative accuracy measure groups detectors according to the magnitudes of the corresponding entries in the vector $\mathbf{T}$. Note that if $\lambda_i > \lambda_j$ then as r increases the gap between λ_i^r and λ_j^r will become larger and larger. At some point, we will have $\lambda_i^r >> \lambda_j^r$, that is, λ_i^r becomes much larger than λ_j^r making the latter almost negligible. We now examine (4.3) equipped with this knowledge. In practical calculation, we may expect a strict ordering of the (positive) eigenvalues such that $\lambda_1 > \lambda_2 > \lambda_3 > \ldots \geq 0$. When we run iterations in the algorithms which essentially increases r, at certain point we would see $\lambda_1^r >> \lambda_2^r >> \lambda_2^r >> \ldots \geq 0$. At the convergence of the algorithms, the group of largest entries in the relative accuracy vector $\mathbf{T}$, according to (4.3), would correspond to $\lambda_1^r \mathbf{e}_1(\mathbf{e}_1^\top \mathbf{1})$, or more precisely, the larger entries in the eigenvector $\mathbf{e}_1$ which essentially determines the direction of $\mathbf{e}_1$ in the n-dimensional space with each dimension representing a detector. Similarly, the second group of detectors identified by the relative accuracy would match up with the larger entries in the second eigenvector $\mathbf{e}_2$ of the similarity matrix $\mathbf{S}$. Just like an entry r_{jk} of the correlation matrix $\mathbf{R}$ in the PCA represents how similarly two variables X_j and X_k co-vary with each other, an entry s_{jk} of the similarity matrix $\mathbf{S}$ in the relative accuracy represents how similarly two detectors D_j and D_k classify files. Therefore, while the ordered eigenvectors of $\mathbf{R}$ in the PCA represents the ordered directions where the variables $\{X_1, \ldots, X_n\}$ co-varies the most, the ordered eigenvectors of $\mathbf{S}$ in the relative accuracy represents the ordered directions where the detectors $\{D_1, \ldots, D_n\}$ make similar decisions the most. This verifies that the grouping yielded by the relative magnitudes of entries in the vector $\mathbf{T}$ is indeed a meaningful one.

The Bellwether Detector

From the inception of the algorithm, the highest rated detector was specified to have a relative accuracy rating of 1. In contrast, the lowest rated detector was not pinned to a specific value,

except to keep it in the range of [0,1]. The intent was to create a relative ranking, not an absolute ranking, and the floating nature of the lower ranked detectors allowed for some interpretations of the data that was still valid while considering the lack of ground truth. With some ground truth, a sense of scale between the rankings would be viable. The Bellwether detector is proposed to allow a known entity without including any ground truth for the data being evaluated. The purpose of this is to provide a singular detector that has a known accuracy rating when no ground truth for the data exists.

In order to have a known accuracy with no known ground truth values for the data, a random classification from the available groupings can be fairly selected at random. Since there are two classes in this case, that means that the Bellwether detector rates each file with a $50/50$ chance as either benign or malicious. If the detector selects either a benign or malicious classification, it has an equal chance of being correct or incorrect. Since we have two classes in our classification, the Bellwether detector's accuracy is $1/2$. As the number of classes n increases, the Bellwether detector's accuracy lowers to $1/n$.

The Bellwether detector doesn't look at any of the relevant data, and blindly classifies each file. With enough data, the accuracy can be reliably calculated. This allows for an additional point of reference when scaling the data. This increases the accuracy of the relative accuracy vector, bringing the values closer to the true accuracy of the detectors.

In addition, this provides an indicator for when the number of poor detectors is on the threshold of overwhelming detectors that are functioning well. The Bellwether detector, in expected circumstances, should not be the most highly rated detector. If the Bellwether detector is the highest rated detector, this is an indication that the data is not trustworthy, and that the data needs a thorough examination and sanitization. This can be the case when there are many poor detectors.

Bellwether Scaling

The use of the Bellwether detector also eliminates the need for iterative refinement in the Relative Accuracy algorithm. Originally, iterations were utilized to incrementally approach a stable point,

improving the estimation a small step at each time. In practice, once the Relative Order of the detectors was set by the initial iteration, it did not change, and the additional iterations served to refine the increments between detectors.

The addition of the Bellwether detector eliminates this need, as the known value acts as a reference point and allows for the proper measure of accuracy after the first calculation. Working from Algorithm 4, our experiments show that the changes completely eliminate the need to include a loop, step size, error threshold, or exponential functions.

To utilize the Bellwether detector in this fashion, as shown in Algorithm 5, the Eigenvectors and Eigenvalues are used to calculate an initial Relative Accuracy vector. The values are then translated to move the Relative Accuracy for the Bellwether detector to the zero point on the axis. The remaining values are scaled according to the detector with the maximum absolute value. Due to our Bellwether detector having an expected accuracy of 50%, the other values are scaled between ± 0.5. Another translation now moves the Relative Accuracy vector so that the Bellwether detector sits at a value of 0.5, and the remainder of the detectors have a value between 0 and 1.

Algorithm 5 Computing relative accuracy using eigenvalues and a Bellwether detector

Input: Similarity matrix $\mathbf{S}_{n \times n}$; Output: Relative Accuracy vector $\mathbf{T} = [T_1, T_2, \ldots, T_n]^{\mathsf{T}}$

1: $\delta \leftarrow 2\varepsilon$
2: $\mathsf{c} \leftarrow 1$ (Initial Step)
3: $\mathsf{x} \in \mathbb{N}$ (Step Size)
4: $[\mathsf{U}, \mathsf{D}] \leftarrow eig(S)$
5: $\mathsf{T} \leftarrow U \times D \times U^T / max(U \times D \times U^T)$
6: $\mathsf{T} \leftarrow T - T_{Bellwether}$
7: $\mathsf{T} \leftarrow T / (2 \times max(abs(T))) + 0.5$
8: Return $T \leftarrow \mathsf{T}$

Our results show that utilizing the single iteration and scaling technique shown in Algorithm 5 provides results that differ from the results of utilizing Algorithm 4 with the Bellwether scaling by 0.1%. These results, functionally indistinguishable, and yield the same results for our experiments in regards to recovering metrics, while decreasing algorithmic complexity to be negligible.

Recovering Detector Metrics

Recall that our goal is to infer the standard metrics in the absence of ground-truth labels, namely TPR, TNR and Accuracy.

Algorithm 6 infers the detectors' metrics (TPR, TNR and $Accuracy$) by leveraging weighted voting and examining past decisions from the detectors. The basic idea of the algorithm is described as follows. Given scan results for n detectors over m files, namely V_{mn}, and a Relative Accuracy vector T_n with trust values for each detector, we will get a ratio of malicious and benign votes weighted by the degree we trust each detector. Utilizing a heuristic for determining the status of each file based on its scan results and the weights of these votes, we will then assign the file a label. That is, the heuristic is a simple majority rule with weights such that if the weight of malign votes is greater than half the weight of all the votes, that file is labeled as malicious, otherwise it's labeled as benign. Recall that a vote from detector D_j on file F_i is considered malicious if $V_{ij} = 1$, benign if $V_{ij} = 0$, and not counted if $V_{ij} = -1$ (i.e., D_j did not scan F_j).

It is worth mentioning that other heuristics are possible (e.g., weighting votes more or less depending on their statistical relationship to the mean and standard deviation). Nevertheless, the present simple method, as will be shown empirically, is adequate for the present problem.

Now that there exists relative accuracy scores for each detector D_i, the next step is to apply these scores in such a fashion that we can obtain real and useful information. These scores are used to weigh the vote that each detector has for each file it has examined, and give preference to those detectors with higher accuracy ratings. Using these values as file labels, the scores can be refined in an iterative fashion to obtain ground truth values for each file, with greater reliability than when using majority voting. These ground truth values are then used to calculate metrics for each detector.

After the votes are tallied, each file is labeled and the accuracy of the detectors is re-evaluated based on these labels and the actual votes from each detector. These new metric ratings will then be utilized to repeat the process until a stable point is reached, and these new labels will be considered ground truth.

Algorithm 6 Recovering Detector Metrics

Input: Scan Results $\mathbf{V}_{mn}$; Relative Accuracy vector $\mathbf{T} = [T_1, T_2, \ldots, T_n]^\mathsf{T}$; tolerable error threshold ε

Output:

Recovered Accuracy vector $\mathbf{Acc}' = [Acc_1, Acc_2, \ldots, Acc_n]^\mathsf{T}$;

Recovered TPR vector $\mathbf{TPR}' = [TPR_1, TPR_2, \ldots, TPR_n]^\mathsf{T}$;

Recovered TNR vector $\mathbf{TNR}' = [TNR_1, TNR_2, \ldots, TNR_n]^\mathsf{T}$;

$\delta \leftarrow 2\varepsilon$
$ACC' \leftarrow T$
while $\delta > \varepsilon$ **do**
  $\mathsf{MalignWeight} \leftarrow 0$
  $\mathsf{BenignWeight} \leftarrow 0$
  $TN, TP, FN, FP \leftarrow ([0, 0, \ldots, 0]_{1\times n})^\mathsf{T}$
  for i:=1 **to** m **do**
    for j:=1 **to** n **do**
      if $V_{ij} = 1$ **then**
        $\mathsf{MalignWeight} \leftarrow \mathsf{MalignWeight} + \mathsf{ACC}'_\mathsf{j}$
      else
        $\mathsf{BenignWeight} \leftarrow \mathsf{BenignWeight} + \mathsf{ACC}'_\mathsf{j}$
      end if
    end for
    $\mathsf{Malign} \leftarrow \mathsf{MalignWeight} > \frac{\mathsf{MalignWeight}+\mathsf{BenignWeight}}{2}$
    for j:=1 **to** n **do**
      if $\mathsf{Malign} =$ **false then**
        if $V_{ij} = 1$ **then**
          $FP_i \leftarrow FP_i + 1$
        else if $V_{ij} = 0$ **then**
          $TN_i \leftarrow TN_i + 1$
        end if
      else if $\mathsf{Malign} =$ **true then**
        if $V_{ij} = 1$ **then**
          $TP_i \leftarrow TP_i + 1$
        else if $V_{ij} = 0$ **then**
          $FN_i \leftarrow FN_i + 1$
        end if
      end if
    end for
  end for
  $\delta \leftarrow \sum_1^m |ACC'_i - \frac{TN_i+TP_i}{TN_i+TP_i+FN_i+FP_i}|$
  $Acc' \leftarrow \frac{TN+TP}{TN+TP+FN+FP}$
  $TPR' \leftarrow \frac{TP}{TP+FN}$
  $TNR' \leftarrow \frac{TN}{TN+FP}$
end while
Return $Acc'; TPR'; TNR'$

Additionally, due to the voting scheme, a value for certainty can be obtained for each file, so that edge cases can be identified. This will greatly reduce the amount of hand tuning performed by human agents, and will provide useful metrics for examining the dataset.

4.3 Experiments and Results

In this section, we first conduct experiments to validate the methodology via synthetic data with known ground truth. Then, we apply the methodology to analyze a real-world dataset of malware detectors. In order to make the experimental results comparable to the study reported in [2], we adopt the same datasets in the present study.

4.3.1 Validation Experiments with Synthetic Data

Generating synthetic data. The synthetic data is adopted from [2]. Specifically, the three synthetic datasets, denoted by **D1**-**D3**, are summarized as follows (with each containing one million samples):

- **D1** contains 300,000 malicious files and 700,000 benign files;
- **D2** contains 500,000 malicious files and 500,000 benign files; and
- **D3** contains 700,000 malicious files and 300,000 benign files.

These three datasets have varying malicious:benign ratios, which allows us to investigate the impact of the ratio on the effectiveness of the methodology.

Synthetic experiment setup. We consider 10 experiments. Each experiment uses a (different) number of detectors that are characterized by their true-positive rate (TPR) and a true-negative rate (TNR), while false-positive (FPR) and false-negative(FNR) rates are derived as $FPR = 1 - TNR$ and $FNR = 1 - TPR$ [43]. These values are known in the synthetic experiment.For each detector, we generate its label on a file in **D1**-**D3** according to the detector's FPR and FNR, as follows.

If the ground truth label of the file is malicious, then the detector labels it as 1 (malicious) with probability $1 - FNR$ and labels it as 0 (benign) with probability FNR; If the ground truth label

of the file is benign, then the detector labels it as 0 (benign) with probability $1 - FPR$ and labels it as 1 (malicious) with probability FPR. Given the synthetic labels, the accuracy of a detector can be defined as $ACC^* = \frac{TP^*+TN^*}{TP^*+TN^*+FP^*+FN^*}$, where TP^* is the synthetic true-positive labels (i.e., the ground truth of a file is malicious and the file is also labeled as malicious by the detector in question), TN^* is the synthetic true-negative labels, FP^* is the synthetic false-positive labels, and FN^* is the synthetic false-negative labels. We define the true positive rate as $TPR^* = \frac{TP}{TP+FN}$ and the true negative rate as $TNR^* = \frac{TN}{TN+FP}$.

We use the synthetic datasets **D1**-**D3** to derive the detectors' relative accuracy and then use their relative accuracy to derive their absolute inferred metrics (i.e., accuracy ACC', TNR', and TPR'), which is then compared with the ground truth values (ACC^*, TPR^*, and TNR^*) to determine the effectiveness of our algorithms.

Experiments 1-5 aim to test a variety of situations with various distributions of accuracy in malware detectors.

- **Experiment 1**: Four sets of detectors (50 in total) with varying ground truth accuracy rates are simulated as:
  - 10 detectors with an accuracy range of 95% to 85% (i.e., each detector has an accuracy that is randomly and independently chosen from interval $[85\%, 95\%]$ and then used to generate labels);
  - 10 detectors with an accuracy range of 85% to 75%;
  - 10 detectors with an accuracy range of 80% to 70%;
  - 20 detectors with an accuracy range of 75% to 65%.
- **Experiment 2**: Four sets of detectors (50 in total) with varying ground truth accuracy rates are simulated as:
  - 10 detectors with an accuracy range of 100% to 90%;
  - 10 detectors with an accuracy range of 95% to 85%;

  - 10 detectors with an accuracy range of 90% to 80%;
  - 20 detectors with an accuracy range of 85% to 75%.

- **Experiment 3**: One set of 50 detectors with each detector having a ground truth accuracy rate that is randomly and independently chosen from interval $[90\%, 100\%]$.

- **Experiment 4**: Five sets of detectors (90 in total) with varying ground truth accuracy rates (including some *poor* detectors) are simulated as:
  - 50 detectors with an accuracy range of 100% to 90%;
  - 10 detectors with an accuracy range of 95% to 85%;
  - 10 detectors with an accuracy range of 90% to 80%;
  - 10 detectors with an accuracy range of 85% to 75%.
  - 10 detectors with an accuracy range of 45% to 35%.

- **Experiment 5**: Five sets of detectors (50 in total) with varying ground truth accuracy rates (including some *poor* detectors) are simulated as:
  - 10 detectors with an accuracy range of 100% to 90%;
  - 10 detectors with an accuracy range of 95% to 85%;
  - 10 detectors with an accuracy range of 90% to 80%;
  - 10 detectors with an accuracy range of 85% to 75%;
  - 10 detectors with an accuracy range of 45% to 35%.

Experiments 6-10 are conducted to investigate the threshold such that Algorithm 4 can distinguish *good* detectors from *poor* detectors as the ratio of the good detectors to the poor detectors decreases. In each of these 5 experiments, the good detectors have accuracies that range from 100% to 90%, while the poor detectors have accuracies that range from 45% to 35%.

- **Experiment 6**: Two sets of detectors (50 in total) with varying ground truth accuracy rates (including 10 or 20% *poor* detectors) are simulated as:

  - 40 detectors with an accuracy range of 100% to 90%;
  - 10 detectors with an accuracy range of 45% to 35%.

- **Experiment 7**: Two sets of detectors (50 in total) with varying ground truth accuracy rates (including 20 or 40% *poor* detectors) are simulated as:
  - 30 detectors with an accuracy range of 100% to 90%;
  - 20 detectors with an accuracy range of 45% to 35%.

- **Experiment 8**: Two sets of detectors (50 in total) with varying ground truth accuracy rates (including 25 or 50% *poor* detectors) are simulated as:
  - 25 detectors with an accuracy range of 100% to 90%;
  - 25 detectors with an accuracy range of 45% to 35%.

- **Experiment 9**: Two sets of detectors (50 in total) with varying ground truth accuracy rates (including 30 or 60% *poor* detectors) are simulated as:
  - 20 detectors with an accuracy range of 100% to 90%;
  - 30 detectors with an accuracy range of 45% to 35%.

- **Experiment 10**: Two sets of detectors (50 in total) with varying ground truth accuracy rates (including 40 or 80% *poor* detectors) are simulated as:
  - 10 detectors with an accuracy range of 100% to 90%;
  - 40 detectors with an accuracy range of 45% to 35%.

4.3.2 Experimental Results

For each experiment, there are four sets of values: the *Ground Truth* values (including the ground truth labels, the ground truth true positive rates, the ground truth true negative rates, and the ground truth accuracy rates), the *Relative Accuracy* values, the *Bellwether Accuracy* values, and finally, the

Inferred Metric values (including the inferred labels, the inferred true positive rates, the inferred true negative rates, and the inferred accuracy rates).

Experimental Results with D1

Figure 4.1 plots the experimental results with dataset **D1**. Recall that the Ground Truth Accuracy is the accuracy value that was assigned to each detector when generating the synthetic data. It is used to gauge the effectiveness of Algorithm 4 by comparing its inferred detector accuracy to the ground truth accuracy.

We stress that the relative accuracy of a detector is *not* the same as the Ground Truth accuracy of the detector; rather, it can deviate significantly. Nevertheless, both the *Relative Accuracy* curve and the *Bellwether Accuracy* curve provide data that helps describe the *True Accuracy* curve, with differences in the slope and the degree of change. This indicates that recovering the true accuracy of each detector may be be possible, which will be confirmed later.

Insight 3. *The Bellwether detector can provide an indicator of when there might be a problem with the underlying data (i.e., too many poor detectors), especially when the Bellwether detector is rated as having the highest Relative Accuracy. A detector rated lower than the Bellwether detector indicates that it has less chance of classifying a file than randomly assigning a classification value.*

Second, Experiments 6-9 show that with similar ranges of ground truth accuracy, but varying numbers of *good* vs. *poor* detectors, the difference between the true accuracy and the relative accuracy increases as the number of poor detectors increases. This means that the Bellwether detector can reduce the error in the relative accuracy in all our cases.

Insight 4. *Algorithm 4 reduces the calculation from tens-of-minutes of computation to seconds, while being mathematically equivalent to Algorithm 2.*

Third, from the standpoint of Bellwether Accuracy, we observe that this detector has a known accuracy value based on an even chance of random decisions and thus can be used to scale the accuracy ratings from the base Relative Accuracy algorithm. In all cases, we observe that the inclusion of this detector significantly increases the accuracy of the detector rankings.

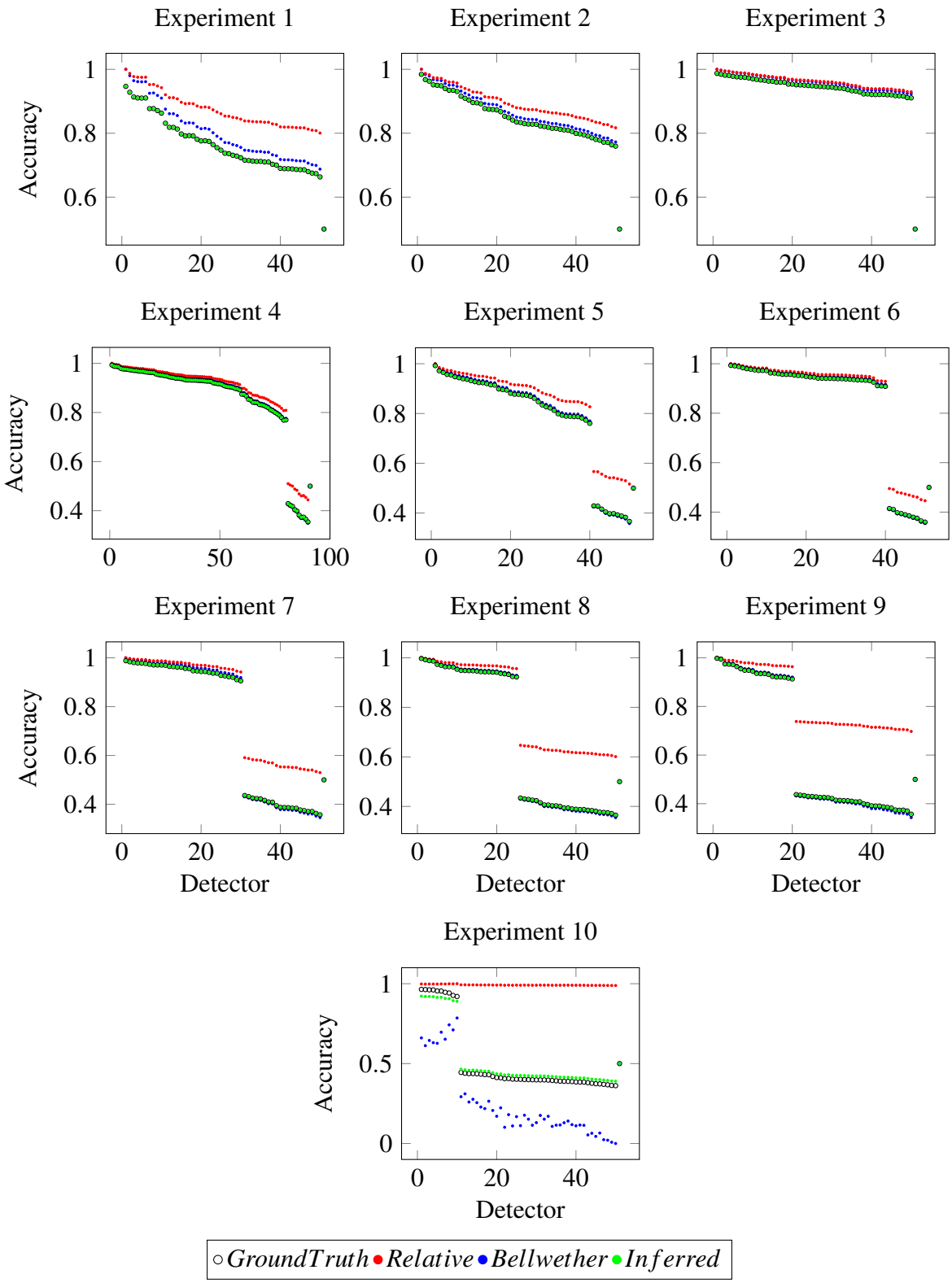


Figure 4.1: Results for Experiments **1-10** with dataset **D1**, showing the Ground Truth, Relative Accuracy, Relative Accuracy w/ Bellwether detector and the Inferred Accuracy values (the *y*-axis) of the detectors (the *x*-axis).

Insight 5. *The Bellwether detector substantially reduces the difference between the Ground Truth Accuracy and the Relative Accuracy, while allowing to detect when the number of poor detectors is approaching the algorithm's limit.*

Fourth, from the standpoint of the Inferred Metrics obtained by using the Bellwether Accuracy as a starting point for Algorithm 6 in Section 4.2.2, we observe that in most cases, this refinement was simple due to the low error rate from Bellwether Accuracy calculations. The exception was in Experiment 10, where the process took longer to finally resolve the values. Even in the case of Experiment 10 though, Figure 4.2 shows that the final Inferred Metric value results are very tightly coupled to the Ground Truth Accuracy values.

In addition, the Bellwether Accuracy values indicate an underlying uncertainty with the base data, and are a signal that further analysis should be considered.

Insight 6. *By using the weights obtained from the Relative Accuracy algorithm with the use of the Bellwether detector, the overall accuracy of the detectors was much more accurately predicted, allowing for an accurate recovery of the unknown metrics belonging to the detectors.*

Furthermore, since there now exist good Ground Truth labels for the data files, it is possible to calculate the true-positive rate (TPR), true-negative rate (TNR), false-positive rates (FPR) and false-negative rates (FNR) for individual detectors. Experiments 1-9 show excellent results, with the highest difference between the Ground Truth Accuracy and the Recovered Accuracy being 2×10^{-6}. Experiment 10 shows a higher error, although the maximum is 4.0%, with the average being 1.2%. The separation between *good and* poor detectors is still easily distinguished, and a removal of poor detectors would show an overall improvement in results.

Insight 7. *By recovering good ground-truth values for our data, it is possible to accurately characterize the TPR and TNR metrics. This additional data gives us greater insight into the factors making up the overall Accuracy of the detectors, and allows us to identify the shortcomings of poorly performing detectors and gives us an option to remove them from the candidate pool.*

Experimental Results with Datasets D2 and D3

The experimental results are almost identical to **D1**. The changes in the file distribution changed the initial ordering, but the results obtained by comparing the true accuracy, TPR, and TNR to the inferred accuracy, TPR, and TNR are straightforward with respect to **D1**, so we don't include them here.

Algorithm 4 provides useful results whether the (true) accuracy of detectors is continuously distributed across a wide range (Experiments 1-2), distributed across a narrow range (Experiment 3), or distributed across a wide range in a discontinuous fashion (Experiments 4-5). Experiments 6-10 show that through all three datasets, the Relative Accuracy provides reliable results up to the 4:1 ratio of poor vs. good detectors in experiment 10. At this threshold, the poor detectors begin to be rated above the good detectors due to the noise introduced by sheer numbers. Despite this noise, the inclusion of the Bellwether Detector detects the inconsistency and allows for the recovery of metrics utilizing Algorithm 6.

Insight 8. *The Bellwether Detector leads to more accurate computation of the relative accuracy and can detect when the number of poor detectors is approaching Algorithm 4's limit.*

4.3.3 Applying the Methodology to Evaluate a Real Dataset

The real-world dataset contains $m \approx 10.7$ million files collected from VirusTotal, each of which was scanned by up to $n = 62$ anti-malware detectors. However, not every detector scanned every file. Each detector labels a file it scanned as malicious (1) or benign (0). The dataset is transformed to matrix $(\mathbf{V}_{ij})_{n\times m}$, from which we derive the similarity matrix $\mathbf{S}$ and then the relative accuracy vector $\mathbf{T}$ according to Algorithm 3, both with and without considering the Bellwether Detector.

Figure 4.3 plots the Relative Accuracy, the Bellwether Accuracy, and the Inferred Metrics of the 62 detectors, in descending order relative to the Inferred Metrics Values. We observe that 4 detectors were rated near 0 in terms of both the Relative, Bellwether and Inferred Accuracy values. We also observe that among these 4 detectors, the most prolific scanned a total of 52 files, with the

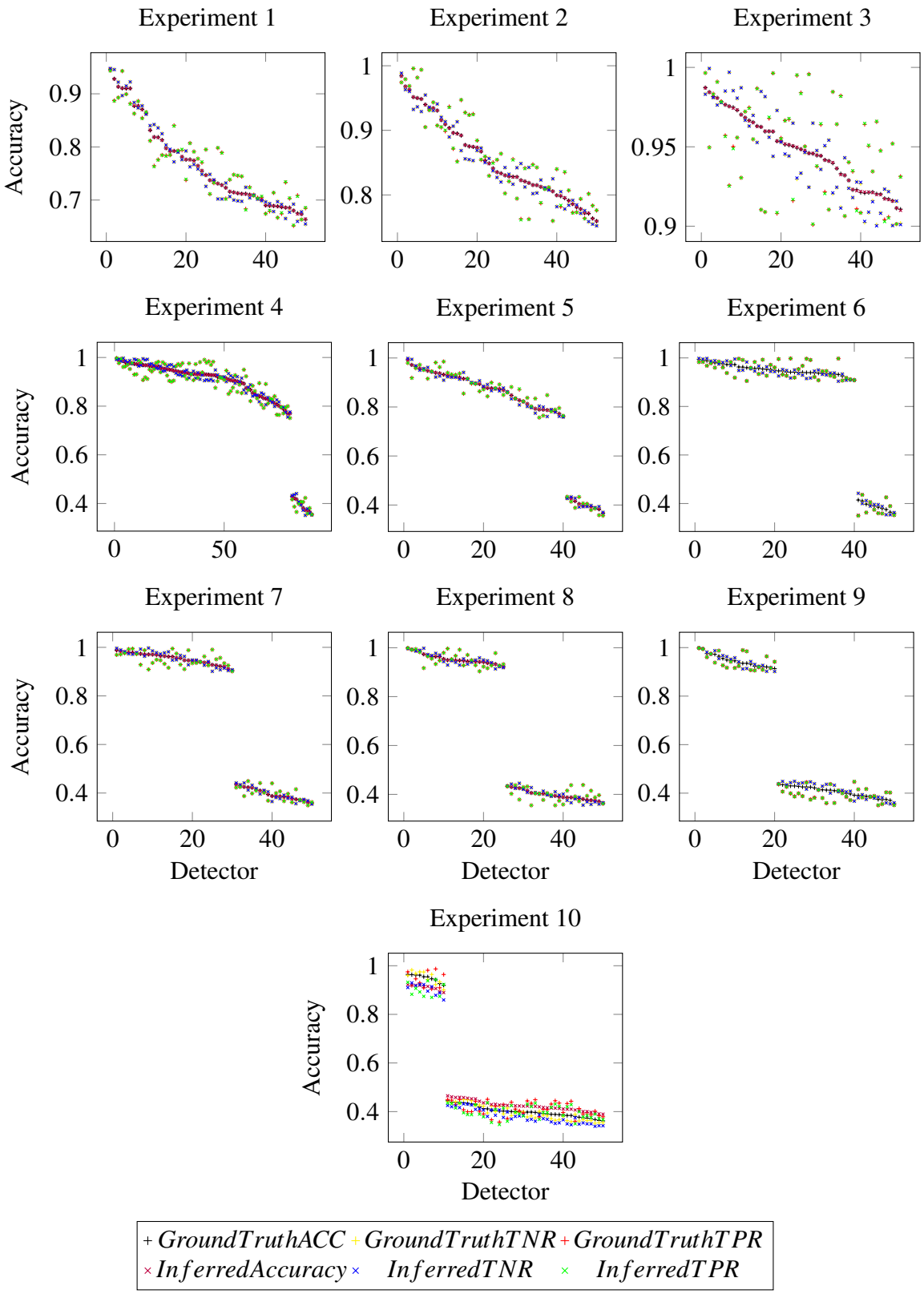


Figure 4.2: Results for Experiments **1-10** with dataset **D1**, showing the Ground Truth TNR, Ground Truth TPR, Inferred TNR, Inferred TPR values (the *y*-axis) of the detectors (the *x*-axis).

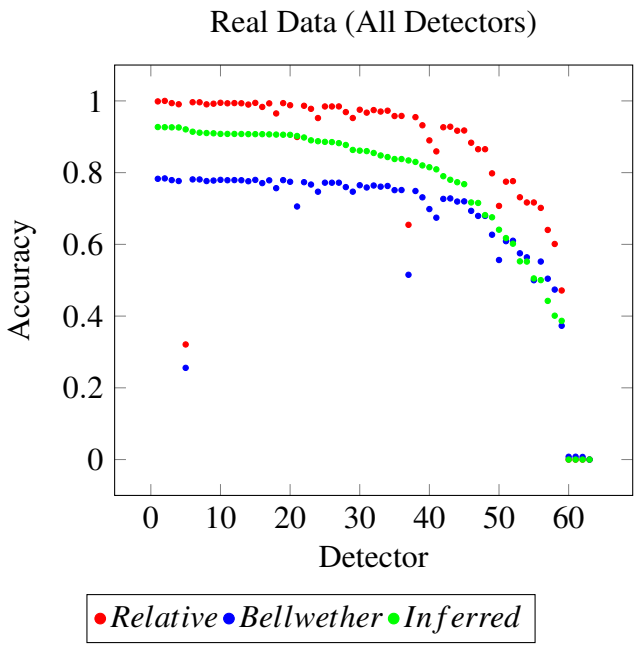


Figure 4.3: Plots of the Relative Accuracy, Relative Accuracy w/ Bellwether detector, and the Inferred Accuracy of the 62 real-world detectors.

others scanning 51, 37, and 1 file.

Two other detectors stand out in Figure 4.3. One had a Relative Accuracy of 0.32 and the other 0.65. These detectors ended up with Inferred Metric values of .90 and .80. Both of these detectors also scanned a relatively low number of files, 31% and 67% of the total, respectively. Comparatively, the majority of detectors scanned more than 90%. This indicates that a large difference in the number of files that are scanned by different detectors can cause lower ratings in Relative and Bellwether Accuracy values.

Figure 4.4 maps the inferred TPR, TNR, and Accuracy. As one would expect, the highest rated detectors have high TPR and TNR scores. Unexpectedly, though, the majority of the lower ranking detectors also have high ratings for their TPR scores. It becomes obvious that most of the lower performing detectors suffer from low TNR ratings. This shows that that the primary reduction to their overall accuracy rating comes from incorrectly classifying benign files as malicious.

Further exception can also be made for the absolute lowest ranked detectors, whose TPR scores are all at 0, showing that they labelled all files as benign whether they were malicious or not. This is objectively worse for malware detectors than having a low TNR. Allowing malicious files through

as benign (i.e., high FNR) is much more harmful than labelling benign files as malicious. These detectors should thus be considered worse than poor, and considered harmful to any system that relies on them.

Insight 9. *The algorithm for inferring detection metrics can explain why some detectors are rated poorly.*

Insight 9 says that there is a means to detect, therefore possibly eliminate, poor-performing detectors.

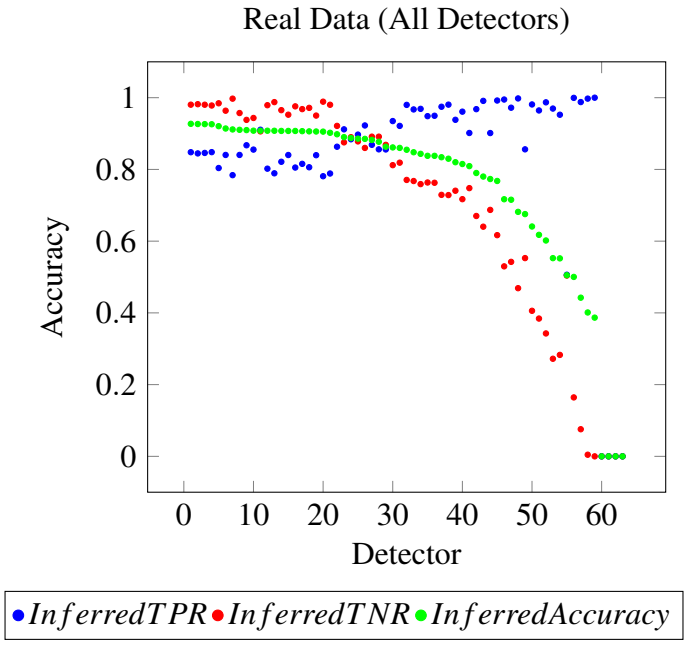


Figure 4.4: Plots of the Inferred TPR and the Inferred TNR of the 62 real-world detectors.

Summarizing the preceding discussion, we obtain the following insights:

Insight 10. *A few detectors in the real-world dataset are not useful and actually provide nothing but noise. The Relative and Bellwether Accuracy measures can detect the presence of such detectors. Additionally, recovering the security metrics for the detectors can also indicate when detectors are more than just useless and are harmful in their classifications.*

4.4 Chapter Summary

In this work, we refined the calculation for *relative accuracy* of malware detectors in the absence of ground truth and presented an improved algorithm to derive the relative accuracy. The relation of relative accuracy to the principal component analysis method was explored and analyzed. The idea of the Bellwether detector was introduced and used to increase the accuracy of our calculations. Finally, it was shown how those refined values can be used to recover the metrics of malware detectors. We then used these same methods to evaluate the real world dataset from VirusTotal, and rank the detectors.

CHAPTER 5: LEVERAGING INFERRED METRICS AND THE SIMILARITY MATRIX TO SELECT HETEROGENEOUS DETECTORS

5.1 Chapter Introduction

Having investigated the problem of recovering cyber security metrics for malware detectors with binary classification results (i.e., whether a file is malicious or benign), this chapter investigates how to cope with multiclass classification, where malware detectors may predict a file into specific classes. In this context, it is also important to know which malware detectors are more accurate than others. Specifically, we investigate two problems: Can the methods introduced in the previous chapters be extended to tackle the more general problem we aim to address in the present chapter? How can these methods be applied to not just provide cyber security metrics, but provide cyber security researchers a method selecting detectors?

Chapter contributions. In this chapter, we make five contributions. First, we extend the algorithms presented in the previous chapters to cope with multiclass classification. In order to accomplish this, we consider a new dimension of the problem, leading to the extension to the notions of similarity matrix and recovered accuracy measurements.

Second, we show that accurate measurements across different file types allow for a simple calculation that creates a file identification classification for new files that are scanned. If a detector scans a file and detects it as class o, it assigns a detected label to that specific detection vector; otherwise, it assigns a label of not found. This provides a set of decision vectors per detector per file type. These decision vectors can then be combined with the recovered metrics for all detectors for a given file type in order to provide information on the classification of the files scanned.

Third, we show that using the calculated absolute metrics allows for the selection of a subset of good detectors while excluding detectors that are rated poorly. This set of good detectors provides an improvement over simply weighting the votes from all of the detectors. By eliminating the detectors that are rated poorly, we construct a set of good detectors.

Fourth, we show how the metrics derived from the detectors, when combined with the similarity matrix, allow for the selection of heterogeneous good detectors. That is, they allow for the selection of a subset of dissimilar, yet accurate, detectors to provide good coverage over scanned files. The comparison between these limited detectors and the results from the full set of detectors using unweighted voting is examined.

Fifth, we evaluate the preceding contributions by conducting experiments using synthetic and real world data. For the synthetic data, we know the ground truth in terms of (i) which file type each scanned file is and (ii) the malware detectors' absolute accuracy in regards to detection of each file type. Experimental results using synthetic data show that each malware detector's absolute accuracy metrics can be inferred with minimal error across the various file types when compared with the known ground truth. This shows that the similarity matrix can be used to select heterogeneous detectors even in the absence of known file types. We compare this with the results of using the standard majority voting method with all available detectors We analyze the real world data and make a selection of 10 out of the available 62 real world antivirus detectors and show that the combined accuracy of these detectors is greater than any of them individually, and much greater than the majority vote accuracy.

Chapter outline. The rest of the chapter is organized as follows. Section 5.2 presents the problem statement and our methodology. Section 5.3 describes our experiments and results. Section 5.4 summarizes the present chapter.

5.2 Problem Statement and Methodology

5.2.1 Accommodating Multiple File Types

Consider a set of m files, which are denoted by $F_1, \ldots, F_m$, n malware detectors, which are denoted by $D_1, \ldots, D_n$, and o file types, which are denoted by $E_1, \ldots, E_o$. The input is a matrix

$V = (V_{ijk})_{1 \leq i \leq n, 1 \leq j \leq m, 1 \leq k \leq o}$, where

$$V_{ijk} = \begin{cases} 1 & \text{if } D_i \text{ detects } F_j \text{ as } E_k, \\ 0 & \text{if } D_i \text{ detects } F_j \text{ as not } E_k, \\ -1 & \text{if } D_i \text{ did not scan } F_j \text{ (for any file type } E_k). \end{cases}$$

Note that for a fixed i ($1 \leq i \leq n$), the 2-dimensional matrix $Vi = (V_{ijk})_{1 \leq j \leq m, 1 \leq k \leq o}$ represents the outcome of using detector D_i to label the m files with respect to the o file types.

Given this input, the research problem is to infer, *in the absence of ground truth*, the following standard malware detection metrics [43] for each detector in regards to each file type.

In order to achieve this goal, we adapt the approach that was refined to recovery binary metrics as shown in Chapter 4. Recall that the relative Accuracy calculated using an iterative algorithm has been previously explored [2] and shown to be effective at ranking malware detectors (Chapter 3). It was then shown that absolute metrics could be recovered [3] in regards to those detectors (Chapter 3). In what follows we show how the Similarity Matrix, Relative Accuracy and Recovery Metrics algorithms can be adapted to recover information regarding multiple file types and be used to allow quick classification. Further, we show how the Similarity Matrix can be used to select an effective heterogeneous subset of detectors, and why this is better than the default majority voting scheme when evaluating the ground truth classification of potential malware file types.

5.2.2 Methodology

Multidimensional Similarity Matrix

Now we extend the definitions presented in Chapter 3 to incorporate one additional dimension, which allows us to account for file types in the Similarity Matrix.

Definition 10 (similarity matrix; extended from [2])**.** *Denote by $D = \{D_1, \ldots, D_n\}$ a universe of n detectors. Denote by $F = \{F_1, \ldots, F_m\}$ a universe of m files. Denote by $E = \{E_1, \ldots, E_o\}$ a universe*

of o file types. Let matrix $V = (V_{ijk})_{1 \le i \le n, 1 \le j \le m, 1 \le k \le o}$ *be defined as*

$$V_{ijk} = \begin{cases} 1 & \text{if } D_i \text{ detects } F_j \text{ as being file type } F_k, \\ 0 & \text{if } D_i \text{ detects } F_j \text{ as not being file type } F_k, \\ -1 & \text{if } D_i \text{ did not scan } F_j. \end{cases}$$

Each vector $V_{io} = (V_{i1o}, \ldots, V_{imo})$*, where* $1 \le i \le n$*, is detector* D_i*'s labels on the m files with respect to file type o.*

Given two detectors, D_i and D_k, where $1 \le i, k \le n$ and $i \ne k$, the following definitions describe the similarity between the two detectors in terms of how they label files.

Definition 11 (agreement matrix; extended from [2])**.** *The extended agreement matrix counts the number of files that are labeled by detectors* D_i *and* D_k *consistently with respect to a file type, namely*

$$A_{iko} = A_{kio} = \sum_{\ell=1}^{m} \begin{cases} 1 & \text{if } V_{i\ell o} = V_{k\ell o} \wedge V_{i\ell o} \ne -1 \wedge V_{k\ell o} \ne -1 \\ 0 & \text{if } V_{i\ell o} \ne V_{k\ell o} \vee V_{i\ell o} = -1 \vee V_{k\ell o} = -1. \end{cases}$$

Definition 12 (count matrix; extended from [2])**.** *The count matrix describes the number of files that have been labelled by two detectors* D_i *and* D_k*, regardless of whether their file type labels are consistent or not. That is,*

$$C_{iko} = C_{kio} = \sum_{\ell=1}^{m} \begin{cases} 1 & \text{if } V_{i\ell o} \ne -1 \wedge V_{k\ell o} \ne -1, \\ 0 & \text{if } V_{i\ell o} = -1 \vee V_{k\ell o} = -1. \end{cases}$$

Definition 13 (similarity between detectors)**.** *The notion of similarity between two detectors* D_i *and* D_k *measures their degree of consistency in labeling files in terms of their types.*

Definition 13 is a general metric that can be instantiated in multiple ways to quantify the consistency between two detectors. It is general in the sense that it can be adapted to accommodate

various kinds of information relevant to the detectors, such as: the file types that are detected by the detectors, the labels that are relevant (e.g., binary vs. multiclass classification). Although Definition 13 can be instantiated in many ways, any meaningful instantiation should satisfy the following requirement:

Definition 14 (similarity requirements; extended from [2])**.** *A quantitative measure of similarity, S_{ikj} between two detectors D_{ij} and D_{kj} where $1 \leq i,k \leq n, 1 \leq j \leq o$, should satisfy two natural requirements:*

- $S_{ikj} = S_{kij}$*, because similarity should be symmetric with respect to any file type; and*
- $S_{iij} = 1$ *for any* $1 \leq i \leq n, 1 \leq j \leq o$ *to reflect self-similarity.*

A particular instantiation of Definition 13 was proposed in [2] and presented below.

Definition 15 (one instantiation of general similarity; extended from [2])**.** *This similarity is defined as the ratio of the number of labels where D_{ij} and D_{kj} agree, A_{ikj}, over the total number of files that are processed or scanned by both, C_{ikj}, namely where A_{ikj} and C_{ikj} are elements of the matrices $A_{n\times n\times o}$ and $C_{n\times n\times o}$ respectively.*

$$S_{ikj} = \frac{A_{ikj}}{C_{ikj}}$$

By considering all of the n detectors, Definition 15 naturally leads to the following notion of the *multidimensional similarity matrix*:

Definition 16 (multidimensional similarity matrix; extended from [2])**.** *Given n detectors and m files that are scanned as with respect to o possible file types, the multidimensional similarity matrix describes the consistency between the labeling behavior of the detectors and is denoted by $\mathbf{S} = (S_{ikj})_{1\leq i,k\leq n, 1\leq j\leq o}$, where element S_{ikj} is given by Definition 15.*

Multidimensional Bellwether Algorithm

Now we show how to extend the Bellwether detector [3] to accommodate the multidimensional Similarity Matrix. In order to utilize the Bellwether detector in this fashion, as shown in Algorithm 7, we use eigenvectors and eigenvalues to calculate an initial Relative Accuracy vector for each file type. The values are then translated and scaled according to the detector with the maximum absolute value. Since the Bellwether detector has an expected accuracy of 50% per file scanned per file type, the other values are scaled between ± 0.5. The values are translated again such that Bellwether detector sits at a value of 0.5, and the rest of the detectors are normalized to have a value between 0 and 1.

Algorithm 7 Computing relative accuracy using eigenvalues and a Bellwether detector

Input: Similarity matrix $\mathbf{S}_{n \times n \times o}$;
Output: Relative Accuracy vector $\mathbf{T_o} = [T_{1o}, T_{2o}, \ldots, T_{no}]^{\mathsf{T}}$

1: $\delta \leftarrow 2\varepsilon$
2: **for** i:=1 to o **do**
3: $\quad \mathsf{U} \leftarrow eigenvectors(S_i)$
4: $\quad \mathsf{D} \leftarrow eigenvalues(S_i)$
5: $\quad \mathsf{T} \leftarrow U \times D \times U^T / \max(U \times D \times U^T)$
6: $\quad \mathsf{T} \leftarrow T - T_{Bellwether}$
7: $\quad \mathsf{T} \leftarrow T/(2 \times \max(abs(T))) + 0.5$
8: $\quad T_i \leftarrow \mathsf{T}$
9: **end for**
10: Return T

Recovering Detector Metrics

Our goal is to infer the standard metrics in the absence of ground-truth labels, namely TPR, TNR and Accuracy per file type.

Algorithm 8 infers detectors' metrics (TPR, TNR and $Accuracy$) by leveraging weighted voting and examining past decisions of the detectors. The basic idea of the algorithm is described as follows. Given scan results for n detectors over m files and o file types, namely $V_{m \times n \times o}$, and a set of Relative Accuracy vectors $T_{n \times o}$ with trust values for each detector over each file type, we get a ratio of detected file type votes, weighted by the degree we trust each detector for each file

Algorithm 8 Recovering Detector Metrics

Input: Scan Results $\mathbf{V}_{mn}$; Relative Accuracy vector $\mathbf{T_{n\times o}} = [T_{11}, T_{21}, \ldots, T_{n1}]^{\mathsf{T}}$, $[T_{12}, T_{22}, \ldots, T_{n2}]^{\mathsf{T}}$, $\ldots$, $[T_{1o}, T_{2o}, \ldots, T_{n1o]}{}^{\mathsf{T}}$; tolerable error threshold ε

Output: Recovered Accuracy vector $\mathbf{Acc'_{n\times o}} = [Acc_{11}, Acc_{21}, \ldots, Acc_{n1}]^{\mathsf{T}}$, $[Acc_{12}, Acc_{22}, \ldots, Acc_{n2}]^{\mathsf{T}}$, $\ldots$, $[Acc_{1o}, Acc_{2o}, \ldots, Acc_{no}]^{\mathsf{T}}$; Recovered TPR vector $\mathbf{TPR'_{n\times o}} = [TPR_{11}, TPR_{21}, \ldots, TPR_{n1}]^{\mathsf{T}}$, $[TPR_{12}, TPR_{22}, \ldots, TPR_{n2}]^{\mathsf{T}}$, $\ldots$, $[TPR_{1o}, TPR_{2o}, \ldots, TPR_{no}]^{\mathsf{T}}$; Recovered TNR vector $\mathbf{TNR'_{n\times o}} = [TNR_{11}, TNR_{21}, \ldots, TNR_{n1}]^{\mathsf{T}}$, $[TNR_{12}, TNR_{22}, \ldots, TNR_{n2}]^{\mathsf{T}}$, $\ldots$, $[TNR_{1o}, TNR_{2o}, \ldots, TNR_{no}]^{\mathsf{T}}$;

```
δ ← 2ε
ACC' ← T
while δ > ε do
    TypeWeight ← 0
    NotTypeWeight ← 0
    TN, TP, FN, FP ← ([0, 0, ..., 0]_{1×n})^T
    for k:=1 to o do
        for i:=1 to m do
            for j:=1 to n do
                if V_ijk = 1 then
                    TypeWeight ← TypeWeight + ACC'_jk
                else
                    NotTypeWeight ← NotTypeWeight + ACC'_jk
                end if
            end for
            Type ← TypeWeight > (TypeWeight + NotTypeWeight)/2
            for j:=1 to n do
                if Type = false then
                    if V_ijk = 1 then
                        FP_ik ← FP_ik + 1
                    else if V_ijk = 0 then
                        TN_ik ← TN_ik + 1
                    end if
                else if Type = true then
                    if V_ijk = 1 then
                        TP_ik ← TP_ik + 1
                    else if V_ijk = 0 then
                        FN_ik ← FN_ik + 1
                    end if
                end if
            end for
        end for
        δ ← Σ_1^m |ACC'_ik − (TN_ik + TP_ik)/(TN_ik + TP_ik + FN_ik + FP_ik)|;  Acc'_o ← (TN + TP)/(TN + TP + FN + FP);  TPR'_o ← TP/(TP + FN);
        TNR'_o ← TN/(TN + FP)
    end for
end while
Return Acc'; TPR'; TNR'
```

type. Utilizing a heuristic for determining the status of each file based on its scan results and the weights of these votes, we assign the file a label. That is, the heuristic is a simple majority rule with weights such that if the weight of votes in regards to a specific file type is greater than half the weight of all the votes, the file is labeled as as that file type; otherwise, it is labeled as not that file type. Recall that a vote from detector D_j on file F_i is considered type E_k if $V_{ijk} = 1$, not type E_k if $V_{ijk} = 0$, and not counted if $V_{ijk} = -1$ (i.e., D_j did not scan F_j as type E_o at all). We note that while other heuristics are possible, the present method is adequate for the current problem.

Now that there exist relative accuracy scores for each detector D_i for each file type E_k, the next step is to apply these scores in such a fashion that we can obtain real and useful information. These scores are used to weigh the vote that each detector has for each file it has examined, and give preference to those detectors with higher accuracy ratings. Using these values as file labels, the scores can be refined in an iterative fashion to obtain ground truth values for each file type, with greater reliability than when using majority voting. These ground truth values are then used to calculate metrics for each detector.

After the votes are tallied, each file has a labeled file type, the accuracy of the detectors is re-evaluated based on these labels and the actual votes from each detector. New metric ratings will then be used to iterate through the process until stability is reached. These labels will be considered recovered ground truth.

Additionally, due to the voting scheme, a value for certainty can be obtained for each file, so that cases without a clear identification can be flagged. This will greatly reduce the amount of hand tuning performed by human agents, and will provide useful metrics for examining the dataset.

File Identification

For each file scanned by the detectors, a decision vector is created that records the findings of each detector. This vector can then be used as a binary "file identifier" for the file, where each positive detection is labelled as a one, and each negative detection is labelled as zero.

Definition 17 (file identifier vector)**.** *For each file, let F be the matrix $F_{n\times o} = [f_1, f_2, \ldots, f_j]$, $0 \leq j \leq o$, where each column vector $f_{n\times 1}$ represents the decision vector for each given detector i, $0 \leq i \leq n$, scanning the file and assigning a binary label with respect to file type o. We define*

$$F_{ij} = \begin{cases} 1 & \textit{if detector } i \textit{ detects the file as type } j \\ 0 & \textit{if detector } i \textit{ does not detects the file as type } j \end{cases}$$

Let the vector Acc_i be the Inferred Accuracy measurement for File Type o for each of the n detectors.

Definition 18 (identification map)**.** *For each file, let M be the matrix $M_{n\times o} = [m_1, m_2, \ldots, m_j]$, $0 \leq j \leq o$, where each column vector $m_{n\times 1}$ represents an* identification map *defined in relation to the Inferred Accuracy measurement for each file type, where*

$$M_{ij,0\leq i\leq n,0\leq j\leq o} = \begin{cases} 1 & \textit{if } Acc_{ij} > .5 \\ 0 & \textit{if } Acc_{ij} \leq .5 \end{cases}$$

The file identification and the identification map can then be used in conjunction to produce a file type equality score E per each of the o file types.

Definition 19 (Equality Score)**.** *The Equality Score is a numerical representation of the measure of agreement between a file's file identifier matrix F and the identity map M matrix. Let e be the vector $e_{1\times o} = [e_1, e_2, \ldots, e_j]$, $0 \leq j \leq o$ where each entry is a measure of agreement between F and M defined as:*

An equality score *vector e is created for each file with each known file type as:*

$$e_j = \sum_{i=0}^{n} \begin{cases} 1 & \textit{if } M_{ij} = F_{ij} \textit{ for } 0 \leq i \leq n, 0 \leq j \leq o \\ 0 & \textit{if } M_{ij} \neq F_{ij} \textit{ for } 0 \leq i \leq n, 0 \leq j \leq o \end{cases}$$

Note that $e_{1\times o}$ provides a numerical measurement for the comparison between each file identification vector to the identification map. Given the equality score vector $e_{1\times o}$, the file is labeled according to the corresponding index of the maximal entry of e.

5.3 Experiments and Results

Now we conduct experiments to validate the methodology via multi-dimensional synthetic data with known ground truth. Then, we apply the methodology to analyze a real-world dataset of malware detectors.

5.3.1 Validation Experiments with Synthetic Data

Generating synthetic data. The synthetic dataset was created by producing one million example files of eight file types, denoted by **T1**-**T8**, randomly assigned. We consider each distinct file type to be an independent category, such that two files that share the same type would be classed the same and two files that differ would be classed differently. Each detector was assigned a detection rate per file type and recording detection vectors, effectively leading to 8,000,000 scan results.

Synthetic experiment setup. We consider 8 sets of results. Each result uses a set of detectors that are characterized by their true-positive rate (TPR) and a true-negative rate (TNR) per file type, while we derive false-positive (FPR) and false-negative(FNR) rates as $FPR = 1 - TNR$ and $FNR = 1 - TPR$. These values are recorded in the synthetic experiment. Each detector generates a label on the file type of each file according to the detector's FPR and FNR, as follows.

If the ground truth label of the type for the file is detected, then the detector labels it as 1 (detected) with probability $1 - FNR$ and labels it as 0 (not detected) with probability FNR; If the ground truth label of the file type for the file is not present, then the detector labels it as 0 (not detected) with probability $1 - FPR$ and labels it as 1 (detected) with probability FPR. Given the synthetic labels, the efficacy of a detector can be defined as its accuracy, $ACC^* = \sum_{i=1}^{8} \frac{TP_i^* + TN_i^*}{8(TP_i^* + TN_i^* + FP_i^* + FN_i^*)}$, where TP_i^* is the synthetic true-positive label for file type i in the file (i.e., the ground truth of a file type is present in the file and the file type is also detected in the file

by the detector in question), TN_i^* is the synthetic true-negative labels, FP_i^* is the synthetic false-positive labels, and FN_i^* is the synthetic false-negative labels. We define the true positive rate as $TPR_i^* = \frac{TP_i}{TP_i+FN_i}$ and the true negative rate as $TNR_i^* = \frac{TN_i}{TN_i+FP_i}$.

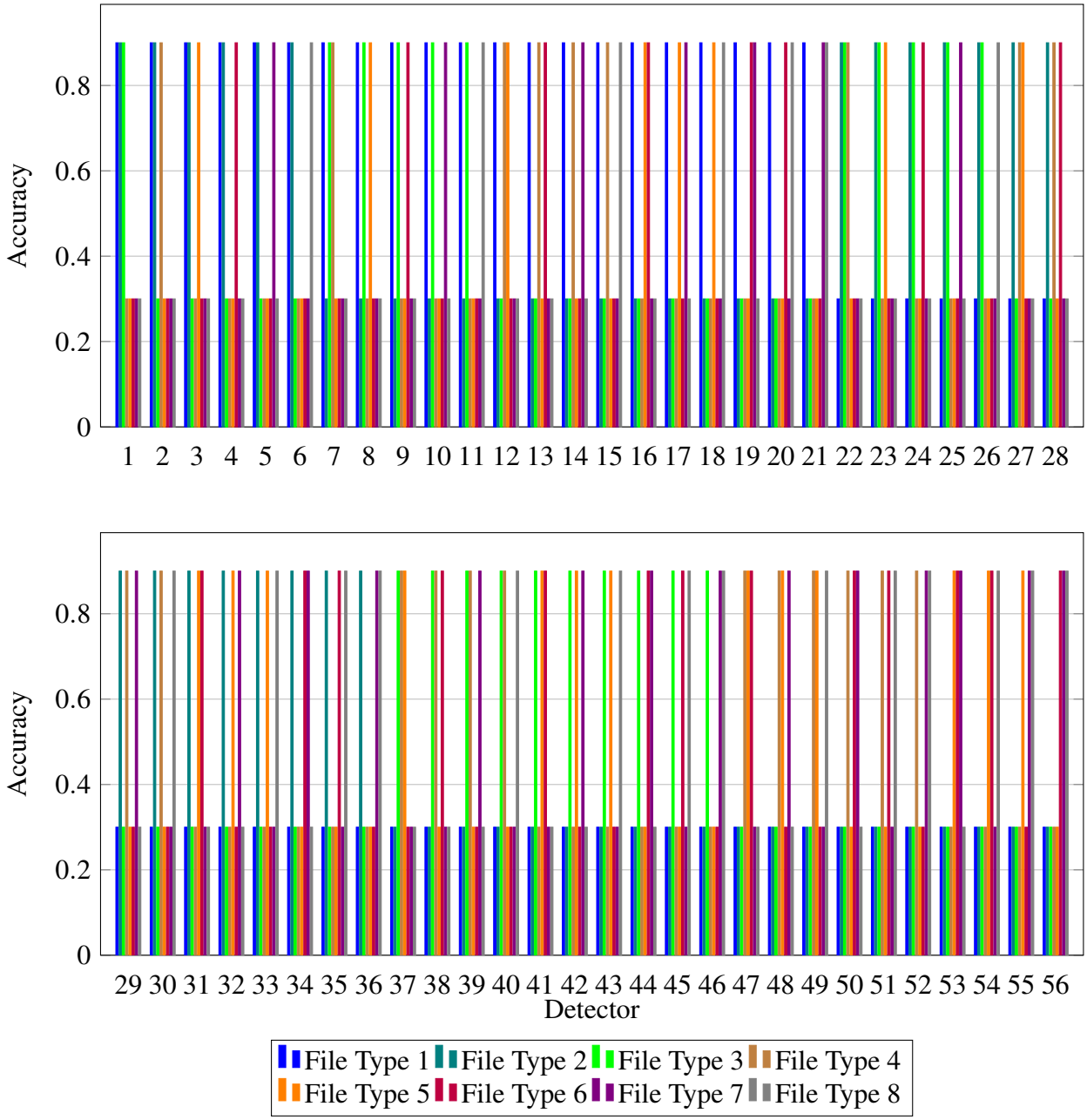


Figure 5.1: Assigned accuracy rates of each detector for each file type. Each detector has an assigned accuracy of 90% for 3 file types and 30% for the other 5 file types. No detectors share the same rates for all of the file types.

We use the synthetic file type sets **T1-T8** to derive the detectors' relative accuracy for each file

type, and then use their relative accuracy to derive their absolute inferred metrics (i.e., accuracy ACC', TNR', and TPR'), which is then compared with the ground truth values (ACC^*, TPR^*, and TNR^*) to determine the effectiveness of our algorithms. We look at both per-type metrics, and also the broader per-file metrics.

Detectors were created such that each detector is considered good ($\approx 90\%$ accuracy) at detecting 3 file types, and poor ($\approx 30\%$ accuracy) at detecting the other 5. Each detector is designed to be good at detecting a distinct set of file types. This design results in a total of 56 unique detectors, plus one Bellwether detector, as presented in [3]. The Bellwether detector is adapted here to scan each type of each file, and will be used as a reference point for the other detectors. This is a novel approach to the use of the Bellwether, which was initially designed to only scan the entire file.

Figure 5.1 displays the assigned ground truth accuracy for each detector, per file type. As seen here, there are exactly 21 detectors per file type that are rated good, and 35 that are rated as poor, all evenly distributed. The Bellwether detector is omitted from display.

5.3.2 Experimental Results

For each file type in the synthetic data, there are four sets of values: the *Ground Truth* values (including the ground truth labels, the ground truth true positive rates, the ground truth true negative rates, and the ground truth accuracy rates), the *Relative Accuracy* values, the *Bellwether Accuracy* values, and finally, the *Inferred Metric* values (including the inferred labels, the inferred true positive rates, the inferred true negative rates, and the inferred accuracy rates), which are elaborated below.

- Ground Truth: The values recorded upon data generation, kept separate from the algorithms and used to evaluate final results.

- Majority Vote: The accuracy measurement if we were to provide each detector with one vote, and count all votes equally. The final result is based on the majority vote; If more than 50% of the detectors label a file type as detected, that label is considered detected, otherwise it is not.

- Bellwether Accuracy: The rational scale provided by Algorithm 7.
- Inferred Metrics: The recovered metrics for each detector provided by Algorithm 8.
- Limited Metrics: A subset of the recovered metrics, using detectors that were well rated and eliminating the vote of poor detectors completely.

Additionally, we look at values concerning the files as a whole, indicating correct classification based on the file type and detectors selected. It also provides information on how to utilize the multidimensional similarity matrix in conjunction with accuracy rating to facilitate the selection of heterogeneous detectors to provide a robust defense against a wide base of threats.

Accuracy per File Type

Figure 5.2 plots the experimental results for each detector's accuracy in regards to each file type. Four values are shown: the Ground Truth Accuracy Rate, the majority vote Accuracy Rate, the Inferred Accuracy Rate, and the Bellwether Accuracy rate [3]. The nature of the way the experiment was built provides for very uniform distribution of values, which is apparent in the consistent results seen across the 56 detectors and 8 file types.

Recall that the Ground Truth Accuracy is the accuracy value that was assigned to each detector when generating the synthetic data. Recall also that the majority vote accuracy measures each detector's accuracy if each is given an equal voting weight, and that the file is labelled when at least half of the detectors agree on a label. In general, with Majority Voting, the poor detectors were measured with an accuracy rate of $\approx 48\%$, while the good detectors were measured with an accuracy rate of $\approx 66\%$. This overestimates the poor detectors by $\approx 18\%$ while underestimating the good detectors by $\approx 24\%$. The Bellwether Accuracy is the result of using Algorithm 7. The accuracy ratings in Figure 5.2 represent the output of the algorithm, and are used as an intermediary step to recovering inferred metric values. These ratings replace the equal voting weights used in the Majority Vote method, and are used in conjunction with Algorithm 8.

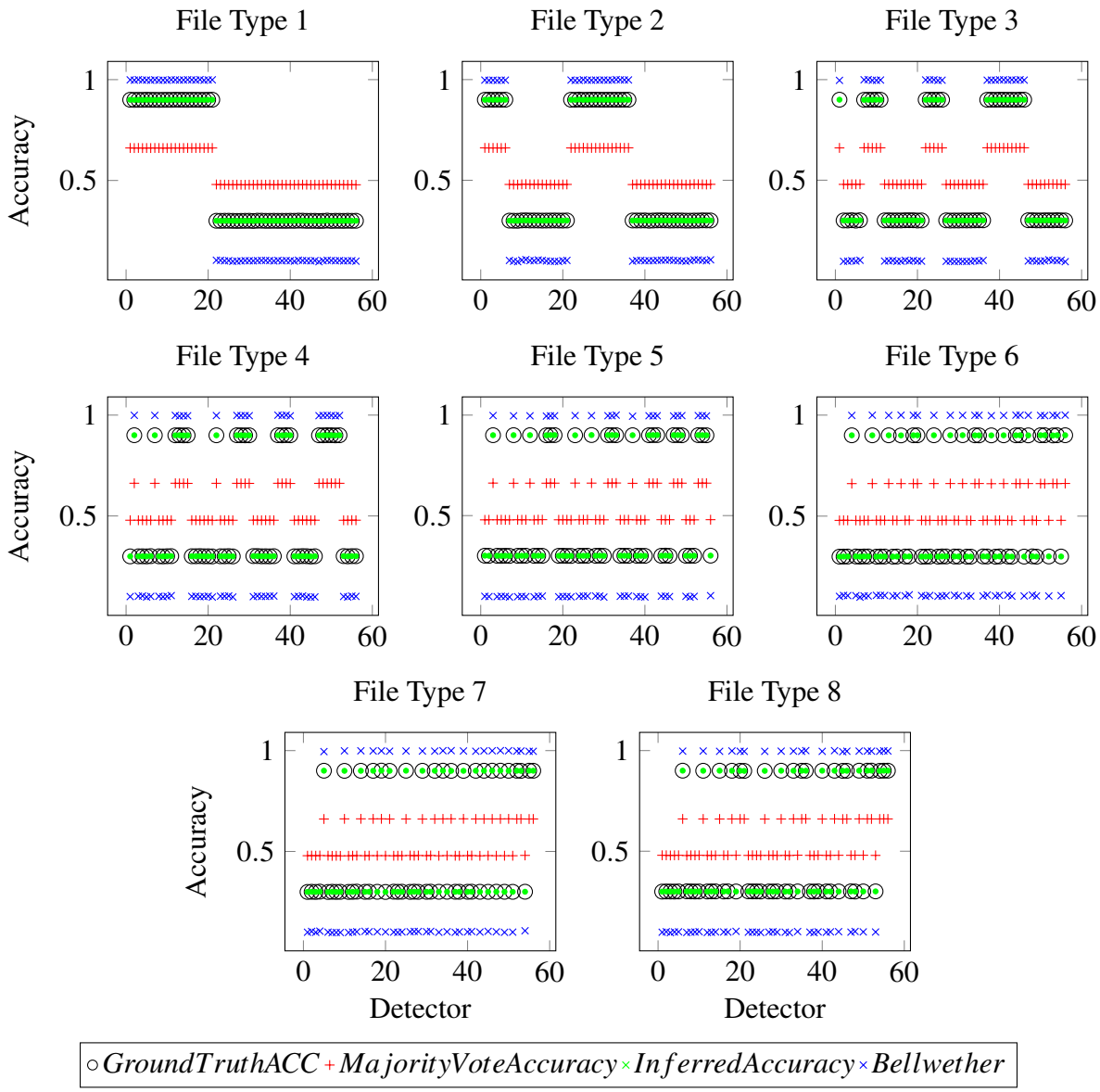
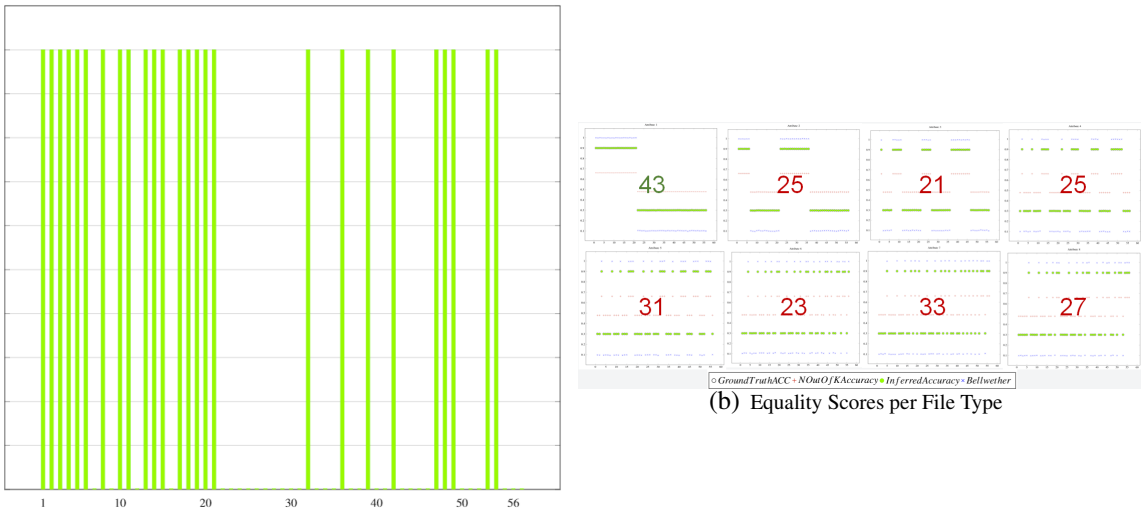


Figure 5.2: Results comparing the Bellwether Accuracy, Inferred Accuracy, and Majority Vote Accuracy measurements, graphed against the recorded ground truth values.

File Identification

In this experiment, each file was scanned by each detector. This means that for each file scanned, the result of that scan is a vector $v_j \in v_{jk} \in V_{ijk}$ as defined in Definition 10, which we utilize as the file identifier vector, $f_j \in F_{n\times o} = [f_1, f_2, \ldots, f_j]$, $0 \leq j \leq o$. This file identification is then used in comparison with each identification map $m_j \in M_{n\times o} = [m_1, m_2, \ldots, m_j]$ for $0 \leq j \leq o$.

Figure 5.3 shows a file identification for a scanned file of file type 1 and the Equality scores



(a) File Identification

(b) Equality Scores per File Type

Figure 5.3: File Identification from a scan of File Type 1 and the corresponding file type Equality scores, per file type.

measured for each file type. File type 1 had a score indicating that 43 out of 56 detectors matched the ID Map provided by the Inferred Accuracy scores. For our test data, we mapped 250,000 files of each type, for a total of 2 million files, using the provided method, and found that the top Equality Score matched the ground truth file type in all cases.

Accuracy Over All File Types

Knowing that each detector was measured with an majority voting accuracy of $\approx 66\%$ for its assigned 3 good file types, and $\approx 48\%$ for its assigned 5 poor file types, we would expect the average majority voting accuracy for each file type to be $\frac{(3*0.66\%+5*0.48\%)}{8} = 0.5475\%$. This result holds up to our measured majority vote accuracy for each detector across all 8 file types, as shown in Figure 5.4

Our inferred accuracy, on the other hand, was measured at $\approx 90\%$ for each detector's good file types and $\approx 30\%$ for each poor file type. This tracked closely with the known ground truth values. The expected accuracy over all file types for each detector would then be $\frac{(3*0.90\%+5*0.30\%)}{8} = 0.525\%$. Figure 5.4 shows that this matches the measurements from the experiments as well.

While the estimated majority vote accuracy is listed as higher in Figure 5.4, this is only because

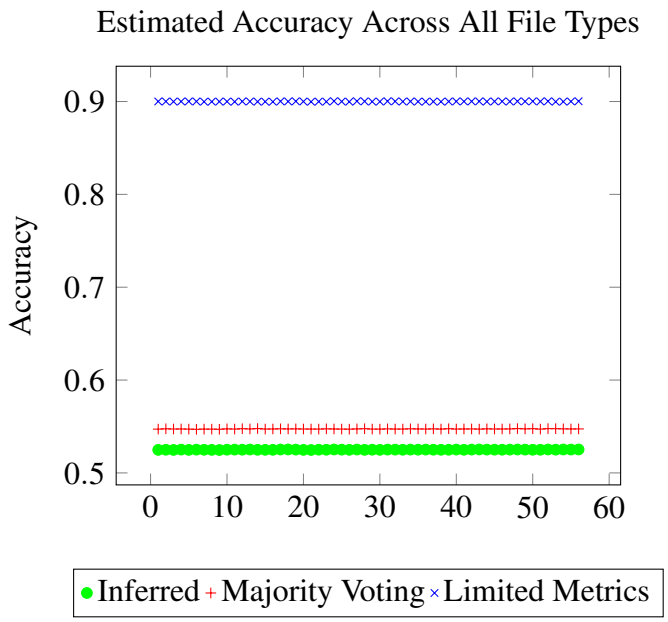


Figure 5.4: Average estimated accuracy of each detector across all files and all 8 file types combined.

the majority vote over estimates the accuracy of each detector. The Inferred accuracy closely matches with the known ground truth accuracy of the detectors when the results are combined across all file types.

Additionally, the inferred metrics are further utilized to allow the selection of votes only from detectors that are rated "good." Knowing the inferred metrics across all 8 file types, the accuracy was measured with these limited detectors, and these measurements were labeled as the *Limited* metrics. As shown in Figure 5.4, the limited metrics select detectors based on which are rated as good and ignores input from poorly performing detectors per file type. This results in an overall increase in accuracy across all 8 million scans, and provided an accuracy of 90% for all detectors.

Insight 11. *While* inferred accuracy *is more correct,* majority voting *overestimates the accuracy of each detector.*

Insight 11 says that majority voting can lead to inaccurate results. Precisely pinning down when majority voting works it left as an outstanding open problem.

Insight 12. *Using only those detectors that are rated well for each file type provides an inherently better accuracy score over all files.*

Insight 12 says that we should always try to filter out the poor detectors.

Next, we calculated the accuracy over all 8 million files, over all file types, using majority voting, weighted voting with inferred metrics, and limited metrics. Figure 5.7, on the left, shows the overall accuracy for each voting type, across each file type.

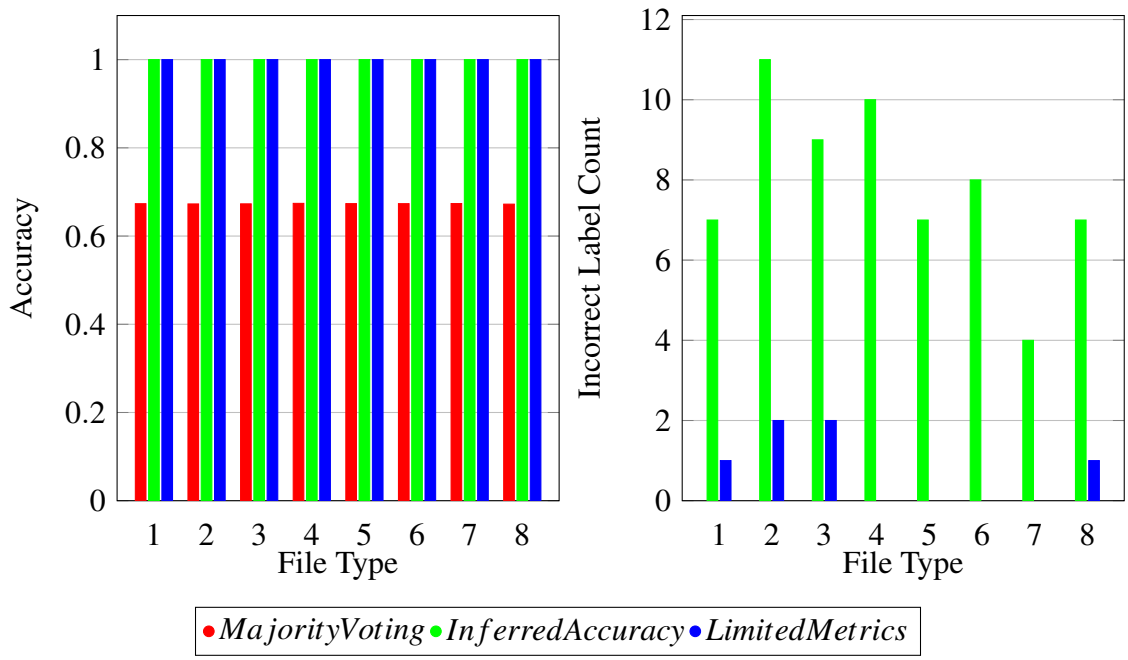


Figure 5.5: On the left, a comparison of correctly labeled files, by file type, for Majority Voting, Inferred Metrics Using All Detectors, and Limited Metrics. On the right, a count of the files *incorrectly* labeled using the Inferred Metrics and Limited Metrics as voting weights.

Here, we see that the majority vote voting strategy had an overall accuracy of $\approx 67.3\%$ across the file types, equating to $\approx 2,616,000$ mislabeled files. The labels for the inferred and limited metrics were much more accurate. Figure 5.7 on the right counts the total number of mislabeled files for these two weighted vote measurements, with the Inferred Accuracy correctly classifying $\approx 99.99\%$ of the files and mislabeling 63 total files. Using limited metrics, only 6 files were mislabeled, nearing 100% accuracy.

Being able to accurately recover cybersecurity metrics allows us to eliminate poor detectors. This gives us the ability to selectively apply good detectors for the given problem and reduce our margin of error. In our experiments, we reduced the margin of error by a factor of 10 between the inferred metrics and the limited metrics.

Selection of Heterogeneous Detectors

The recovery of metrics in regards to detectors has been shown to be highly advantageous. Now, we examine the issue of selecting good set detectors to provide good coverage, where coverage means that there is a good chance of detecting each type of file with the reduced set of detectors. In our experiments, each detector is equally accurate, but across a different set of file types. Using this information, we can select the most dissimilar detectors to provide the best information in accurately categorizing files of each file type.

In order to select dissimilar detectors, it makes intuitive sense to utilize the similarity matrix. Figure 5.6 displays a heat map of the similarity matrix on the left. We note that in this case, there are five distinct classes of similarity that we can recognize.

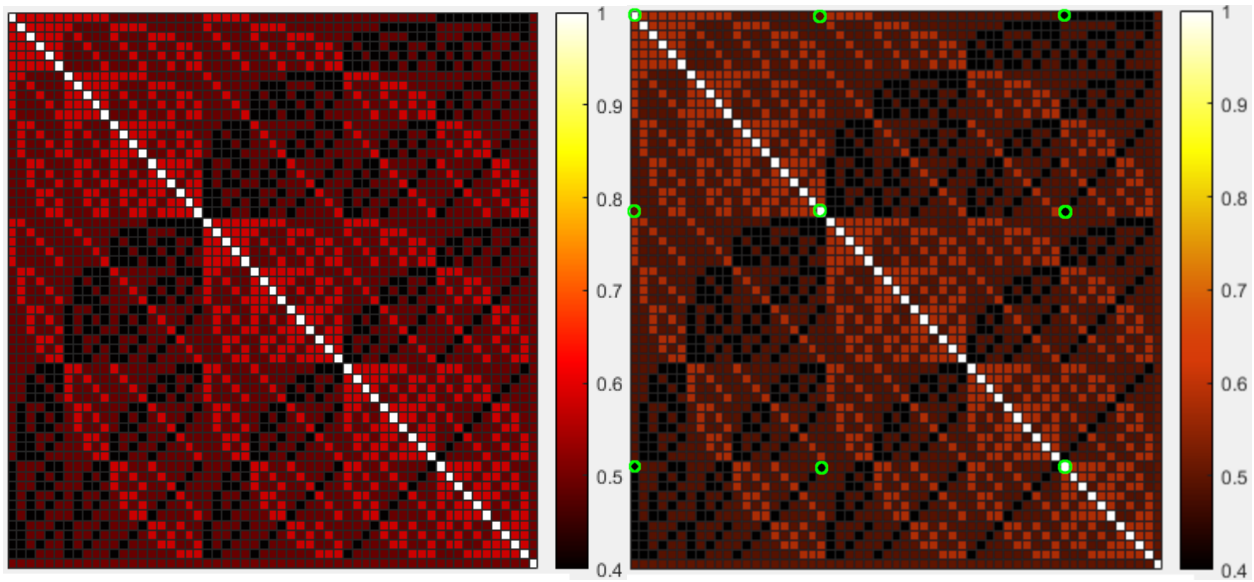
- Self Similar: The main diagonal of the matrix shows that every detector is self similar, and has a similarity score of 1 with itself. This matches Definition 15. Due to the design of our experiments, this is the only case where a detector has a good rating for the same three file types.

- Two File Types Shared: Where two detectors are both rated good with two common file types, we see that they have a similarity rating of $\approx 58\%$. We note that there are fifteen such matches for each detector due to the design of our experiments.

- One File Type Shared: Where two detectors are both rated good with one common file type, we see that they have a similarity rating of $\approx 49\%$. We note that there are ten such matches for each detector due to the design of our experiments.

- No File Types Shared: Where two detectors have no common file types where they are rated good, we see that they have a similarity rating of $\approx 40\%$. We note that there are thirty such matches for each detector due to the design of our experiments.

- Bellwether Similarity: The Bellwether detector is a special case, and shares a similarity rating of $\approx 50\%$ with all detectors.



(a) Experimental Similarity Matrix (b) Dissimilar Detector Selection

Figure 5.6: Heat map of the synthetic similarity matrix on the left, and selection of dissimilar detectors on the right.

For our experimental data, we know the number of classes and we know the which detectors are good at detecting which types of files. Due to this, we know that we can obtain full coverage of all file types with no less than three detectors. Because all detectors in our experiment have a similar accuracy when we score them across all file types, we start by simply selecting a detector. In this case, we select detector 1.

Next, we want to select another detector, but one that is as dissimilar as possible while still having good accuracy. We look for a detector that shares no good ratings for file types in relation to the first detector selected, which means it has a similarity rating of $\approx 40\%$. We select detector 47.

Finally, we select a third detector that's as dissimilar as possible to the previous two. In this case, there is no detector that will not share a good rating with at least one file type in regards to the previously selected detectors. As such, we select one that shares a good accuracy rating with 0 file types in regards to detector 47, and 1 file type in regards to detector 1. We select detector 21.

By examining Figure 5.1, we observe the following: detector 1 is good at detecting file types 1, 2, and 3; detector 21 is good at detecting file types 1, 7, and 8; and detector 47 is good at

detecting file types 4, 5, and 6. This validates our intuition that utilizing the Similarity matrix to select dissimilar detectors can provide good coverage over the file types in question.

Utilizing just these three detectors and their limited metrics, we find that they are able to detect files of each type with a reasonable accuracy, and that they are able to do so at a significantly higher margin than the majority vote voting utilizing all 56 detectors.

Figure 5.7 shows the detection accuracy for each file type when using just these three detectors. Our results show they are able to detect the file type with accuracy rates between 91.03% at the highest and 89.94% at the lowest, when accounting for limited metrics and measuring all 8 million files:

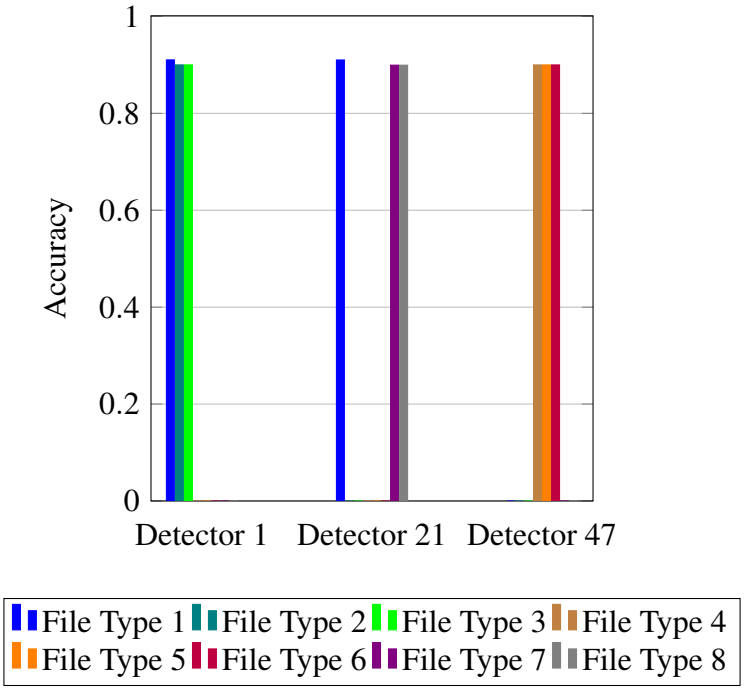


Figure 5.7: Accuracy of detectors 1, 21, and 47 per file type, showing that 3 detectors provide coverage over all 8 file types.

The slightly higher accuracy rate for file type 1 is understood due to the fact that we had two detectors with good accuracy ratings for that file type. The other scores are in range for what would be expected for a single detector with a 90% accuracy rating. Using the majority vote classification, we found an accuracy of 67.30% across each of the file types.

In summary, using the similarity matrix to select three dissimilar detectors allowed us to in-

crease our classification accuracy by approximately 23% over the majority voting, while reducing the number of detectors by $\approx 94.64\%$. This shows that the recovered metrics and similarity ratings can be used together.

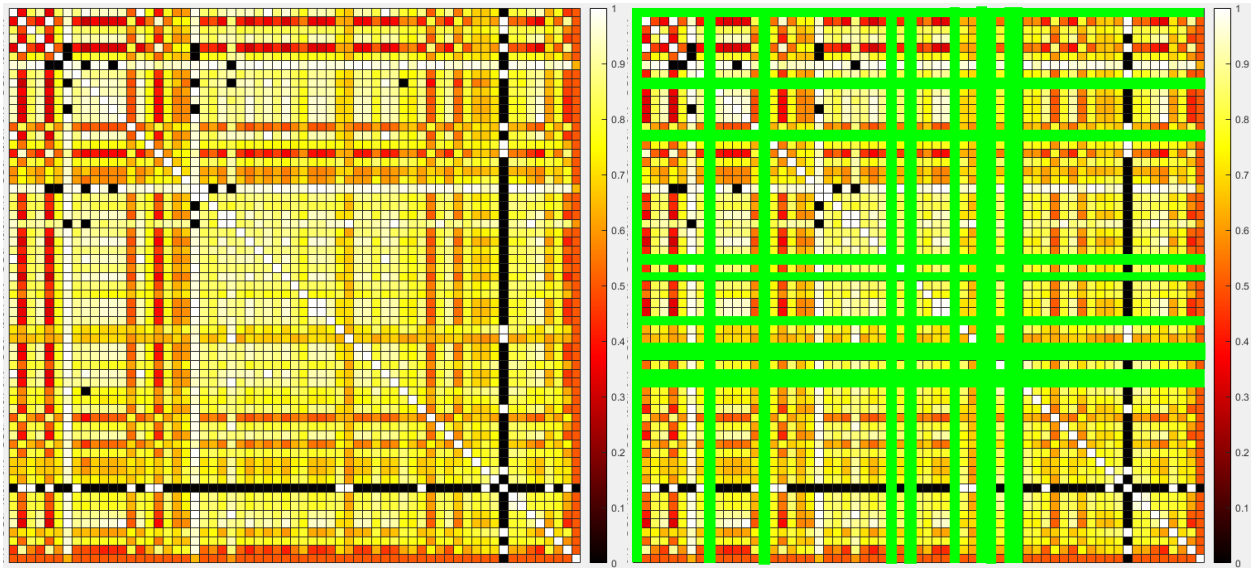
5.3.3 Applying the Methodology to Evaluate a Real Dataset

As mentioned before, the real-world dataset contains $m \approx 10.7$ million files collected from VirusTotal, each of which was scanned by up to $n = 62$ anti-malware detectors. However, not every detector scanned every file. Each detector labels a file it scanned as malicious (1) or benign (0). The dataset is transformed to matrix $(\mathbf{V}_{ij})_{n \times m}$, from which we derive the similarity matrix $\mathbf{S}$ and then the Inferred Metrics $\mathbf{T}$ according to Algorithm 8.

In this case, the underlying file types are hidden from us. This is expected, as the primary data we're working with is the simple detection vector for each detector. However, as shown previously, we can still use the the similarity matrix to make a selection of dissimilar detectors that are highly rated in regards to accuracy. From there, we contrast the results with the majority voting using all the detectors. We note that individually, the selected detectors had accuracy ratings that ranged from a high of 92.67% to a low of 90.80%. When utilized together with weighted voting based on their Inferred Metrics, we were able to correctly identify 96.17% of the files correctly. In contrast, the majority vote accuracy utilizing all the detectors was 56.07%

We select the top 10 most trusted, yet most dissimilar detectors. The Similarity Matrix and the Detector Selection are mapped in Figure 5.8. This shows that the Similarity Matrix, and corresponding algorithms for recovering metrics, can be utilized in tandem to great effect. We are able to identify and eliminate poor detectors from providing noisy votes that would cause improper classification of files, and identify those detectors that provide the best coverage when used together. We have also shown that we can greatly improve upon the standard practice of using majority voting, and that we can do so in the absence of ground truth data that.

In summary, being able to select a few "good" detectors can provide accurate results with less overhead. With less than 20% of the votes, we are able to achieve high accuracy,



(a) Real Similarity Matrix (b) Dissimilar Detector Selection

Figure 5.8: Heat map of the real world data similarity matrix on the left, and selection of dissimilar detectors on the right.

5.4 Chapter Summary

In this work, we extended the Similarity Matrix definition and the algorithms for the Bellwether and Inferred Metrics to account for multiple file types. We then used synthetic data to validate the methods, showing that accurate metrics for each detector can be recovered for each file type. Additionally, we showed how this data could be used to create a file identification and an identification map for each detector type, which allows for quick classification.

We then examined how Inferred Metrics for each detector can be used to eliminate the poor detectors in regards to each file type, while keeping the information from the good detectors. Utilizing this information, we then demonstrated how to select heterogeneous detectors through analysis of the Similarity Matrix, and how to select good detectors by synthesizing data from both sources.

We used these same methods to evaluate the real world dataset from VirusTotal to select a set of heterogeneous and good detectors, and evaluate their effectiveness in identifying real world files as compared to the standard majority vote method of identification.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 Conclusion

Measuring malware detector metrics in the absence of ground-truth labels (or information) is an important problem that has yet to be tackled. The problem is relevant because ground-truth labels are often expensive to obtain in the real world. This is especially true when a large number of malware examples are involved, which is certainly the situation we often encounter in practice. The present dissertation makes a solid step towards ultimately tackling this important problem, by making three contributions.

First, we formulated the problem of estimating the *relative accuracy* of malware detectors in the absence of ground truth and presented an algorithm to derive the relative accuracy. We validated the proposed algorithm based on given synthetic data with ground truth. We then applied the algorithm to a real-world dataset obtained from VirusTotal. Extensive experiments showed that the proposed algorithm is capable of ranking the relative accuracies of the 62 real-world detectors which scanned millions of files. We identified 4 detectors that not only are useless, but also may do more harm than good.

Second, we presented an improved algorithm to derive the relative accuracy of malware detectors in the absence of ground truth. We explored the relationship between the relative accuracy and the principal component analysis method. We introduced the idea of the Bellwether detector for the purpose of improving the accuracy. We showed that our refined method can be used to recover the metrics of malware detectors in the absence of ground truth. We applied these methods to evaluate the real world dataset obtained from VirusTotal.

Third, we investigated how to deal with multiclass malware detectors, namely metrics in regards to individual attributes of files that are scanned by the detectors. We analyzed the specific ability of various detectors and characterized when detectors can provide detailed data in regards to certain traits, and assigning accuracy. We then used this information in conjunction with the Similarity Matrix to select an effective subset of heterogeneous detectors.

6.2 Future Research

This dissertation opens the door to a little investigated, but important, field. There are a number of open problems for future research. First, it is an outstanding open problem to develop a theoretical evaluation framework to precisely characterize when the proposed algorithm is useful and when it isn't. Second, it is interesting to characterize the co-variance and correlation between the accuracy of detectors. Third, it is useful to develop a principled aggregation engine to incorporate detection labels of multiple malware detectors. Fourth, it is interesting to identify the key characteristics of poor detectors, so as to avoid the use of them. Fifth, how can we quantify the (relative) accuracy of multiclass malware detector (as opposed to a binary detector)?

BIBLIOGRAPHY

[1] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, pages 39–57, 2017.

[2] John Charlton, Pang Du, Jin-Hee Cho, and Shouhuai Xu. Measuring relative accuracy of malware detectors in the absence of ground truth. In *MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM)*, pages 450–455. IEEE, 2018.

[3] John Charlton, Pang Du, and Shouhuai Xu. Leveraging relative accuracy to infer metrics in the absence of ground truth. In *TBD*.

[4] H. Chen, J. Cho, and S. Xu. Quantifying the security effectiveness of firewalls and dmzs. In *Proc. HoTSoS'2018*, pages 9:1–9:11, 2018.

[5] H. Chen, J. Cho, and S. Xu. Quantifying the security effectiveness of network diversity. In *Proc. HoTSoS'2018*, page 24:1, 2018.

[6] H. Chen, M. Pendleton, L. Njilla, and S. Xu. A survey on ethereum systems security: Vulnerabilities, attacks, and defenses. *ACM Comput. Surv.*, 53(3):67:1–67:43, 2020.

[7] L. Chen, S. Hou, Y. Ye, and S. Xu. Droideye: Fortifying security of learning-based classifier against adversarial android malware attacks. In *Proc. 2018 IEEE/ACM ASONAM*, pages 782–789, 2018.

[8] Y. Cheng, J. Deng, J. Li, S. DeLoach, A. Singhal, and X. Ou. Metrics of security. In *Cyber Defense and Situational Awareness*, pages 263–295. 2014.

[9] J. Cho, P. Hurley, and S. Xu. Metrics and measurement of trustworthy systems. In *IEEE Military Communication Conference (MILCOM 2016)*, 2016.

[10] J. Cho, S. Xu, P. Hurley, M. Mackay, T. Benjamin, and M. Beaumont. Stram: Measuring the trustworthiness of computer-based systems. *ACM Comput. Surv.*, 51(6):128:1–128:47, 2019.

[11] A. P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):20–28, 1979.

[12] P. Du, Z. Sun, H. Chen, J. H. Cho, and S. Xu. Statistical estimation of malware detection metrics in the absence of ground truth. *IEEE T-IFS*, 13(12):2965–2980, 2018.

[13] X. Fang, M. Xu, S. Xu, and P. Zhao. A deep learning framework for predicting cyber attacks rates. *EURASIP J. Information Security*, 2019:5, 2019.

[14] Zijian Fang, Maochao Xu, Shouhuai Xu, and Taizhong Hu. A framework for predicting data breach risk: Leveraging dependence to cope with sparsity. *IEEE Trans. Inf. Forensics Secur.*, 16:2186–2201, 2021.

[15] Zijian Fang, Maochao Xu, Shouhuai Xu, and Taizhong Hu. A framework for predicting data breach risk: Leveraging dependence to cope with sparsity. *IEEE Trans. Inf. Forensics Secur.*, 16:2186–2201, 2021.

[16] Zijian Fang, Peng Zhao, Maochao Xu, Shouhuai Xu, Taizhong Hu, and Xing Fang. Statistical modeling of computer malware propagation dynamics in cyberspace. *Journal of Applied Statistics*, 0(0):1–26, 2020.

[17] Eric Ficke and Shouhuai Xu. Apin: Automatic attack path identification in computer networks. In *IEEE ISI'2020*, 2020.

[18] Richard Garcia-Lebron, David J Myers, Shouhuai Xu, and Jie Sun. Node diversification in complex networks by decentralized colouring. *Journal of Complex Networks*, 7(4):554–563, 5 2019.

[19] Richard Garcia-Lebron, Kristin Schweitzer, Raymond Bateman, and Shouhuai Xu. A framework for characterizing the evolution of cyber attacker-victim relation graphs. In *IEEE Milcom'2018*. 2018.

[20] Y. Han, W. Lu, and S. Xu. Characterizing the power of moving target defense via cyber epidemic dynamics. In *HotSoS*, pages 1–12, 2014.

[21] Yujuan Han, Wenlian Lu, and Shouhuai Xu. Preventive and reactive cyber defense dynamics with ergodic time-dependent parameters is globally attractive. *CoRR*, abs/2001.07958, 2020.

[22] J. Homer, S. Zhang, X. Ou, D. Schmidt, Y. Du, S. Rajagopalan, and A. Singhal. Aggregating vulnerability metrics in enterprise networks using attack graphs. *J. Comput. Secur.*, 21(4):561–597, 2013.

[23] Luca Invernizzi, Stefano Benvenuti, Marco Cova, Paolo Milani Comparetti, Christopher Kruegel, and Giovanni Vigna. Evilseed: A guided approach to finding malicious web pages. IEEE Symposium on Security and Privacy (2012), pages 428–442.

[24] Alex Kantchelian, Michael Carl Tschantz, Sadia Afroz, Brad Miller, Vaishaal Shankar, Rekha Bachwani, Anthony D Joseph, and J Doug Tygar. Better malware ground truth: Techniques for weighting anti-virus vendor labels. In *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, pages 45–56. ACM, 2015.

[25] Marc Kührer, Christian Rossow, and Thorsten Holz. Paint it black: Evaluating the effectiveness of malware blacklists. In *Proc. Research in Attacks, Intrusions and Defenses (RAID'14)*, pages 1–21.

[26] D. Li, Q. Li, Y. Ye, and S. Xu. A frameowrk for enhancing deep neural networks against adversarial malware examples. *CoRR*, abs/2004.07919, 2020.

[27] D. Li, Q. Li, Y. Ye, and S. Xu. Sok: Arms race in adversarial malware detection. *CoRR*, abs/2005.11671, 2020.

[28] Deqiang Li, Qianmu Li, Yanfang Ye, and Shouhuai Xu. A framework for enhancing deep neural networks against adversarial malware. *IEEE Trans. Netw. Sci. Eng.*, 8(1):736–750, 2021.

[29] X. Li, P. Parker, and S. Xu. Towards quantifying the (in)security of networked systems. In *21st IEEE International Conference on Advanced Information Networking and Applications (AINA'07)*, pages 420–427, 2007.

[30] Xiaohu Li, Paul Parker, and Shouhuai Xu. A stochastic model for quantitative security analyses of networked systems. *IEEE Transactions on Dependable and Secure Computing*, 8(1):28–43, 2011.

[31] Z. Li, D. Zou, S. Xu, H. Jin, H. Qi, and J. Hu. Vulpecker: an automated vulnerability detection system based on code similarity analysis. In *Pro. ACSAC'16*, pages 201–213, 2016.

[32] Z. Li, D. Zou, S. Xu, H. Jin, Y. Zhu, Z. Chen, S. Wang, and J. Wang. Sysevr: A framework for using deep learning to detect software vulnerabilities. *IEEE Transactions on Dependable and Secure Computing (accepted for publication)*, 2021.

[33] Z. Li, D. Zou, S. Xu, H. Jin, Y. Zhu, Z. Zhang, Z. Chen, and D. Li. Vuldeelocator: A deep learning-based system for detecting and locating software vulnerabilities. IEEE TDSC (accepted for publication), 2021.

[34] Zongzong Lin, Wenlian Lu, and Shouhuai Xu. Unified preventive and reactive cyber defense dynamics is still globally convergent. *IEEE/ACM Trans. Netw.*, 27(3):1098–1111, 2019.

[35] Zhaofeng Liu, Ren Zheng, Wenlian Lu, and Shouhuai Xu. Using event-based method to estimate cybersecurity equilibrium. *IEEE CAA J. Autom. Sinica*, 8(2):455–467, 2021.

[36] W. Lu, S. Xu, and X. Yi. Optimizing active cyber defense dynamics. In *Proc. GameSec'13*, pages 206–225, 2013.

[37] J. Mireles, E. Ficke, J. Cho, P. Hurley, and S. Xu. Metrics towards measuring cyber agility. *IEEE T-IFS*, 14(12):3217–3232, 2019.

[38] A. Mohaisen and O. Alrawi. Av-meter: An evaluation of antivirus scans and labels. In *Proc. DIMVA*, pages 112–131, 2014.

[39] Rosana Montanez, Edward Golob, and Shouhuai Xu. Human cognition through the lens of social engineering cyberattacks. *Frontiers in Psychology*, 11:1755, 2020.

[40] Jose Morales, Shouhuai Xu, and Ravi Sandhu. Analyzing malware detection efficiency with multiple anti-malware programs. In *Proc. CyberSecurity*, 2012.

[41] Steven Noel and Sushil Jajodia. *A Suite of Metrics for Network Attack Graph Analytics*, pages 141–176. Springer International Publishing, Cham, 2017.

[42] Cameron Nowzari, Victor M. Preciado, and George J. Pappas. Analysis and control of epidemics: A survey of spreading processes on complex networks. *IEEE Control Systems*, 36(1):26–46, 2016.

[43] M. Pendleton, R. Garcia-Lebron, J. Cho, and S. Xu. A survey on systems security metrics. *ACM Comput. Surv.*, 49(4):62:1–62:35, 2016.

[44] Chen Peng, Maochao Xu, Shouhuai Xu, and Taizhong Hu. Modeling and predicting extreme cyber attack rates via marked point processes. *Journal of Applied Statistics*, 44(14):2534–2563, 2017.

[45] Chen Peng, Maochao Xu, Shouhuai Xu, and Taizhong Hu. Modeling multivariate cybersecurity risks. *Journal of Applied Statistics*, 0(0):1–23, 2018.

[46] Roberto Perdisci and ManChon U. Vamo: Towards a fully automated malware clustering validity analysis. In *Proceedings of the 28th Annual Computer Security Applications Conference*, ACSAC '12, pages 329–338, 2012.

[47] Mir Pritom, Kristin Schweitzer, Raymond Bateman, Min Xu, and Shouhuai Xu. Characterizing the landscape of covid-19 themed cyberattacks and defenses. In *IEEE ISI'2020*, 2020.

[48] Mir Pritom, Kristin Schweitzer, Raymond Bateman, Min Xu, and Shouhuai Xu. Data-driven characterization and detection of covid-19 themed malicious websites. In *IEEE ISI'2020*, 2020.

[49] A. Ramos, M. Lazar, R. H. Filho, and J. J. P. C. Rodrigues. Model-based quantitative network security metrics: A survey. *IEEE Communications Surveys Tutorials*, 19(4):2704–2734, 2017.

[50] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Anna Jerebko, Charles Florin, Gerardo Hermosillo Valadez, Luca Bogoni, and Linda Moy. Supervised learning from multiple experts: Whom to trust when everyone lies a bit. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 889–896, 2009.

[51] Piet Van Mieghem, Jasmina Omic, and Robert Kooij. Virus spread in networks. *IEEE/ACM Trans. Netw.*, 17(1):1–14, February 2009.

[52] L. Wang, S. Jajodia, and A. Singhal. *Network Security Metrics*. Springer, 2017.

[53] L. Wang, S. Jajodia, A. Singhal, P. Cheng, and S. Noel. k-zero day safety: A network security metric for measuring the risk of unknown vulnerabilities. *IEEE TDSC*, 11(1):30–44, 2014.

[54] Q. Wang, W. Guo, K. Zhang, A. Ororbia II, X. Xing, X. Liu, and C. Giles. Adversary resistant deep neural networks with an application to malware detection. In *ACM KDD*, pages 1145–1153, 2017.

[55] L. Xu, Z. Zhan, S. Xu, and K. Ye. Cross-layer detection of malicious websites. In *ACM CODASPY*, pages 141–152, 2013.

[56] L. Xu, Z. Zhan, S. Xu, and K. Ye. An evasion and counter-evasion study in malicious websites detection. In *IEEE CNS*, pages 265–273, 2014.

[57] M. Xu, L. Hua, and S. Xu. A vine copula model for predicting the effectiveness of cyber defense early-warning. *Technometrics*, 59(4):508–520, 2017.

[58] M. Xu and S. Xu. An extended stochastic model for quantitative security analysis of networked systems. *Internet Mathematics*, 8(3):288–320, 2012.

[59] Maochao Xu, Gaofeng Da, and Shouhuai Xu. Cyber epidemic models with dependences. *Internet Mathematics*, 11(1):62–92, 2015.

[60] Maochao Xu, Kristin M. Schweitzer, Raymond M. Bateman, and Shouhuai Xu. Modeling and predicting cyber hacking breaches. *IEEE Trans. Information Forensics and Security*, 13(11):2856–2871, 2018.

[61] S. Xu. Emergent behavior in cybersecurity. In *Proc. HotSoS*, pages 13:1–13:2, 2014.

[62] S. Xu. The cybersecurity dynamics way of thinking and landscape (invited paper). In *ACM Workshop on Moving Target Defense*, 2020.

[63] S. Xu, W. Lu, and L. Xu. Push- and pull-based epidemic spreading in networks: Thresholds and deeper insights. *ACM TAAS*, 7(3), 2012.

[64] S. Xu, W. Lu, and Z. Zhan. A stochastic model of multivirus dynamics. *IEEE Transactions on Dependable and Secure Computing*, 9(1):30–45, 2012.

[65] Shouhuai Xu. Cybersecurity dynamics. In *Proc. HotSoS'14*, pages 14:1–14:2, 2014.

[66] Shouhuai Xu. Cybersecurity dynamics: A foundation for the science of cybersecurity. In *Proactive and Dynamic Network Defense*, pages 1–31. 2019.

[67] Shouhuai Xu, Moti Yung, and Jingguo Wang. Seeking foundations for the science of cyber security. *Information Systems Frontiers*, 2021/04/28.

[68] W. Xu, Y. Qi, and D. Evans. Automatically evading classifiers: A case study on pdf malware classifiers. In *NDSS*, January 2016.

[69] L. Yang, P. Li, X. Yang, and Y. Tang. A risk management approach to defending against the advanced persistent threat. *IEEE TDSC*, pages 1–1, 2018.

[70] L. Yang, X. Yang, and Y. Tang. A bi-virus competing spreading model with generic infection rates. *IEEE Trans. Netw. Sci. Eng.*, 5(1):2–13, 2018.

[71] Zhenxin Zhan, Maochao Xu, and Shouhuai Xu. Characterizing honeypot-captured cyber attacks: Statistical framework and case study. *IEEE Transactions on Information Forensics and Security*, 8(11):1775–1789, 2013.

[72] Zhenxin Zhan, Maochao Xu, and Shouhuai Xu. Predicting cyber attack rates with extreme values. *IEEE Transactions on Information Forensics and Security*, 10(8):1666–1677, 2015.

[73] Jing Zhang, Zakir Durumeric, Michael Bailey, Mingyan Liu, and Manish Karir. On the mismanagement and maliciousness of networks. In *Proc. NDSS'14*, 2014.

[74] M. Zhang, L. Wang, S. Jajodia, A. Singhal, and M. Albanese. Network diversity: A security metric for evaluating the resilience of networks against zero-day attacks. *IEEE Trans. Inf. Forensics Secur.*, 11(5):1071–1086, 2016.

[75] R. Zheng, W. Lu, and S. Xu. Active cyber defense dynamics exhibiting rich phenomena. In *Proc. HotSoS*, 2015.

[76] R. Zheng, W. Lu, and S. Xu. Preventive and reactive cyber defense dynamics is globally stable. *IEEE TNSE*, 5(2):156–170, 2018.

[77] Deqing Zou, Sujuan Wang, Shouhuai Xu, Zhen Li, and Hai Jin. μvuldeepecker: A deep learning-based system for multiclass vulnerability detection. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1, 01 2019.

[78] Deqing Zou, Yawei Zhu, Shouhuai Xu, Zhen Li, Hai Jin, and Hengkai Ye. Interpreting deep learning-based vulnerability detector predictions based on heuristic searching. *ACM Trans. Softw. Eng. Methodol.*, 30(2), March 2021.

VITA

Education

Bachelor of Science in Mathematics, Colorado State University, December 2011.

Bachelor of Science in Computer Science, Colorado State University, December 2011.

Masters of Science in Computer Science, University of Texas, San Antonio, December 2020.

Academic Employments

Undergraduate Research Assistant, Colorado State University, Fort Collins Colorado, August 2010, December 2011

Senior Research Computer Scientist, Southwest Research Institute, San Antonio TX, January 2012, Present

Publications

J. Charlton, P. Du, J. Cho, S. Xu, Measuring Relative Accuracy of Malware Detectors in the Absence of Ground Truth, Military Communications Conference, MILCOM 2018 IEEE

Presentation and Professional Development

Graduate Presenter, Measuring Relative Accuracy of Malware Detectors in the Absence of Ground Truth, MILCOM 2018, Los Angeles, California, 2018