Castles Built on Sand: Observations from Classifying Academic Cybersecurity Datasets with Minimalist Methods

Laurens D'hooge[a], Miel Verkerken[b], Tim Wauters[c], Filip De Turck[d] and Bruno Volckaert[e]
IDLab-Imec, Department of Information Technology, Ghent University,
Technologiepark-Zwijnaarde 126, 9052 Ghent, Belgium

Keywords: Machine Learning, Cybersecurity, Data Quality Issues, Baselines.

Abstract: Machine learning (ML) has been a staple of academic research into pattern recognition in many fields, including cybersecurity. The momentum of ML continues to speed up alongside the advances in hardware capabilities and the methods they unlock, primarily (deep) neural networks. However, this article aims to demonstrate that the non-judicious use of ML in two prominent domains of data-based cybersecurity consistently misleads researchers into believing that their proposed methods constitute actual improvements. Armed with 17 state-of-the-art datasets in traffic and malware classification and the simplest possible machine learning model this article will show that the lack of variability in most of these datasets immediately leads to excellent models, even if that model is only one comparison per feature.

1 INTRODUCTION

The term pattern recognition has largely become synonymous with machine learning, especially in academic research. While in itself this is not a bad direction since these methods have proven capable of solving hitherto unsolvable tasks, eager adoption of increasingly complex ML models in security-oriented pattern recognition tasks presents a problem.

That problem is evaluating the true merit of these newly proposed complex models on datasets where the margins at the top are very narrow. While model authors claim the superiority of their novel models, the reality is that it is not possible to rank-order them effectively because extremely simple ML models often score equally well.

This article's chief contribution is the continued affirmation of the observation made by (Holte, 1993). Within data-based cybersecurity, even the simplest ML models perform well or even perfectly on the available datasets. This observation from experiment severely undercuts novel ML models' claims that they significantly improve detection. To the authors' knowledge, the concern of increased model complexity without commensurate gains in classification performance has been raised by researchers in the field, but has not yet been experimentally validated. Our motivation for this work stems from both our search for the smallest effective models for high-throughput low-latency intrusion detection systems and from our prior work that established that well-known models still reach outstanding classification scores even under severe data restrictions (D'hooge et al., 2021).

This article presents true baselines for 17 security datasets obtained with a model at the lower bound of ML model complexity, one rule (OneR, 1R). In many cases its performance sits remarkably close to that of most recently proposed classification methods.

The article is structured as per the template. In section 2 the literature on two broad domains of pattern recognition in cybersecurity is outlined with a particular focus on model innovation. Those two domains are network traffic classification (primarily intrusion detection) and malware detection. The methodology is described in section 3 and focuses on the models OneR and ensemble OneR. The results are conveyed per dataset in chronological order in section 4. The discussion aggregates the results and presents the insights and recommendations to improve new detection algorithm proposals and to improve the datasets themselves. The conclusion 6 and future work 7 round out the article.

[a] https://orcid.org/0000-0001-5086-6361
[b] https://orcid.org/0000-0002-1781-900X
[c] https://orcid.org/0000-0003-2618-3311
[d] https://orcid.org/0000-0003-4824-1199
[e] https://orcid.org/0000-0003-0575-5894

D'hooge, L., Verkerken, M., Wauters, T., De Turck, F. and Volckaert, B.
Castles Built on Sand: Observations from Classifying Academic Cybersecurity Datasets with Minimalist Methods.
DOI: 10.5220/0011853300003482
In *Proceedings of the 8th International Conference on Internet of Things, Big Data and Security (IoTBDS 2023)*, pages 61-72
ISBN: 978-989-758-643-9; ISSN: 2184-4976

2 RELATED WORK

In both security-oriented network traffic classification and malware classification, academic research had already embraced machine learning as the method of choice for further investigation by the 2010s. (Buczak and Guven, 2015) and (Mishra et al., 2018) made this observation the central theme of their landmark surveys in intrusion detection. Across the entire spectrum of classical data mining and machine learning methods, they were able to compile and compare dozens of proposed algorithms. A bibliographical review of malware research by (Ab Razak et al., 2016) equally noted the rising prevalence of machine learning in malware recognition. (Shalaginov et al., 2018) noted that static analysis was quickly getting infused with ML methods to cope with the explosive growth in the volume and variations of malware.

The aforementioned (older) surveys focus mostly on classical ML methods, but a more recent survey by (Ahmad et al., 2021) explicitly compares the amount of new proposals with classical methods to new deep learning approaches in intrusion detection. In their set of recently proposed methods, they find that 60% are now pure deep learning methods, 20% are hybrids between deep learning and classical methods and the remaining 20% continue to innovate purely on classical methods. Two years prior, (Liu and Lang, 2019) selected 26 proposals in intrusion detection, 14 of which relied on deep learning. In the challenges and future directions, the authors already identified the low efficiency of the ever more complicated methods as a growing issue to be solved.

Within the suite of available deep architectures, auto-encoders (AE), convolutional neural networks (CNN), recurrent neural networks (RNN), deep belief networks (DBN) and restricted Boltzmann machines (RBM) were the most widely studied before 2020 (Berman et al., 2019), (Aldweesh et al., 2020). The same architectures enjoyed the greatest popularity in malware detection during the same period (Naway and Li, 2018), (Qiu et al., 2020).

Due to the speed with which new architectures are proposed, tested and adopted in more well-known applications of ML (e.g. vision and language modelling tasks), they are adapted equally rapidly by data-based security researchers. Large models based on the transformer architecture, originally designed for language modelling and quickly transmuted for vision tasks, are the latest inspiration for methods like MalBERT (Rahali and Akhloufi, 2021) and DDoSTC (Wang and Li, 2021).

The overall trend in the surveys and the latest methods is clear. Ever more complicated detection algorithms are proposed with performance gains indistinguishable from run-to-run variance. Occasionally model authors compare their new proposal to simpler ML methods (very often to a stock random forest). The impulse is commendable, but the execution is too rushed. More adequate comparisons would include XGBoost or Catboost and spend the same amount of time and resources to optimize them as well as the newly proposed algorithm. The step down to a simpler model rarely occurs and even so, this article will demonstrate that another model with a tiny fraction of the complexity of XGBoost or Catboost still performs good or even great on the same academic cybersecurity datasets.

This literature section constrains itself to high-quality reviews which surveyed hundreds of individual methods. 17 datasets were evaluated in this article and due to the limited space, it was not possible to provide individual examples for each dataset. For those readers who are not familiar with the domain, the reviews capture many of the included datasets and do detailed reporting of the achievable performance. Similarly, the dataset authors / publishers often included baselines (relevant citations in table 1).

2.1 Included Datasets

Seventeen academic datasets have been evaluated in this analysis. Broadly, they fall in two categories: traffic classification and malware classification. The traffic classification datasets primarily include (multi-class) intrusion detection datasets as well as some specialty datasets. The set of malware datasets is not as expansive, but stills cover Android, Windows and Linux malware as well as one specialized dataset on malware delivery through malicious PDF files. Table 1 presents an overview of the included datasets with their relevant citation(s) and year of publication. The author maintains clean versions of every dataset on Kaggle updated in accordance with the latest research. All computation, preprocessing and analysis are publicly available at https://www.kaggle.com/dhoogla/datasets.

3 METHODOLOGY

The methodology section introduces the two algorithms which have been used throughout the analysis. Their simplicity is a deliberate design choice informed by (Holte, 1993) meant to highlight the ease with which many of the state-of-the-art ML security datasets can be classified.

Table 1: The Included ML-Focused Security Datasets.

Name	Year	Purpose
Traffic Classification Datasets		
NSL-KDD	2009 (Tavallaee et al., 2009)	Multi-class intrusion detection
CTU-13	2014 (García et al., 2014)	Botnet detection
UNSW-NB15	2015 (Moustafa and Slay, 2015)	Multi-class intrusion detection
CIDDS-001	2017 (Ring et al., 2017a)	Multi-class intrusion detection
CIDDS-002	2017 (Ring et al., 2017b)	Port scanning detection
CIC-NIDS Collection		
CIC-IDS	2017 (Sharafaldin. et al., 2018)	Multi-class intrusion detection
CIC-DoS	2017 (Jazi et al., 2017)	DoS detection
CSE-CIC-IDS	2018 (Sharafaldin. et al., 2018)	Multi-class intrusion detection
CIC-DDoS	2019 (Sharafaldin et al., 2019)	DDoS detection
CIC-Darknet	2020 (Habibi Lashkari et al., 2020)	VPN & Tor detection
CIRA-CIC-DoHBrw	2020 (MontazeriShatoori et al., 2020)	DNS covert channel detection
CIC-Bell-DNS-EXF	2021 (Mahdavifar et al., 2021)	DNS data exfil detection
USB-IDS-1	2021 (Catillo et al., 2021)	DoS detection
Distrinet-CIC-IDS	2021 (Engelen et al., 2021)	Corrected issues in CIC-IDS2017
Malware Classification Datasets		
CCCS-CIC-AndMal (Rahali et al., 2020)	2020 (Keyes et al., 2021)	Android malware
CIC-Malmem	2022 (Carrier et al., 2022)	Obfuscated malware classification
CIC-Evasive-PDFMal	2022 (Issakhani et al., 2022)	Hidden malware in PDF files

3.1 The Simplest ML Model: One Rule

If you're allowed only one comparison, which feature and which value would you pick? That's the central question of the One Rule (OneR) model. Implementation-wise it is identical to a decision tree model with just a root node. This article uses OneR exclusively, but keeps a OneR model for every feature in each dataset, rather than keeping only one. Figure 1 visually demonstrates the model.

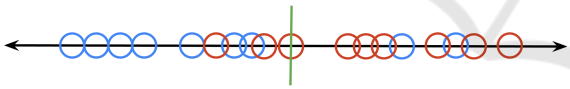

Figure 1: OneR: one optimal split point on a number line.

The hard point splits that are found for each feature will either have some predictive power if a disproportionate amount of samples from one class fall below (or above) the split point. No such predictive power will reveal itself for two possible reasons and two edge cases.

1. there is an equal proportion of the classes on each side of the split point
2. enough negative samples sit on either side of the value range for the feature for the positive samples

The edge case for reason 1 is that all samples' values for the feature fall on the split point (i.e. a feature with zero variability). Such features should be removed by the dataset authors prior to publication. Reason 2 does not need to be a problem if you are allowed more than 1 decision. Its edge case is that there is full overlap between the observed values for a feature for all negative and all positive samples with a majority of negative samples at each value.

3.1.1 The Simplest Ensemble

A single OneR model disregards all features but one. This is extremely restrictive. We have opted to also include an ensemble method with threshold optimization. The simplest ensemble takes the outputs of each OneR model with any predictive power and averages them. These average outputs per sample then serve as the final predictions. In general, an uplift in classification performance is observed for the ensemble because it combines the predictions of independent models (so long as a majority of the independent classifiers predicts the right outcome).

However, partly because an unweighted averaging was used, it is possible to see a loss in classification performance. If there is a preponderance of models with weak classification performance then they will drag the stronger models down. A change in the weight contribution of the individual OneR models tied to their individual performance would alleviate this.

One final trick which was employed is the use of Youden's J statistic (Youden, 1950). J is calculated as the true positive rate minus the false positive rate and allows setting the optimal threshold to assign predictions to a class (in a binary context). In the implementation, the maximum J and subsequent optimal threshold are calculated based on the predictions for the training set and the same threshold is re-used for the test set to avoid leaking information.

3.2 Sampling

Contrary to the prevailing wisdom and the literature, an 80-20 split is not required at all to train performant models. The loss in performance from inverting the standard split to its complement 20% train 80% test split hardly affects performance. For the datasets with millions of samples even more aggressive sampling methods (i.e. 1-99 or even 0.1-99.9 train-test) remain effective with negligible losses in performance (https://www.kaggle.com/code/dhoogla/cic-ids2017-03-minimal-data-binary). For the datasets with pre-designated splits (a.o. NSL-KDD, UNSW-NB15, USB-IDS-1), those splits were kept to keep comparability with the state-of-the-art even though more aggressive sampling methods work equally well.

3.3 Common Dataset Preprocessing

All datasets are provided as comma-separated-value files (CSV). CSVs are readable as text, but not efficient compute or storage wise. Every dataset underwent at least the following preprocessing steps.

- Dropping metadata shortcut features (D'hooge et al., 2022a) and other contaminant features (D'hooge et al., 2022b) (only for NSL-KDD, UNSW-NB15 and the CIC-NIDS collection)
- Downsizing the features into appropriate types
- Removing samples with missing or corrupt values
- Removing duplicate samples to avoid inflating classification scores

3.4 A Note on Classification Resolution

Throughout this analysis, the choice was made to work with binary classification. For most datasets this is the most logical level because it coincides with the labeling by the dataset authors. For the CIC collection which has massive differences in sample size for its available classes, classification was done on each available attack class versus benign traffic. True multi-class evaluation was eschewed because it would obscure the class imbalance and because the model is too simplistic.

4 RESULTS

The results are presented in two big blocks with the network traffic datasets first and the malware datasets second. Additional preprocessing (if any) is mentioned, the top-5 features are listed and the performance of the unweighted ensemble of the successful OneR models is presented. To keep the article maximally self-contained, the results are written in a dense format. We recommend reading table 2 first which presents a summary of the numerical results and then circling back to the detailed results for the individual datasets.

4.1 Network Traffic Classification Datasets

The results on the fourteen network traffic classification datasets are presented first and in chronological order of dataset publication. Even though some datasets are outdated by now and superseded by more recent entries, certain design aspects of the older datasets positively influence their usability and legitimately decrease OneR's performance (a detailed explanation is part of section 5).

4.1.1 NSL-KDD

NSL-KDD is included because it has been the most studied NIDS dataset to date. Although its attacks are no longer reflective of a modern network landscape, the original dataset authors (KDD98/99) made sound decisions to produce a varied dataset with separate train and test sets with non-overlapping sets of cyber attacks. For this evaluation, the given train-test split was respected.

Even though NSL-KDD is already a rework of the KDD99 dataset, more issues with contaminating features have recently been found (D'hooge et al., 2022b). Those features have been removed and the highest quality version of NSL-KDD is publicly available https://www.kaggle.com/dhoogla/datasets/nslkdd.

OneR models are just moderately capable on this dataset with the top-5 features all scoring between 0.745 and 0.819 AUROC (scr-bytes, dst-bytes, flag, service, dst-host-same-srv-rate). OneR models for 22 out of the 36 features had at least some predictive power ($AUROC > 0.5$) and their unweighted ensemble scored 0.929 AUROC (precision 0.94 and recall 0.86).

4.1.2 CTU-13

After removing contaminating metadata features which act as shortcuts for ML models, the botnet dataset CTU-13 only has 9 features. Worse still, the observed values for benign and malicious samples

overlap nearly perfectly. None of the OneR models yield any results better than random guessing.

4.1.3 UNSW-NB15

UNSW-NB15 has pre-designated train and test sets, so those were respected in the analysis. Nearly all published articles that use this dataset are unaware that there are contaminating features, but for this analysis, the improved version of the dataset is used.

With the corrected UNSW-NB15 dataset OneR models remain capable with the top-5 features being rate, sload, dbytes, dpkts and dmean each scoring above .766 AUROC (max 0.781). Compared to the full set of features (contaminating ones included) a flat loss of 0.1 AUROC has been observed. The features stll and ct-state-ttl used to be the top-2 performers with AUROCs above 0.85. For this dataset, the first instance of the ensemble underperforming an individual feature is observed as the ensemble reaches just 0.776 AUROC with precision at 0.96 and recall at 0.62.

4.1.4 CIDDS-001

After the removal of contaminating metadata features, CIDDS-001 has 12 features, 6 of which are one-hot-encoded versions of TCP flags. With the additional imbalance in benign versus malicious traffic (4,354,282-11875) and full overlap in values and value ranges for the malicious traffic, OneR models are not sophisticated enough to create useful distinctions.

4.1.5 CIDDS-002

Even though CIDDS-002 has the same minuscule set of features and a worse class balance (benign 2,640,306 - malicious 3829) compared to CIDDS-001, a few OneR models prove capable. During the construction of CIDDS-002 only port scan attacks were gathered.

Three features demonstrate splitting power with a single comparison each. The observed values for proto (0.833 AUROC), tos (0.782) and bytes (0.598) have overlap between the benign and malicious samples. However, for the useful features, the malicious samples have a majority on the edges of the range, granting them their predictive power. The ensemble OneR model reaches an AUROC of 0.878 with 78% precision and 76% recall.

4.1.6 CIC-NIDS-Collection

Rather than evaluating the individual CIC-NIDS datasets, the analysis has been conducted on the global aggregate dataset of the NIDS collection. Working on the collated version should improve the variability due to the large increase in the number of represented attacks.

Additionally, the CIC collection as used takes (D'hooge et al., 2022a) and (D'hooge et al., 2022b) into account, removing all shortcut metadata features and all content features which have been identified as contaminants. In total the dataset has more than 9 million unique samples with 58 statistical features. For academic researchers, it is the most comprehensive, labeled NIDS dataset available.

The results will be discussed per represented attack class. The analysis was done at this level of resolution because there are millions of DDoS and DoS samples, whereas Portscan is represented by a mere 2255 (positive) samples. Collapsing all attack classes into one dataset would skew the results to a DDoS / DoS model.

DDoS: top-5 features: total-backward-packets, bwd-packets/s, avg-fwd-segment-size, fwd-packet-length-mean and fwd-packet-length-max (AUROCs between 0.702 and 0.728). 27/57 features contributed to the ensemble OneR model which reaches 0.842 AUROC with 0.655 precision and 0.8653 recall. Having the combination of the DDoS samples of three datasets (IDS17, IDS18 and DDoS19) definitely improved the variability in the data (e.g. when compared to 4.1.11 DDoS).

DoS: top-5 features: packet-length-variance, bwd-packet-length-std, flow-bytes/s, fwd-seg-size-min, bwd-packets/s (AUROCs between 0.701 and 0.817). 39/57 OneR models contribute to the ensemble which reaches 0.905 AUROC with 0.820 precision and 0.811 recall. Again, having the DoS samples of three NIDS datasets (IDS17, DoS17 and IDS18) increased the classification difficulty, especially for a model such as OneR.

Bruteforce: top-5 features: subflow-fwd-bytes, fwd-packets-length-total, fwd-header-length, fwd-act-data-packets, bwd-header-length (AUROCs between 0.951 and 0.964). 40/57 OneR models contribute to the ensemble which reaches 0.973 AUROC with 0.979 precision and 0.936 recall.

Portscan: top-5 features: bwd-packet-length-mean, bwd-packets-length-total, subflow-bwd-bytes, packet-length-variance and packet-length-std (AUROCs between 0.849 and 0.865). 34/57 OneR models contribute to the ensemble which reaches 0.926 AUROC with 0.692 precision and 0.925 recall.

Botnet: top-5 features: flow-bytes/s, flow-iat-mean, avg-bwd-segment-size, bwd-packet-length-mean and bwd-packets/s (AUROCs between 0.852 and 0.870). 38/57 OneR models contribute to the ensemble which

reaches 0.9793 AUROC with 0.996 precision and 0.948 recall.

Webattack: top-5 features: avg-packet-size, packet-length-max, flow-bytes/s, packet-length-variance and packet-length-std (AUROCs between 0.801 and 0.804). 40/57 OneR models contribute to the ensemble which reaches 0.886 AUROC with 0.728 precision and 0.796 recall.

Infiltration: is represented in both CIC-IDS datasets, but only with 30 samples in CIC-IDS2017. CSE-CIC-IDS2018 expanded the infiltration attacks, but considers the traffic from downloading a malicious file from Dropbox as part of the malicious traffic, heavily padding the sample count. The quality of the experiment to gather this attack class has not been sufficient. As a consequence no models perform well on this attack class. For completeness: the ensemble OneR model relied on positive contributions from just 4 features and reached an AUROC of 0.57.

On the individual data sets of the CIC-NIDS collection results are more in-line with the results on the updated Distrinet-CIC-IDS2017 (subsection 4.1.11). The grouping into one larger, more varied dataset and the removal of contaminating features has reduced the effectiveness of the OneR model.

4.1.7 CIC-Darknet2020

The CIC Darknet dataset from 2020 has two components. The dataset is designed around VPN and Tor recognition and provides this labeling resolution to its users (Tor/NonTor & VPN/NonPN).

On the Tor/NonTor task, the top-5 features were bwd header length, total bwd packets, ack flag count, fwd init win bytes and bwd init win bytes with AUROCs ranging from 0.753 up to 0.898. The ensemble model reaches 0.980 AUROC with an F1 score of 0.732. A middling performance on the positive class but high overall AUROC indicates that the model predicts very few false positives.

The VPN/NonVPN task had significantly worse performance. Its top-5 features do not reach higher than 0.719 AUROC. (subflow bwd bytes, bwd packet length max, total length of bwd packet, bwd packet length mean and bwd segment size avg). The ensemble with contributions by 53/77 features improves slightly to 0.779 AUROC with an F1-score of 0.739, indicating a pretty mixed performance in terms of type I and type II errors.

4.1.8 CIRA-CIC-DoHBRW2020

CIRA-CIC-DoHBrw provides two layers of tasks. L1 focuses on the split between DoH and NonDoH traffic while L2 focuses on the recognition of DoH tunnel abuse for exfiltration / communication purposes for malware as opposed to legitimate use.

On the L1-task, the top-5 features are packet-time-mean, packet-time-stdev, packet-time-var, duration and packet-length-mode (AUROCs between 0.767 and 0.856). The ensemble model reaches 0.903 AUROC with 60% precision and 82% recall. Nine of the twenty-nine features contributed to the ensemble.

On the L2-task, the top-5 features are packet-time-stdev, packet-time-var, flow-sent-rate, flow-bytes-sent and flow-bytes-received (AUROCs between 0.604 and 0.698). The ensemble model reaches an AUROC of 0.799 with a precision-recall pair of 0.971 and 0.972. The ensemble took the predictions of 13/29 models into account.

4.1.9 CIC-Bell-DNSExf2021

CIC-Bell-DNSExf2021 is the second CIC dataset targeting covert communication and data exfiltration over DNS. Two versions of the cleaning process for this dataset have been performed. The first tries to stay as close to the data as possible. The second version is much more opinionated and focuses on getting the data as close to its own documentation as possible. For this analysis, the minimalistically cleaned files were chosen because they deviate the least from the raw dataset.

Individual OneR models are relatively capable of discerning the malicious from the benign DNS traffic. The top-5 features are FQDN-count, subdomain-length, special, sld and longest-word (AUROCs from 0.784-0.796). The ensemble model reaches 0.846 AUROC with an F1 score of 0.862. OneR models for 24/41 features contributed to the ensemble.

4.1.10 USB-IDS-1

USB-IDS-1 focuses on DoS attacks and their potential defenses. Having captured millions of samples of four DoS attacks under various defensive circumstances, it was expected that there would be a great deal of variety in the samples. The compatibility with the feature set of the CIC collection also opens up further investigations into generalization.

The expectation of a varied dataset, impossible to be classified by a model as simple as a single comparison, did not hold up. The top-5 features are bwd bytes/bulk avg, total length of bwd packet, fwd iat min, bwd psh flags and fwd init win bytes, all with AUROCs above 0.95. A single comparison with the top feature fwd init win bytes produces a model with 0.999 AUROC. 46/77 features have immediate separating power. There is no need for the ensemble model, but for completeness' sake, its performance

was 0.997 AUROC with 0.982 F1 score.

4.1.11 Distrinet-CIC-IDS2017

The Distrinet update of CIC-IDS2017 offers the same attack classes as the CIC collection. Even though this is a single dataset, the distinction will be kept because the number of available samples for each of the classes varies greatly.

DDoS: top-5 features: bwd-packet-length-mean, avg-bwd-segment-size, packet-length-variance, packet-length-std and bwd-packet-length-std (AUROCs from 0.993-0.998). 60/77 OneR models contribute to the ensemble which reaches 0.999 AUROC with 0.995 precision and 0.979 recall.

DoS: top-5 features: fwd-iat-max, bwd-iat-mean, flow-duration, fwd-iat-total and fwd-seg-size-min (AUROCs from 0.858-0.899). 64/77 OneR models contribute to the ensemble which reaches 0.965 AUROC with 0.906 precision and 0.823 recall.

Bruteforce: top-5 features: bwd-header-length, bwd-psh-flags, psh-flag-count, fwd-psh-flags and down/up-ratio (AUROCs from 0.951-0.975). 53/77 contribute to the ensemble which reaches 0.998 AUROC with 0.998 precision and 0.987 recall.

Portscan: top-5 features: fwd-packet-length-max, fwd-packets-length-total, subflow-fwd-bytes, fwd-packet-length-mean and avg-fwd-segment-size (AUROCs from 0.951-0.955). 45/77 features contribute to the ensemble which reaches 0.993 AUROC with 0.878 precision and 0.983 recall.

Botnet: top-5 features: packet-length-mean, avg-packet-size, subflow-bwd-bytes, avg-bwd-segment-size and bwd-packet-length-mean (AUROCs from 0.910-0.950). 53/77 features contribute to the ensemble which reaches perfect 1.0 AUROC with 0.998 precision and perfect recall.

Webattack: top-5 features: bwd-header-length, bwd-iat-std, init-bwd-win-bytes, init-fwd-win-bytes, fwd-seg-size-min (AUROCs from 0.844-0.894). 56/77 features contribute to the ensemble which reaches 0.969 AUROC with 0.895 precision and 0.855 recall.

Even though the Distrinet lab materially improved CIC-IDS2017, it remains a trivial dataset to classify.

4.2 Malware Classification Datasets

Although far fewer malware classification datasets have been included, the results on them highlight the same issue.

4.2.1 CCCS-CIC-AndMal2020

This mixture of around 400k samples of android applications (half of which are malware) has 14 categories of malware including adware, backdoors, spyware, trojans, scareware, ... It has features captured both through static and through dynamic analysis. For the benign software samples, the 150+ dynamic features were not captured, leaving *only* the 9500+ static features. The static features were not individually named in the dataset so they will be represented simply by their index (Fxxxx).

The top-5 features were F50, F57, F37, F48, F58 all with AUROC scores above 0.84 (max 0.875). Of the 9503 features less than 500 had direct separating power. The ensemble OneR model reaches 0.953 AUROC with 0.953 precision and 0.869 recall.

4.2.2 CIC-MalMem2022

The second malware dataset in this article has been designed for the detection of obfuscated malware with features derived from memory dumps. It is a tabular dataset, horizontally with 55 unique features are provided and vertically it has 58,596 records perfectly balanced with 29,298 benign and 29,298 malicious samples.

Our additional preprocessing was limited to adding intermediate levels of labeling. Only the most abstract level, the binary split between malicious and benign samples is taken into account.

The top-5 features were svcscan.nservices, svcscan.shared-process-services, svcscan.kernel-drivers, handles.nmutant and dlllist.avg-dlls-per-proc, all landing with AUROC scores above 0.987. Of the 55 available features, 51 can yield single comparison models with separating power beyond 0.5 AUROC. The ensemble model of the useful single feature models reached 0.996 AUROC and 0.984 F1 (0.974 precision, 0.993 recall).

4.2.3 CIC-Evasive-PDFMal2022

The third and final malware dataset centers on malware hidden in PDF files. It is almost balanced with 5555 malicious samples and 4468 benign samples.

Our preprocessing for this dataset was quite extensive because two classes of data inconsistencies exist in CIC-Evasive-PDFMal2022. First, there are features where a negative value is impossible based on the documentation, but negative values occur. Second, there are features which should be numeric, but some samples have string or other non-numeric values.

The top-5 single-decision, single feature models were built on startxref, metadatasize, javascript, js and stream, all with AUROCs above 0.814. Luckily, unlike CIC-MalMem2022 it is not possible to solve the dataset well with one feature and one comparison.

For 20 of the 31 available features, single comparison models have (at least some) effectiveness.

The ensemble of the 20 contributing OneR models reaches an AUROC of 0.986 with an F1 score of 0.9539 (precision 0.969, recall 0.939).

5 DISCUSSION

The result section 4 is extremely dense, but the core thread is clearly visible. Even with a computational budget of one comparison per feature, good to excellent models can be found. Table 2 summarizes the results on each dataset. The table only mentions the performance of the ensemble of OneR models, since it's more often the better model. Still for every entry with a †, an individual OneR model had the best performance. Figure 2 visualizes a blatant example of lacking variability which leads to OneR being so powerful.

Table 2: Result Summary.

Traffic Classification Datasets			
Dataset	AUROC	Precision	Recall
NSL-KDD	0.93	0.94	0.86
CTU-13	0.5	0.5	0.5
UNSW-NB15†	0.78	0.96	0.62
CIDDS-001	0.5	0.5	0.5
CIDDS-002	0.88	0.78	0.76
CIC-NIDS Collection			
DDoS	0.84	0.66	0.87
DoS	0.91	0.82	0.81
Bruteforce	0.97	0.98	0.94
Portscan	0.93	0.69	0.92
Botnet	0.98	1.0	0.95
Webattack	0.89	0.73	0.80
CIC-Darknet-Tor	0.98	0.59	0.97
CIC-Darknet-VPN	0.78	0.63	0.90
CIRA-CIC-DoHBrw-L1	0.90	0.60	0.82
CIRA-CIC-DoHBrw-L2	0.80	0.97	0.97
CIC-Bell-DNS-EXF	0.85	0.80	0.94
USB-IDS-1†	1.0	1.0	0.96
Distrinet-improved-CIC-IDS2017			
DDoS	1.0	1.0	0.98
DoS	0.97	0.91	0.82
Bruteforce	1.0	1.0	0.99
Portscan	1.0	1.0	1.0
Botnet	1.0	1.0	1.0
Webattack	0.97	0.90	0.86
Malware Classification Datasets			
CCCS-CIC-AndMal	0.95	0.95	0.87
CIC-Malmem†	1.0	0.97	0.99
CIC-Evasive-PDFMal	0.99	0.97	0.94

Three additional observations present themselves:

1. The CIC-NIDS collection does introduce more variability, not just additional volume which leads to weaker performance for the OneR model
2. Datasets with intelligently predefined train-validation-test splits are less affected
3. More malware datasets have to be examined since they were amongst the most affected

The first observation leads to a hopeful conclusion for future work. Datasets should strive towards interoperability with the others to quickly achieve higher variability, not just higher volume. The differences in experimental setup and execution will lead to new, unique samples. A comparison of figures 3 (Distrinet-IDS-2017-DDoS) and 4 (CIC-NIDS-Collection-DDoS) immediately reveals how the difference in distributions for the same features leads OneR to score so well on the former and a lot poorer on the latter.

NSL-KDD and UNSW-NB15, even though they are quite a bit older than the latest datasets, have a feature that protects them from easily finding models with inflated performance. Creating designated train, validation and test splits that are no mere consequence of random sampling, but contain unseen attacks from the same classes guarantees new patterns to more rigorously test generalization beyond the training.

Some datasets are not well-suited to ML evaluation despite being created for that purpose. Both CTU-13 and CIDDS-001 have so few features and so much overlap in the available values for those features between benign and malicious samples that no method, however sophisticated, will reach adequate performance. Both datasets are also extremely class-imbalanced (CIDDS: benign 4,354,282, malicious 11,875, CTU-13: benign 8,432,312, malicious 190,210). A visual explanation of the issue is visible in figure 5. That visualization also reveals a shortcoming in the experiment's design. All attacks were short in duration and therefore did not generate a large amount of traffic (bytes). Worse still is the distribution of the third feature "flows" which has 0 variance because it has only one potential value (1.0).

Since OneR is the simplest model that can be devised, it is not always a genuine contender compared to recently proposed models. The gap in complexity between OneR and the state-of-the-art proposals is hard to overstate. Even if method authors do not compare their proposal to OneR, they should compare it to XGBoost (Chen et al., 2015), Catboost (Prokhorenkova et al., 2018) or even just to randomized decision trees (Geurts et al., 2006). For the included datasets which cover the state-of-the-art in NIDS and a fair portion of malware classification, method authors would discover that their proposals do not outperform simpler, computationally less expensive models. This rings particularly true for the slew

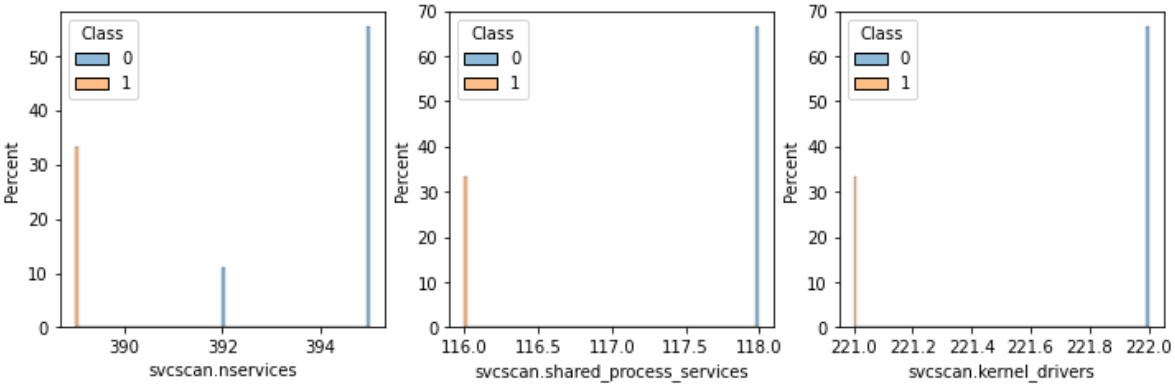

Figure 2: Lackluster variation in the data creates OneR's excellent performance on CIC-Malmem2022.

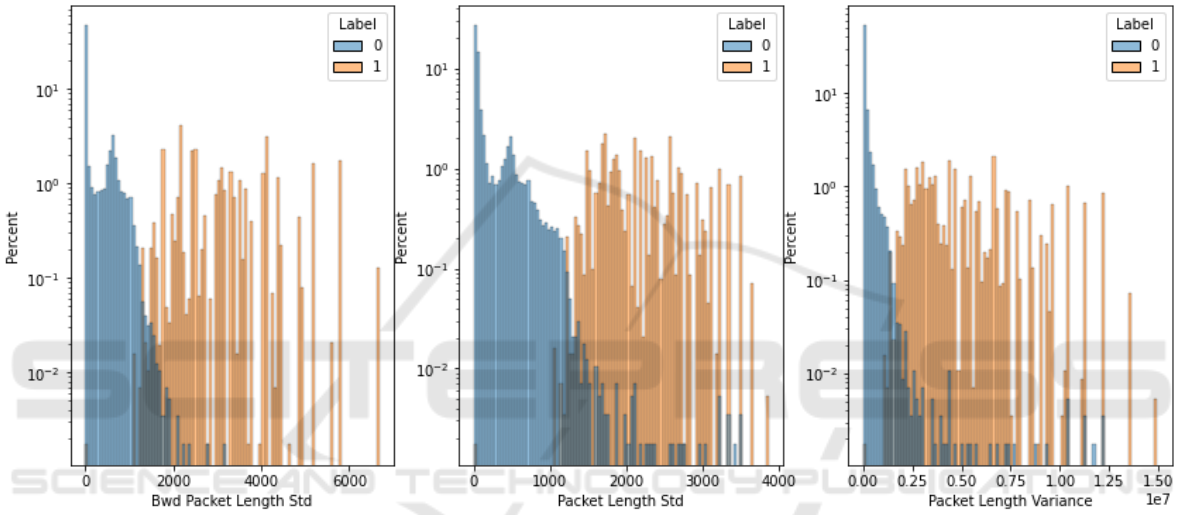

Figure 3: The class-grouped distributions of Distrinet-DDoS demonstrate why it is so easily classified.

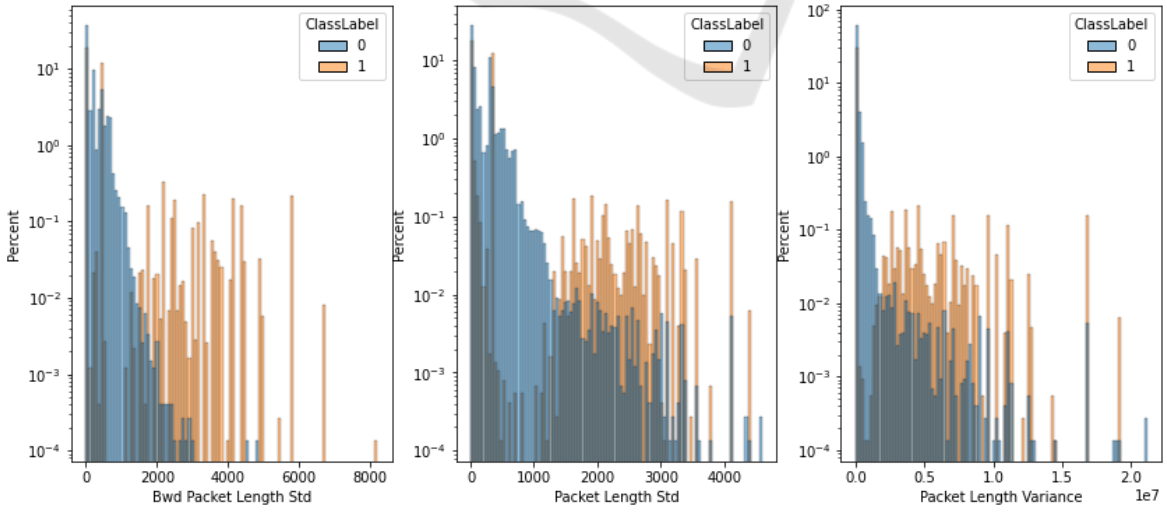

Figure 4: The class grouped distributions of CIC-NIDS collection DDoS show significantly more variability in its samples.

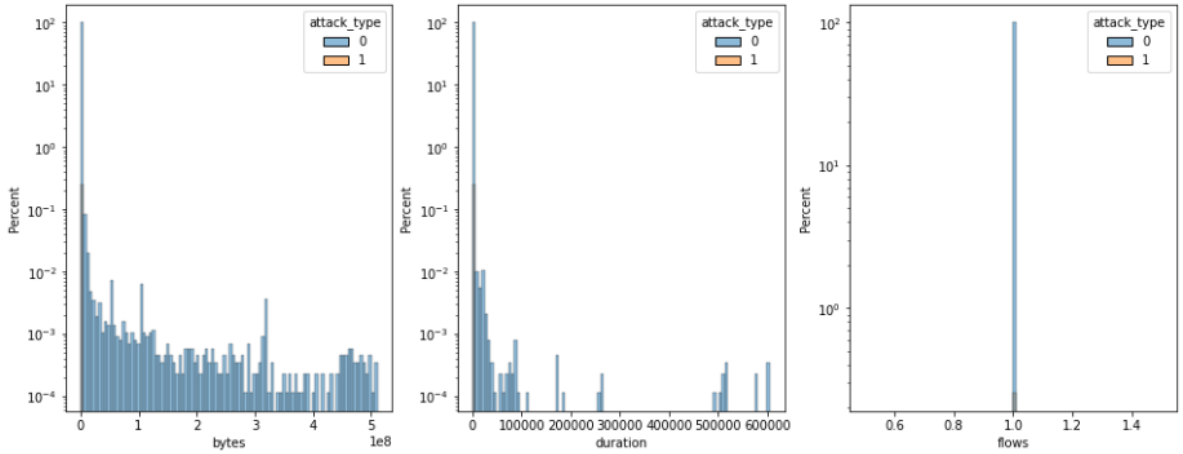

Figure 5: CIDDS-001: Heavy class imbalance and complete overlap in the value ranges for the benign and malicious samples or no variation at all.

of deep neural networks that have been proposed in recent years.

6 CONCLUSION

Cybersecurity research has ventured far into machine learning for its pattern recognition tasks. New model proposals are numerous and often borrow / recombine model advances discovered for other ML tasks. Along the way it has lost its commitment to delivering models with excellent performance while keeping computational cost low. Survey authors had already noticed that the optimization for ever higher classification metrics resulted in growing model complexity without keeping in mind that many security-related pattern recognition tasks are time-sensitive and benefit from a dual optimization of model complexity and computational efficiency. Their objections have not dissuaded the research community from continuing to increase model complexity even though the margins for improvement on many state-of-the-art datasets are slim to non-existent.

This article is inspired by (Holte, 1993) which demonstrated that the simplest supervised machine learning model, one rule (OneR, 1R) performed well on the most commonly used datasets of its time. On 17 state-of-the-art cybersecurity datasets for network traffic classification and malware recognition, the effectiveness of OneR and ensemble OneR is established. Despite being nothing more than a single comparison, OneR (or ensemble OneR) often proves itself competitive with recent detection proposals. Further investigation into the datasets themselves has revealed that many suffer from a lack of variability which allows even OneR to be effective.

From a practical standpoint, the authors of this article want to urge authors of new detection proposals to compare their work to a well-tuned XGBoost or Catboost model before claiming superiority for their model. Researchers working on any dataset should also be aware of OneR's performance even if they don't explicitly mention it in their proposals. Ultimately, the authors of this article hope that this practice will lead more researchers to investigate the datasets themselves, to uncover their flaws and dedicate their future work to improving dataset quality so that the more complex models will actually become necessary. As it stands today, the complexity of many novel detection methods is not warranted for the available datasets.

7 FUTURE WORK

For the network traffic datasets this article should clearly convey the issue, but the sampling of malware classification datasets is too sparse to carry the same weight. Future work (primarily on Kaggle) will continue to include more security datasets (especially in malware classification) to establish baselines. Cybersecurity datasets have to improve, particularly when it comes to variety. Future work will focus on this data generation problem with particular interest in dataset compatibility.

In network intrusion detection datasets another fundamental problem exists that may be tied to the lack of variability in the datasets. Proposed methods yield excellent results in the standard intra-dataset evaluation (the same dataset for training, validation and testing), but they fail to generalize to inter-dataset

evaluation, even if the second dataset has been collected under very similar circumstances.

ACKNOWLEDGEMENTS

This work has partially been supported by the AIDE project which is a three-way partnership between government, academia and industry. It aims to achieve robust, resilient and adaptive protection of computer systems with a particular role for federated learning. The AIDE project is funded by the Belgian Chancellery of the Prime Minister, a federal public service, as part of their financing for the development of artificial intelligence.

REFERENCES

Ab Razak, M. F., Anuar, N. B., Salleh, R., and Firdaus, A. (2016). The rise of "malware": Bibliometric analysis of malware study. *Journal of Network and Computer Applications*, 75:58–76.

Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., and Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, 32(1):e4150.

Aldweesh, A., Derhab, A., and Emam, A. Z. (2020). Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowledge-Based Systems*, 189:105124.

Berman, D. S., Buczak, A. L., Chavis, J. S., and Corbett, C. L. (2019). A survey of deep learning methods for cyber security. *Information*, 10(4):122.

Buczak, A. L. and Guven, E. (2015). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications surveys & tutorials*, 18(2):1153–1176.

Carrier, T., Victor, P., Tekeoglu, A., and Lashkari, A. H. (2022). Detecting obfuscated malware using memory feature engineering. In *ICISSP*, pages 177–188.

Catillo, M., Del Vecchio, A., Ocone, L., Pecchia, A., and Villano, U. (2021). Usb-ids-1: a public multilayer dataset of labeled network flows for ids evaluation. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pages 1–6. IEEE.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., et al. (2015). Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4):1–4.

D'hooge, L., Verkerken, M., Volckaert, B., Wauters, T., and De Turck, F. (2022a). Establishing the contaminating effect of metadata feature inclusion in machine-learned network intrusion detection models. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 23–41. Springer.

D'hooge, L., Verkerken, M., Wauters, T., Volckaert, B., and De Turck, F. (2021). Hierarchical feature block ranking for data-efficient intrusion detection modeling. *Computer Networks*, 201:108613.

D'hooge, L., Verkerken, M., Wauters, T., Volckaert, B., and De Turck, F. (2022b). Discovering non-metadata contaminant features in intrusion detection datasets. In *2022 19th Annual International Conference on Privacy, Security & Trust (PST)*, pages 1–11. Ieee.

Engelen, G., Rimmer, V., and Joosen, W. (2021). Troubleshooting an intrusion detection dataset: the cicids2017 case study. In *2021 IEEE Security and Privacy Workshops (SPW)*, pages 7–12. IEEE.

García, S., Grill, M., Stiborek, J., and Zunino, A. (2014). An empirical comparison of botnet detection methods. *Computers & Security*, 45:100–123.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine learning*, 63(1):3–42.

Habibi Lashkari, A., Kaur, G., and Rahali, A. (2020). Didarknet: a contemporary approach to detect and characterize the darknet traffic using deep image learning. In *2020 the 10th International Conference on Communication and Network Security*, pages 1–13.

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11(1):63–90.

Issakhani, M., Victor, P., Tekeoglu, A., and Lashkari, A. (2022). Pdf malware detection based on stacking learning. In *Proceedings of the 8th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP,*, pages 562–570. INSTICC, SciTePress.

Jazi, H. H., Gonzalez, H., Stakhanova, N., and Ghorbani, A. A. (2017). Detecting http-based application layer dos attacks on web servers in the presence of sampling. *Computer Networks*, 121:25–36.

Keyes, D. S., Li, B., Kaur, G., Lashkari, A. H., Gagnon, F., and Massicotte, F. (2021). Entroplyzer: Android malware classification and characterization using entropy analysis of dynamic characteristics. In *2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS)*, pages 1–12. IEEE.

Liu, H. and Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. *applied sciences*, 9(20):4396.

Mahdavifar, S., Hanafy Salem, A., Victor, P., Razavi, A. H., Garzon, M., Hellberg, N., and Lashkari, A. H. (2021). Lightweight hybrid detection of data exfiltration using dns based on machine learning. In *2021 the 11th International Conference on Communication and Network Security*, pages 80–86.

Mishra, P., Varadharajan, V., Tupakula, U., and Pilli, E. S. (2018). A detailed investigation and analysis of using machine learning techniques for intrusion detection. *IEEE communications surveys & tutorials*, 21(1):686–728.

MontazeriShatoori, M., Davidson, L., Kaur, G., and Lashkari, A. H. (2020). Detection of doh tunnels using time-series classification of encrypted

traffic. In *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pages 63–70. IEEE.

Moustafa, N. and Slay, J. (2015). Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)*, pages 1–6.

Naway, A. and Li, Y. (2018). A review on the use of deep learning in android malware detection. *arXiv preprint arXiv:1812.10360*.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. (2018). Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31.

Qiu, J., Zhang, J., Luo, W., Pan, L., Nepal, S., and Xiang, Y. (2020). A survey of android malware detection with deep neural models. *ACM Computing Surveys (CSUR)*, 53(6):1–36.

Rahali, A. and Akhloufi, M. A. (2021). Malbert: Malware detection using bidirectional encoder representations from transformers. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 3226–3231. IEEE.

Rahali, A., Lashkari, A. H., Kaur, G., Taheri, L., Gagnon, F., and Massicotte, F. (2020). Didroid: Android malware classification and characterization using deep image learning. In *2020 The 10th international conference on communication and network security*, pages 70–82.

Ring, M., Wunderlich, S., Grüdl, D., Landes, D., and Hotho, A. (2017a). Creation of flow-based data sets for intrusion detection. *Journal of Information Warfare*, 16:40–53.

Ring, M., Wunderlich, S., Grüdl, D., Landes, D., and Hotho, A. (2017b). Flow-based benchmark data sets for intrusion detection. In *Proceedings of the 16th European Conference on Cyber Warfare and Security (ECCWS)*, pages 361–369. ACPI.

Shalaginov, A., Banin, S., Dehghantanha, A., and Franke, K. (2018). Machine learning aided static malware analysis: A survey and tutorial. *Cyber threat intelligence*, pages 7–45.

Sharafaldin., I., Habibi Lashkari., A., and Ghorbani., A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISSP,*, pages 108–116. INSTICC, SciTePress.

Sharafaldin, I., Lashkari, A. H., Hakak, S., and Ghorbani, A. A. (2019). Developing realistic distributed denial of service (ddos) attack dataset and taxonomy. In *2019 International Carnahan Conference on Security Technology (ICCST)*, pages 1–8.

Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A. (2009). A detailed analysis of the kdd cup 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pages 1–6.

Wang, H. and Li, W. (2021). Ddostc: A transformer-based network attack detection hybrid mechanism in sdn. *Sensors*, 21(15):5047.

Youden, W. J. (1950). Index for rating diagnostic tests. *Cancer*, 3(1):32–35.