Bad Design Smells in Benchmark NIDS Datasets

Robert Flood
University of Edinburgh
Edinburgh, United Kingdom
r.flood@ed.ac.uk

Gints Engelen
KU Leuven
Leuven, Belgium
gints.engelen@kuleuven.be

David Aspinall
University of Edinburgh
Edinburgh, United Kingdom
da@ed.ac.uk

Lieven Desmet
KU Leuven
Leuven, Belgium
lieven.desmet@kuleuven.be

Abstract—**Synthetically generated benchmark datasets are vitally important for machine learning and network intrusion research. When producing intrusion datasets for research, providers make complex, subtle and sometimes unwary decisions that can affect data utility. Unfortunately, examining network data is difficult, so these decisions are rarely audited. We perform an in-depth manual analysis of seven highly-cited benchmark datasets, discovering six suspect design patterns, which we term 'data design smells'. We formulate six heuristics to measure the prevalence of these issues. These design choices, if not properly accounted for, can introduce severe experimental bias, which we demonstrate with four concrete examples. We then conduct a systematic impact analysis of the wider literature that relies on these datasets. Our results suggest that bad design smells correlate with poor data diversity, murky labelling and poorly-defined generalisation criteria. Worryingly, we find that improper usage of these datasets can weaken their utility as benchmarks which, in turn, biases downstream intrusion detection research. We conclude with some recommendations for using and creating NIDS datasets to help alleviate these issues.**

1. Introduction

Benchmark datasets are vitally important in machine learning research. Datasets such as MNIST [60] and CIFAR-10 [57] allow researchers to compare methodologies on a fixed playing field, helping drive forward innovation. Unfortunately, datasets are rarely perfect real-world representations. It is well known that statistical properties of datasets may be considerably simplified when compared to that of real-world data. The ML pipeline is delicate; improper data may introduce experimental bias, weakening research findings. Even established benchmark datasets can contain defects, such as mislabeling in CIFAR-10 [76], arbitrary class distinctions in ImageNet [17] or run-to-failure bias in the Yahoo S5 dataset [110]. In the absence of high-quality datasets and critical examination, experimental bias may be endemic to an entire research field. To maximise the utility of datasets, identifying mistaken assumptions and eliminating their downstream effects are vital.

Our work shows that widely-used network intrusion detection system (NIDS) datasets could suffer similar issues when used for benchmarking ML methods. Unlike fields such as image or voice recognition, popular NIDS datasets can consist of synthetically generated data. Often, this data is generated via a series of scripted interactions across a testbed of virtual machines which are recorded and converted into summary network flow statistics. The research community should be grateful for these datasets; publicly available network data is precious and these issues do not make these datasets unsuitable for all purposes. However, properly understanding their suitability for a given task is critical.

We are not the first to highlight this: research discussing issues with NIDS datasets is a well-established topic [12], [23], [24], [52], [53], [97] alongside critiques of specific datasets [39], [66]. In contrast to prior work, we aim to assess the potential impact of *data design* of NIDS datasets on downstream research. When creating an image dataset, many choices are ultimately arbitrary such as the contents or size of each class — there is little reason for CIFAR-10 to contain frogs over, say, turtles. In image recognition, the feature space and problem space are also closely aligned. In contrast, NIDS datasets rely heavily on domain-specific choices: dataset providers must decide how the network is configured, what attack classes to include, how attacks are launched, what features to extract, how to imitate benign traffic, and so on, which are typically obfuscated in the feature space. These choices provide a contextual backdrop that significantly alters threat models, attacker behaviour and implicitly defines generalisation standards. We describe these choices, conscious or unconscious, as *data design*. Questionable data design choices are difficult to correct, potentially introducing serious bias.

In this work, we analyse seven well-cited NIDS datasets, each with varying levels of documentation. To abstract away from the available documentation in our analysis, we distilled six indicators of potential design violations. Analogous to the term *design smell* [16], [99] in software engineering — signals of questionable design practises — we observe *data design smells*. We find dubious practises in all datasets: attacks launched against closed ports, labels leaked via features and millions of near-duplicate flows, to name a few. Altogether, our work shows that smells are ubiquitous in modern NIDS datasets.

For all seven datasets we analyse, we identify six data design smells — **poor data diversity**, **highly dependent features**, **unclear ground truth**, **traffic collapse**, **artificial diversity** and **wrong labels**.

In Section 2, we introduce these seven datasets. Then, in Section 3, we examine 38 papers that use these datasets, seeded from top conferences and their citations. We investigate four papers in detail, recreating their work when necessary, and demonstrate how NIDS dataset design could undermine their results. We find that the performance of

CADE [113], a state-of-the-art concept drift detector, could rely on dataset artefacts, presenting worse performance on fixed data. We also can outperform AJSMA [91], an adversarial perturbation method, with a trivial attack, suggesting that "smelly" data must be used with caution when benchmarking adversarial attacks on NIDS. These examples and two more are covered in Sections 3.1–3.4. Looking at the remaining papers more generally, we assess authors' assumptions about NIDS data design, either explicitly stated or implied via their methodologies. We find that questionable assumptions are common and that examination of raw network data is rare, possibly leading to experimental bias. We discuss this in Section 3.5.

In order to fully understand the scope of these issues, we then undertake a thorough manual and automated analysis of these datasets. We discuss the design of our manual analysis in Section 4.1 and we comprehensively analyse 65+ attacks. In Section 4.2, we develop six heuristic measures to assess the severity of these issues automatically, which can be used on future datasets. Both stages of our analysis uncover suspect design choices in all datasets, covered in Section 5. We catalogue these so researchers using these datasets can avoid experimental pitfalls. For instance, we show that it is often trivial to get perfect classification accuracy using a single feature that is unrelated to the underlying mechanism of an attack. Moreover, our automated heuristics score far better on general tabular anomaly datasets [82], suggesting that the impact of these design choices are particular to NIDS datasets. Our results enable a deeper understanding of the implications that questionable data design may have on downstream NIDS research.

In Section 6, we discuss some potential recommendations to account for NIDS dataset design choices and to improve the standard of intrusion detection research. We conclude with related work in Section 7, limitations in Section 8 and discussion in Section 9. To summarise, our contributions include:

- **Dataset Analysis**: We devise a novel, comprehensive methodology for analysing network data design, identifying potential research pitfalls stemming from data design. We apply this to seven popular synthetic NIDS datasets and 65+ individual attack classes. We find that dubious data design practises are ubiquitous across popular NIDS datasets, necessitating their careful treatment as benchmarks.
- **Design**: Analogous to *code smells* in computer programming, we identify six indicators of potentially bad data design choices that we call *data design smells*.
- **Prevalence**: For each smell, we distill a heuristic measure to evaluate its severity. These heuristics are designed to be lightweight, allowing us to measure 65+ attack classes. Our results show that negligible data diversity, severe mislabelling and trivial classification complexity are common in NIDS datasets.
- **Impact**: We study 38 papers, seeded from top conferences, that rely on these datasets, summarising their questionable assumptions. We also investigate four papers in detail, recreating their methodologies when necessary, and demonstrate how bad data design smells impact their experimental results.
- **Recommendations**: We propose guidelines for using/developing NIDS datasets to minimise the impact of these design smells, providing insights into how to improve data usage such that we as a community ensure higher quality intrusion detection research.

2. Background

Releasing real-world data has severe privacy drawbacks and establishing the ground truth of real-world traffic is notoriously difficult [22]. Thus, synthetic datasets are commonly used in IDS research, generated using data collection testbeds.

We examine NIDS datasets that consist of two parts: the original traffic, in PCAP format, and a set of preprocessed statistics summarising each flow. This is a limitation of our approach as we require traffic captures that we can manually audit and assume there is an accompanying feature set for our automated analysis. Despite this commonality between our chosen datasets, there are fundamental differences that complicate comparisons. For instance, despite attempts to standardise feature sets [11], [85], researchers often use the bespoke flow statistics that accompany a dataset.

In this section, we provide a brief overview of each dataset examined and the documented design choices of their authors. We omit the popular datasets KDD Cup [3] and NSL-KDD [101], as these are both derivatives of DARPA 1999 [65], which has long known to be faulty [69].

There are some modified versions of these datasets [39], [66], [85], which either fix some labelling issues or alter feature sets, however, we examine the underlying dataset design, which cannot be changed by modifying feature sets. We use the original versions — unless where otherwise stated — as we aim to evaluate existing research, which predominately uses these original datasets.

TABLE 1: Dataset Summary

Dataset	Year	Class	Feat.	Hosts	Cit.[1]
CIC IDS 2017	2017	14	80	14	3264
CIC IDS 2018	2018	16	80	500	3264
ICSX 2012	2012	2/5[2]	20	25	1365
UNSW-NB15	2015	10	49	45	2817
Ton_IoT	2019	10	44	12	254
Bot-IoT	2021	5	45	10	1217
CTU-13	2014	13	15	-	866

CIC IDS 2017/18 [89] – Developed by the Canadian Institute of Cybersecurity, *CIC IDS 2017* (*CIC 17*) and *CSE-CIC IDS 2018* (*CIC 18*) share a similar design, the latter being an expanded version of the former. *CIC 17* comprises 14 hosts and *CIC 18* comprises of 500 hosts.

Flow statistics were calculated using the CICFlowMeter tool [1], which measures 80 features. The authors of these datasets emphasize their role as '*benchmark*' datasets and prioritize the '*realism*' of the background traffic and the '*diversity*' of attacks.

1. Number of citations according to Google Scholar, 21/03/2024

2. We split ISCX's attack class into 5 based on destination port information, corresponding with each unique stage. Splitting the attack traffic is common in the wider research [32], [40].

ISCX 2012 [94] – The *ISCX 2012* dataset is the predecessor to the above-mentioned *CIC 17 and 18* datasets, with some differences. First, the authors used the IBM QRadar tool [2] tool to extract a unique set of statistical features. Second, attacks were launched as four overlapping, multi-stage 'scenarios'. We consider each unique stage of these scenarios to be a distinct attack and label them accordingly.
UNSW NB15 [74] – Unlike the other datasets examined, *UNSW NB15* contains data created using a hardware traffic generator, combined with benign traffic from a cyber range. The feature set is divided into four categories: basic, content, time and additional features, consisting of protocol specific flags and connection rate-based features.

The dataset's design emphasises its recency; the authors say that it contains 'contemporary synthesized attack activities' and they claim that the dataset is more complex than *KDD Cup* [3], [75].
TON_IoT [73] – *TON_IoT* is a collection of datasets, including sizeable network traffic. The feature set contains few continuous features; instead, most are protocol specific flags, relating to DNS, SSL and HTTP connections, such as the domain name of a DNS query. Design claims emphasise the '*heterogeneity*' and '*complexity*' of the data, with the authors publishing work quantifying this [21].
Bot-IoT [56] – The paper describing *Bot-IoT* stresses realism as a design goal, stating that it is a '*realistic . . . dataset*' with a '*massive amount*' of '*realistic benign traffic*'. The authors provide two variants of *Bot-IoT*: a full version with a truncated feature set and a condensed version with an additional 16 aggregate features, such as the number of packets per IP. We evaluate the condensed version as it is more commonly used in the papers discussed in Section 3.
CTU-13 [41] – CTU-13 is a botnet dataset consisting of 13 'scenarios'. It differs from the other datasets we analyse in two major ways: first, data from the infected host is labelled in a highly granular manner and, second, it contains both *Background* and *Normal* traffic. In Section 5, we use the normal traffic for our comparative measures, as this was used as the benign traffic in the original accompanying paper. Due to the highly granular nature of the labels, we combine similar labels to form our classes, providing more detail in the Appendix.

3. Bad Smells and their Downstream Impact

As ML-based NIDS in research often use flow statistics rather than raw network data, the underlying traffic is obfuscated, such as the services within or low-level choices about the attacks. The papers accompanying these datasets sometimes provide limited descriptions of the generation process [21], [74], [88] and there is no comprehensive account of what specific traffic is in these datasets. As a result, a naive security researcher could be unaware of what they are detecting beyond high-level labels, such as 'Exploits'. Although researchers could produce such an account themselves, in Section 3.5, we argue that assuming that datasets can be used 'off-the-shelf' with limited analysis has become the default in the research community. Thus, there has been little auditing to uncover potential complications in these datasets. We aim to bridge this gap in knowledge.

To evaluate the usage of these datasets in research, we systematically review a subset of well-regarded papers that rely on these datasets. For our selection criteria, we began with works published between 2015 and 2023, inclusive, at the seven top-ranked non-cryptography computer security conferences — according to [116] — which cite at least one of these datasets. As we could not find many papers citing ISCX 2012, Ton-IoT, Bot-IoT or CTU-13 via this list, we expanded our criteria to include a greater number of security conferences, including CNS, RAID and DIMVA, as well as networking and data mining conferences, including KDD, WWW, CIKM and InfoCOM. Thus, we source papers from USENIX Security, S&P, EuroS&P, CCS, AsiaCCS, CNS, RAID, DIMVA, KDD, InfoCOM, WWW, SAC, ACSAC and CIKM. We collate a list of 38 papers via this process, excluding systematisation of knowledge papers and papers whose main aim is to point out issues in other areas of NIDS dataset usage (for more details on our paper selection criteria, see Appendix B).

We look more closely at four example papers — two directly from the above overview and two cited by papers in the overview — and demonstrate how questionable data design may have impacted their results. In doing do, we observe that these complications stem from data patterns which we explicitly highlight. These observations lead directly to our bad data design smells, which we ***emphasise*** in the text. Altogether, our aim is to demonstrate that these datasets are being used at top-level conferences with little auditing or examination of the underlying data, whilst referencing similarly unsuspecting work.

We stress that we choose the phrase 'data design smell' because, just like bad smells in software design, they are merely *indicators* of *potentially* bad practises, and using "smelly" data does not immediately invalidate research results. In the following examples, we do not claim to negate the methodologies of the examined work. Instead, we wish to demonstrate how assumptions about NIDS data design can produce misleading conclusions.

3.1. Example 1 - LUCID

Original Paper. LUCID [36] is a highly-cited, state of the art DDoS classifier, evaluated using the DDoS traffic in *ISCX 2012*, *CIC 17* and *CIC 18*. The authors' code has been made open-source [35].

At *LUCID*'s core is a traffic preprocessing algorithm. Ten features are extracted from the first n packets of a flow, zero padding when necessary, combining packet-level and flow-level information. This produces a 2-dimensional data structure of size $10 \times n$ which feeds into a Convolutional Neural Network to discriminate DoS flows from benign.

Data Design. Upon examining *CIC 17*, we find that a single webpage is attacked across all DoS classes, namely, the default Apache page. As a result, the packet size features are extremely narrowly distributed, with 97% of total backwards packet size features approximately equal to $11595 \pm 1\%$ bytes (discounting flow calculation artefacts [39]). Moreover, flows with this value do not appear in the benign data.

Experiment. Fixing $n = 10$, we repeat LUCID's feature extraction process on *CIC 17*. The authors consider a number of values for n and also truncate flows according to timing parameters. We found these modifications negligible and use the default values from [35]. We compare this to a simpler feature extraction process as a baseline experiment. Whilst we still consider just the first n packets of a flow, we extract only 3 features: total TCP size, total packet size and flow duration. Note that we've discarded the granular packet-level information, resulting in a massive reduction of LUCID's 100 features. We use a random forest as our classifier.

TABLE 2: Results of LUCID, Baseline and Baseline (Corrected) on *CIC 17*

Classifier	ACC	F1	TPR	TNR
LUCID	0.997	0.997	0.9988	0.9953
BL	0.997	0.997	0.9985	0.996
BL (C)	1.000	1.000	1.000	1.000

Results & Analysis. Table 2 shows that we achieve comparable results to LUCID, despite using a much smaller feature set. For both LUCID and our baseline model, most misclassified flows were failed TCP handshakes. Filtering these flows, we produce a *corrected* version of the DoS dataset used by *LUCID*. On this dataset, a random forest achieves perfect accuracy and recall (BL (C)).

The design of LUCID implicitly assumes that predictive power can be gained by combining packet-level and flow-level information. However, given the severe lack of variation in *CIC 17*, this is not true. The paucity of variation in *CIC 17*'s DoS traffic stems from two data design choices made by the dataset authors: only launching DoS attacks against a single webpage and using fixed network conditions. Our analysis of *ICSX 2012* and *CIC 18* suggests that similar results would hold for those datasets.

Note that we cannot pass judgement on the effectiveness of LUCID in other, more realistic settings, where a more complex architecture might be justified. However, due to the design of the chosen test datasets, *LUCID's complexity is not justified by the experiments performed by Doriguzzi et al.*

Bad Smell 1. Many NIDS datasets contain data generated via automated tooling with fixed configurations or limited exploration of an attack's capabilities. This homogeneity causes ***poor data diversity***, inadequately testing a model's generalisation capabilities and rewarding overfitting.

3.2. Example 2 - AJSMA

Original Paper. Considering adversarial attacks in constrained domains, Sheatsley et al. [92] present the Augmented Jacobian Saliency Map Attack (AJSMA), a white-box attack evaluated on *NSL-KDD* and *UNSW NB15*. The motivating insight of AJSMA is that, in intrusion detection, the problem space and feature space are distinct and arbitrary transformations may result in invalid data. Thus, when perturbing features, attacks must adhere to constraints. The ability of AJSMA to generalise across models is tested using five neural networks (trained using a stratified shuffle-split and labelled M_A - M_E) as well as other ML models, including Decision Trees (DT).

Data Design. As a result of *UNSW NB15*'s testbed, a subset of features are highly performant across multiple attack categories, despite being apparently unrelated to the attacks' underlying mechanisms. In particular, the *Protocol* and *TTL* features overlap minimally between the benign and malicious classes and it is possible to separate these classes with 98% accuracy using these features alone. We provide more insight into why this is the case in Section 5.1

Experiment. As a baseline comparison to AJSMA on *UNSW NB15*, we consider a simple feature perturbation attack: by modifying features, we 'convert' all attack flows to UDP (by altering the 'Protocol' and 'RTT' features), and TTL values to match those of benign traffic. These modifications are considered valid under the constraints that AJSMA adheres to; the original paper assumes that sound attack traffic can be created by converting TCP traffic to UDP and vice versa, provided constraint satisfaction. Unlike AJSMA, we do not assume access to the model's gradients or parameters. Because of this, we cannot replicate Sheatsley et al.'s evaluation process exactly. However, we do consider the performance of our attack across multiple models.

TABLE 3: Results of AJSMA and our Heuristic Attack on *UNSW NB15*. We consider the average accuracy across the tests presented by Sheatsley et al., using the notation $M_i \rightarrow M_j$ to denote an attack on model M_j using the gradients of M_i where i, $j \in A, B, C, D, E$ and $i \neq j$.

Attack	$(M_i \rightarrow) M_i$	$(M_i \rightarrow) M_j$	$(M_i \rightarrow)$ DT
AJSMA	1.000	0.790	0.166
HA	1.000	1.000	1.000

Results & Analysis. Table 3 shows our attack achieves identical performance to AJSMA on *UNSW NB15*. However, we note that AJSMA's performance degrades when generalising across models. As our attack does not rely on the gradients of a specific model, we maintain perfect adversarial accuracy across all models tested.

We note that the performance of our attack is not due to any inherent qualities of malicious traffic. Instead, the large disparity between the benign and attack traffic is the result of data design choices, in the form of protocol and TTL choice in *UNSW NB15*. As a result, whilst AJSMA may be a superior attack in the general setting of constrained adversarial examples, the presented accuracy on *UNSW NB15* is not a meaningful measure of the attack's effectiveness. Moreover, as producing perfect adversarial perturbations is trivial, *worthwhile comparisons between AJSMA and alternative attack methodologies using UNSW NB15 are impossible.*

Bad Smell 2. Poor design of simulation testbeds can result in features of outsized importance that are unrelated to the underlying mechanism of an attack. Such ***highly dependent features*** reduce the complexity of attack detection and lead to overly optimistic interpretations of classifier performance.

3.3. Example 3 - Domain Adaptation (ADA)

Original Paper. Due to the high rate of concept drift in security tasks, such as intrusion detection, ensuring that a deployed classifier can generalise to unseen attack classes is important. Singla et al. [96] propose a methodology for training NIDS to a rarely seen attack class via adversarial domain adaptation, evaluated on *UNSW NB15*.

Singla et al. preprocess *UNSW NB15* into two datasets: a *source* dataset, containing benign traffic and eight attack classes, and a *target* dataset, containing benign traffic and a ninth attack class, not included in the source dataset. At training, only a small number of samples from the target dataset are used. We focus on the case where 100 samples are used as this situation is highlighted by Singla et al, who consider the *Exploits*, *Reconnaissance* and *Shellcode* classes as holdouts.

Singla et al.'s ADA architecture has two parts, a generator and discriminator model. The generator has two goals, taking samples from both the source and target datasets, converting them into a domain-invariant embedding. This embedding is fed into a softmax layer, which classifies a sample as malicious or benign. Simultaneously, the discriminator identifies whether the embedding comes from the source or target dataset. The generator is trained such that this embedding fools the discriminator. Once finished, the generator functions as a NIDS, capable of high performance on the target dataset despite having access to only a small number of samples.

Data Design. Analysing *UNSW NB15*, we found heavy overlap between many of the attack classes, as well as features that correlate highly with all classes. As a result, it is dubious whether the target dataset can meaningfully be considered distinct from the source dataset, an implicit assumption in Singla et al.'s training methodology. In particular, the three most common combinations of forward and backward packets counts for the *Exploits* class also make up 40% of the *Shellcode* and *Reconnaissance* classes, and many of these flows appear to be notionally identical. This overlap between attack classes also leads to an overlap in highly discriminative features. Having highly similar attacks across disconnected attack categories is an implicit data design choice that, if unaccounted for, leads to test set leakage in experiments similar to Singla et al.'s.

Experiment. We recreate Singla et al.'s set-up, reproducing their results. We then repeat the experiment whilst removing entries from the source dataset that are found in multiple classes, identified via the source and destination packets features.

We also remove features that are unjustifiably performant on the malicious data, such as *sttl*, *dttl* and *synack* (specifically, we remove the 'TTL' and 'RTT' features with HDF_C values higher than 0.7, detailed in Section 4.2). This process results in a large number of samples being removed from both the *Reconnaissance* and *Shellcode* classes, preventing us from repeating Singla et al.'s experiments using those classes as holdout classes. Thus, we only consider the case where the target dataset contains samples from the *Exploits* class.

Results & Analysis. On the unmodified version of *UNSW NB15*, we reproduce similar results to those presented by Singla et al., achieving a 6% gap between

TABLE 4: Best Reported Accuracy of ADA model vs Base case model for 100 target training samples.

Classifier	Original	Modified
Base (60 epochs)	0.82	0.82
Base (1000 epochs)	0.8484	0.8442
ADA (10000 iterations)	0.8804	0.8350

the base case and ADA models[3]. However, when we remove the problematic attack samples and classes, this advantage drops to a 1.5% performance gap. When removing malicious traffic, we downsample the benign traffic to maintain the same benign/malicious ratio as before. We also note that the base model neural network can exceed this score by extending its training regime.

From our experiment, Singla et al.'s results are biased by several data design issues, namely, unclear attack classes with incomplete attack capture, which lead to poorly defined boundaries between attack classes. We note that the plurality of this overlap stems from attacks in UNSW with no apparent effect, and it is unclear how legitimate this attack traffic is. We emphasise that the assumptions made by Singla et al. about *UNSW NB15* are completely reasonable; *Exploits*, *Reconnaissance* and *Shellcode* are distinct categories of attacks and there is little reason to assume that this conceptual blurring between classes would be present in the data. However, this demonstrates that, *without modification, UNSW NB15 is unsuitable for evaluating the ability of classifiers to generalise between attack categories.*

Bad Smell 3. Datasets can lack clear labelling logic, often labelling background services as attacks for unspecified reasons. This ***unclear ground truth*** creates a disconnect between what researchers understand a class to contain and what it actually contains, limiting their ability to reason about their methodology and results.

3.4. Example 4 - CADE

Original Paper. Yang et al. [113] also combat concept drift via contrastive learning with CADE. CADE leverages *contrastive learning* to detect drifting samples, including an evaluation on *CIC 18*.

In their experimental setup, the authors picked one day's worth of benign traffic in addition to malicious traffic from the *Infiltration*, *DoS Attacks - Hulk* and *SSH - Brute Force* classes. They then iteratively train their classifier on the benign traffic and two malicious classes; the third malicious class represents the 'unseen' class and is only used in the test set.

Data Design. Analysing *CIC 18*, we note that attacks take place within short time frames. Whilst this is a legitimate data design choice, downstream researchers must be aware of this fact when evaluating their methodologies.

3. Although Singla et al. report that they use a source training dataset with 83,961 samples, it's unclear what ratio of benign to malicious traffic they use. We achieve similar results using a source training dataset with 53,112 benign and 38,679 malicious samples.

Unfortunately, CADE does not remove the *Timestamp* feature during evaluation, leading to a potentially *highly dependent feature*.

Experiment. We use a corrected version of *CIC 18* [66] after verifying the author's fixes to the labelling and feature extraction process, and reran the CADE experiments (with the original and fixed versions of the dataset), whilst removing the *Timestamp* feature.

Results & Analysis. Table 5 shows our results. The performance is severely degraded for all but the original setup. These results reinforce **Bad Smell 2** and we note that mislabelled data cause serious experimental bias.

Bad Smell 4. Inaccurate ground truth of generation testbeds can lead to mislabelled data. The ***wrong label*** smell degrades the ML pipeline by altering classification complexity. Furthermore, researchers discovering disparate subsets of labelling issues can lead to inconsistent benchmarks, complicating direct comparison between techniques or architectures.

This example highlights how data design smells are **not** harmful in all contexts: including the *Timestamp* is not unreasonable if the data shows periodic behaviour, as in UGR'16 [68]. However, in *CIC 18* attacks reside within narrow time-windows [88], making *Timestamp* a *highly dependent feature* for this dataset, as it is both highly performant with no connection to the attack's underlying mechanism.

TABLE 5: F1 results for CADE on the original and fixed *CIC 18* dataset, with and without the *Timestamp* feature.

	With Timestamp		Without Timestamp	
	Original	Fixed	Original	Fixed
SSH - BF	0.8687	0.4968	0.1214	0.0
DoS Hulk	0.9997	0.9988	0.7614	0.9987
Infiltration	0.9999	0.9929	0.0537	0.9964

3.5. Potential Data Bias in Research

We now look at the papers more generally. We assess implicit assumptions made across four criteria, reflecting our smells thus far. First, we consider assumptions about *data diversity*, which we subdivide into attack variation (AV) — the number of distinct interactions in a class — and feature variation (FV) — the variability of features in a class. We then assess whether papers include critical, post-hoc analysis of feature importance, connecting features to mechanism of an attack. In the absence of such analysis, we say the paper assumes that the data was free from *highly dependent features* (HDF). Finally, we consider whether papers assume the data was free from *wrong labels* or *unclear ground truth*, which we combine into a single criteria (W/U). This process was undertaken by a single author and then repeated by a second author on a random subset of 25% of papers, who then cross-referenced their findings to ensure agreement. The full paper analysis methodology as well as paper selection criteria are detailed in Appendix B.

Results & Analysis. We list our results in Table 7, marking assumptions as 'unclear' when unable to fairly judge. As papers made few comments about the data, we judged assumptions implicitly via their methodologies.

Many papers applied techniques that seem unjustifiably complex given the low attack variation in classes, such as training individual models for each attack [105] or using a complex setup such as a LSTM variational autoencoder [108], whilst a minority aim to generalise between attack classes [47], intentionally injecting variety. Without a critical examination of feature importance, we believe that papers in our overview overwhelmingly assumed the datasets were free of *highly dependent features*. Such analysis was rare, with [44], [52], [59], [86] being notable. Some papers used techniques that assumed greater feature variety than is present, such as oversampling via SMOTE [13]. Due to low variability, we demonstrate how naive application of techniques such as SMOTE [25] fails to introduce feature variety in Table 6 and how packet-level features succumb to low variation in Section 3.1. Papers also made statements about the properties of NIDS data *generally*, without investigating whether specific dataset design characteristics were responsible for their results [40], [90].

With a few exceptions [52], [59], [114], we found no evidence that papers audited any raw PCAP data. As a result, almost all papers used mislabelled or unclear data, highlighted in Section 5.1. Based on their research aims, mislabelled data was irrelevant for some papers [26], [51] (as such, these papers were left out of our scope), and a minority used a corrected version of *CIC 17* [86] and *CIC 18* [59]. Although these corrected dataset were released after many of these papers, cursory manual analysis would have also uncovered mislabelling issues.

Often, misconfigured testbeds result in failed attacks e.g., attacks launched against closed ports. We understand that detecting these connections is a reasonable goal for an IDS. However, due to the homogeneity of the traffic, these can be trivial to detect in a machine learning setting, as standard, randomised train/test splits result in data leakage. No papers in our overview commented on this or amended their evaluation process, and all were seemingly unaware of these discrepancies. We believe it is exceedingly likely that papers would re-evaluate their proposed model architectures/pipelines if aware of the simplicity of the classification task. This suggests a new bad smell.

Bad Smell 5. Simple configuration mistakes can extinguish data diversity from a class. We call this problem ***traffic collapse***, as statistics 'collapse' into a trivial distribution, preventing models from learning any meaningful information from features.

In *CIC 17* and *Bot IoT*, because of spurious network conditions, some attack traffic is questionable, such as malformed connections or extraordinarily high retransmission rates. We believe these out-of-distribution flows are pernicious, preventing classifiers from achieving **perfect** accuracy, instead presenting **near-perfect** accuracy. The former implies that classification may be trivial whilst the latter does not, justifying the use of complex ML architectures. Only very few papers appeared to examine misclassifications to some extent [55], [108], [114]. This introduces our final bad smell.

Bad Smell 6. When generating network traffic, a number of difficult-to-control variables — poor network conditions, network capture failures, retransmission rates etc. — impact the structure of flows. If not properly managed, these variables create ***artificial diversity***, causing researchers to overestimate classification complexity.

TABLE 6: *Total Length of Fwd Packet* is highly discriminitive in *CIC 17*. For 72.7% of *Heartbleed* flows, this feature equals 7920. Furthermore, all variability in this class is caused by *artifical diversity*. Correcting the original data (C) exaggerates this effect.

	Org.	Org. (C)	SMOTE	SMOTE (C)
TLFP	72.7%	100%	71.4%	100%

4. Finding Bad Smells

In this section, we introduce a methodology for examining NIDS datasets. This analysis has two stages: a manual stage — a qualitative evaluation of the problems with these datasets — and an automated stage — a quantification of bad smell prevalence and severity via heuristic measures. To assess the rate of false positives, we design our methodology without reference to *CTU-13* and use it as a test case.

4.1. Manual Analysis

We aim to document all flows within each attack class. Complete coverage is vital; it is likely that researchers using these datasets will find a subset of problems and remove the offending traffic. Thus, researchers are not comparing their results on a fixed dataset but rather on several disparate datasets, each corrected in a unique manner. Standardising this process requires a full examination of the underlying PCAP data.

Examining each flow is onerous and time-consuming. Instead, we assume that we can identify unique attacker behaviour via unique values of the *Total Source Bytes* feature, a standard inclusion in the accompanying feature sets. Similarly, we assume that unique values for the *Destination Port* and *Total Destination Bytes* features correspond with unique victim behaviours. We cluster flows that attain the same values on these features, up to small variations, reducing the number of flows to be analysed from tens of thousands to a small number of clusters based on CSV data alone.

For each of these clusters, we randomly select an exemplar flow and locate it in the PCAP data via Source and Destination IPs/Ports and Timestamp information. Using these exemplar flows, we examine each cluster in parallel. For each flow cluster, we aim to understand the generation process that give each cluster its characteristic properties. These include understanding the attack, how the attack is realised, the target service, the labelling logic and the intra- and inter-cluster relationships between flows. We survey each flow via a series of yes/no questions, each related to a bad smell from Section 3.

Q1: Wrong Label – Does the flow's label accurately describe its behaviour? Relying on the provided documentation and contextual clues, we assess whether its label is correct. Labels have varying degrees of granularity, including specific attacks, attack classes and high-level descriptions — e.g., *Heartbleed*, *Reconnaissance* and *Attack*. We rely MITRE's *CVE*s [70] and *CWE*s [71] for attack definitions. For vaguer labels, we rely on personal assessment. If we can't associate a flow cluster to its label, the *wrong label* smell is present.

Q2: Unclear Ground Truth – Does the flow originate from the attacker network and/or is directed towards the victim network? With the exception of 'insider' attacks, we expect attacks to occur between the attacker and victim networks, or within the attacker network for, say, C&C traffic. Failing this indicates that *unclear* traffic has been labelled alongside the attack. We check whether these flows are associated with any background processes to confirm this.

Q3: Highly Dependent Features – Do distinct clusters share similar properties? Although we aim to capture similar flows in our clusters, it is problematic when attacker behaviour (determined in **Q1**) is identical across clusters (whilst differing from the benign data). These properties may be reflected in the chosen feature set, biasing models via unrelated features. If so, we consider the *highly dependent feature* smell to be present.

Q4: Artificial Diversity – Is the primary difference between clusters due to network artefacts? Unrealistic network conditions may lead to similar flows (which we determine via **Q1**) being distributed across several clusters. We consider failed handshakes, aborted flows, frequent network anomalies — e.g, dropped packets or retransmissions — and differences in flow termination as network artefacts. If clusters differ due to these phenomena, the *artificial diversity* smell is present.

Q5: Poor Data Diversity – Are clusters large, relative to the size of the class? Based on our assumptions, large cluster sizes indicate that a class mostly consists of both the attacker and victim engaging in the same behaviour repeatedly. If a cluster (and any related clusters identified in **Q4**) contains over 25% of all flows, we assume there is *poor data diversity*. When possible, we confirm the source of this lack of variety — such as the reliance on automated tooling — via the details gathered during **Q1** and **Q3**. We note that it is reasonable to expect certain volumetric attacks, such as ACK floods, to have low data diversity. However, detecting such attacks via ML still requires careful evaluation, due to the risk of overfitting to arbitrary features or leaking test set data.

Q6: Traffic Collapse – Has the attack been fully executed? When an attack is not fully realised, due to, say, a secure service, the response from the victim is typically limited across the entire interaction. We examine flows for evidence of host responses. Unexpected behaviours include backwards flows containing only RST packets and flows with no response. Here, we say the *traffic collapse* smell is present.

A high-level summary of this process is in Figure 1. If a question's answer is unclear based on a single flow,

TABLE 7: Paper Assumptions. ✓: assumption present, ✓*: assumption partially present, ✘: assumption not present/relevant, -: unclear. I: *ISCX 2012*, C: *CIC 2017*, C2: *CIC 2018*, U: *UNSW NB15*, CT: *CTU-13*, B: *Bot IoT*, T: *Ton_IoT*.

		Assumptions			
Paper	Dataset	FV	AV	HDF	W/U
[40]	I,U	✓	✓	✓	✓
[115]	I,CT	✓	✓	✓	✓
[79]	U	✓	✓	✓	✓
[96]	U	✓	✓	✓	✓
[6]	I,CT	✓	✓*	✘	✓
[61]	C	✓	✓	✓	✓
[112]	CT	✓	✓	✓	✓
[15]	C	✓	✓	✓*	✓
[62]	C	✓	✓	✓	✓
[100]	C	✓	✓	✓	✓
[114]	U	✓	✓*	✓	✓
[47]	C	✓	✓	✓	✓
[113]	C2	✓	✓	✓*	✓
[18]	C2,U	✓	✓	✓	✓
[5]	I,C,C2	✓	✓	✘	✓
[49]	I,C	✓	✓	✓	✓
[108]	U	✓	✓*	✓	✓
[107]	C	✓	✓	✓	✓*
[52]	C	✘	✘	✘	✓
[63]	C,C2	✓	✓	✓	✓
[19]	U	✓	✓	✓	✓
[14]	B,T	✓	✓	✓	✓
[87]	I,C2,CT	✓	✓	✓	✓
[111]	U	✓	✓	✓	✓
[13]	T	✓	✓	✓	✓
[106]	C2	✓	✓	✓	✓
[48]	C2	✓	✓	✓	✓
[59]	C	✘	✘	✘	✘
[55]	U	✓	✓	✓	✓*
[33]	C	✓	✓	✓	✓
[44]	U	✓*	✓*	✘	✓
[105]	CT,T	✓	✓	✓	✓
[86]	C	-	-	✘	✓*
[90]	U	✓	✓	✓	✓
[34]	B	-	-	-	-
[30]	C,C2	✓*	✓	✓	✓
[78]	U	✓	✓	✓	✓
[20]	I,C	✘	✘	✘	✓

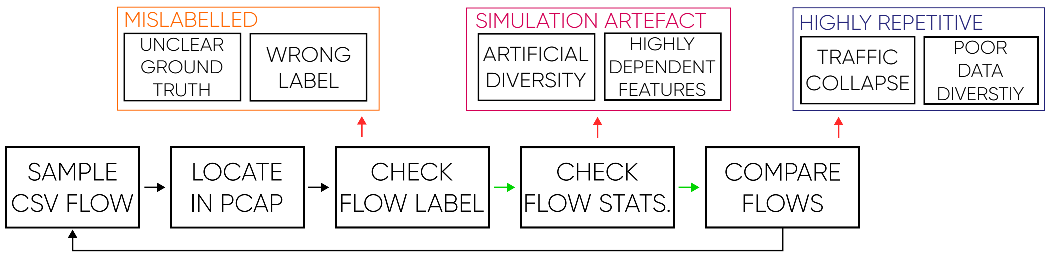

Figure 1: Overview of Manual Analysis Process. To facilitate easier discussion, note that we group our bad smells into three categories: ***Mislabelled***, ***Simulation Artefact*** and ***Highly Repetitive***.

we sample new flows until we can satisfactorily answer, indicated by the backwards arrow in Figure 1.

4.2. Automated Prevalence Analysis

Although a qualitative analysis is necessary, it is arduous and time-consuming. An automated process that can highlight problems quickly when generating a dataset would be highly beneficial. However, PCAP data is a complicated format. As the properties of each dataset are highly variable, it is difficult to verify all attacks across all network conditions. Instead, we design some heuristics for CSV data, with minimal reference to the original PCAPs.

We preprocess all features according to standard practises: we use a min-max scaler and convert categorical features to ordinal or one-hot encoded features. Our aim was to mimic how these datasets may be used by a researcher who hasn't checked the underlying data.

Mislabelled. We perform two tests for *mislabelled* traffic, corresponding to the *unclear ground truth* and *wrong label* smells respectively. Naive labelling algorithms based on IPs have a tendency to mislabel background traffic that are orthogonal to the mechanism of an attack as malicious, such as authentication traffic, discovery services, and advertising features. Thus, labeling decisions can become ambiguous or unclear. To address this, we maintain a list of ports related to well-known background services and flag a flow as *unclear* if its destination port feature, denoted as $F_{\text{Dst Port}}$, belongs to this set of ports (indicated by BG Ports[4]). Note that we base our ports on the datasets examined; services operating on these ports can still be abused by attackers, and the specific ports chosen may not be suitable for other datasets, leading to false positives. We quantify potentially *unclear* flows by calculating the ratio of flows sent to these ports to the total number of flows in the dataset. Thus, for given class C:

$$UGT_C = \frac{|F_{\text{Dst Port}} \in \text{BG Ports}|}{|C|} \qquad UGT_C \in [0,1] \qquad (1)$$

To estimate the number of flows with the *wrong label* smell, we use the Edited Nearest Neighbour Rule (ENN), proposed by Wilson et al. [80], [109]. ENN identifies a sample as close to a decision boundary if its label differs from the majority of its k-Nearest Neighbours. We modify the original ENN process by breaking ties in favour of the mislabelled class.

Setting $k = 4$, we define the majority class identified by ENN as $ENN(x)$ where $x \in C$. We then measure the degree of mislabelling via the percentage of elements of C misclassified by ENN, or:

$$WL_C = \frac{|\hat{C}|}{|C|} \text{ where } x \in \hat{C} \textit{ iff } \text{ENN}(x) \neq C, \;\; WL_C \in [0,1] \qquad (2)$$

Simulation Artefacts. To detect *highly dependent features*, we use a maximal feature efficacy process. Looking at each attack class C separately, we measure the F1 score of a random forest classifier distinguishing C from the background when trained on a single feature, F_i. Intuitively, if F_i is highly dependent, we expect an unreasonably high F1 score. Although this process can detect multiple artefacts, for brevity, we report only the most severe instance. Thus, we define HDF_C as:

$$HDF_C = MAX(F1(F_i))_i \quad HDF_C \in [0,1] \qquad (3)$$

4. BG Ports $= \{0, 53, 67, 68, 111, 123, 137, 161, 179, 389, 427, 520, 1723, 1900\}$

During manual analysis, *artifical diversity* was mostly seen when unstable network conditions caused high numbers of dropped or re-transmitted packets. We check that problematic packets remained within the bounds set out by prior work [37]. As this tended to be general across the network capture, we only report this in Section 5 when notable for a dataset. This is the only heuristic which requires access to the original PCAPs.

Highly Repetitive Traffic. Rather than classification complexity [50], we aim to measure data diversity independent of classification. Thus, our measures for the *traffic collapse* and *poor data diversity* smells use a two-stage clustered similarity process: first, we estimate the number of clusters, N, within class C via the Elbow method [102]. We then apply KMeans to assign each data point a cluster, C_i. Via cosine similarity, we measure similarity between randomly sampled pairs as:

$$CS_{C_i} = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \quad \text{where} \quad \mathbf{A} \sim C_i,\, \mathbf{B} \sim C_i, \\ CS_{C_i} \in [0,1] \tag{4}$$

We record the average CS_{C_i} between M[5] pairs from each cluster weighted by cluster size, expecting a score of approximately 1 for near-identical pairs. This provides quick insight into a class's cluster sizes as an approximate measure of data homogeneity, corresponding to our *poor data diversity* smell:

$$PDD_C = \sum_{i<N} \sum_{j<M} \frac{|C_i|}{|C|} \frac{CS_{C_i}}{MN} \qquad PDD_C \in [0,1] \tag{5}$$

For our *traffic collapse* bad smell, we wish to measure an egregious lack of data diversity, potentially caused by configuration issues. We repeat the above process, but measure the percentage of pairs from each cluster where CS_{C_i} exceeds a threshold value of 0.95[6], indicating that the flows are functionally identical. We report the maximum value across our clusters. Using Iverson brackets, we write this as:

$$TC_C = \max_i \left(\sum_{j<M} \frac{[CS_{C_i} > 0.95]}{M} \right) \qquad TC_C \in [0,1] \tag{6}$$

5. Results

5.1. Manual Analysis

We apply our manual analysis process to over 65 attack classes. As part of our efforts, we will release a comprehensive, open-access account that documents all problems we identified, helping researchers navigate experimental pitfalls. In this section, we demonstrate the scope of the problems we uncovered with examples.

5. We found scores to be converge consistently with $M \approx 100$

6. We select this value as corresponds to an angle of approximately $\frac{\pi}{10}$ between sampled flows, or 10% of maximal dissimilarity

Mislabelled. Mislabelling stems from design choices throughout the generation process. Researchers must be clear about what they are generating. However, this can be murky, such as in *UNSW NB15*, which used the predetermined 'strikes' of the IXIA PerfectStorm tool [54]. This causes *unclear ground truth*, such as the *Fuzzing* class containing dubious routing attacks. These have no associated CVE and simply alter minor aspects of the protocol, creating flows that are statistically identical to their benign equivalents. We also note that the definition for *UNSW NB15*'s *Generic* class, confusingly, involves block cipher vulnerabilities with no relationship to the dataset's contents [74].

Labelling flows via IPs and timestamps is challenging. Researchers must account for background traffic of the attacker network; naive logic may mislabel these flows. *TON_IoT* consistently treats DNS traffic as malicious; in the *DoS* and *XSS* classes, 55% and 28% of functioning flows are DNS requests, respectively. Labelling multi-stage attacks is complex, leading to errors. In *CIC 2018*, the *Infiltration* class misses entire attack stages, incorrectly labelling them as benign. Mislabelling can also occur during final processing steps. Again, in *CIC 2018*'s *Infiltration* class, several benign flows were duplicated and included, exacerbating the previous issue. In contrast with other datasets, *CTU-13* provides highly granular labelling, making it easier to discard mislabelled flows. Despite this, some problems persist, such as OS updates labelled as malicious adware and flows that appear to have been accidentally filtered from the network capture.

Aside from black-and-white errors, NIDS datasets are plagued by murky labelling, exacerbated by poor documentation. Although better than other datasets, we found discrepancies between the documentation [94] for *ISCX 2012* and the PCAPs. For example, *HTTP DoS* is reportedly executed using *Slowloris*, which overwhelms a server with incomplete HTTP requests. However, we found no evidence of these partial requests; instead, the attack consisted of generic GET requests.

Simulation Artefacts. Simulation artefacts can affect an entire dataset. Consider *UNSW NB15* whose features include the hosts' *time-to-live* values. This appears to inadvertently fingerprint operating systems and in, say, the *Exploits* class, the distribution of operating systems among attacked machines differs significantly from that of the machines receiving benign traffic. Consequently, this *highly dependent feature* simplifies classification. The ratio of TCP to UDP traffic between the benign/attack classes causes similar issues. Subtle choices, such as attackers targeting small webpages whilst benign users visit large webpages can bias features. In *CIC 17*, this results in highly discriminative *total packet length* features, even when unrelated to an attack's underlying behavior. Mistaken flow calculation can also cause artefacts: in *CTU-13 Scenario 4*, several hundred non-existent UDP flows with impossible characteristics are recorded due to mistaken processing. Alongside mislabelled NetBIOS traffic, these account for 99.9% of malicious *UDP Attempt* traffic.

Highly Repetitive. *Highly Repetitive* data undermines the ubiquitous train/validation/test pipeline. This prevents building meaningful holdout sets and, consequently, the generalisation abilities of ML classifiers, a primary research goal, are not examined. Our analysis reveals

TABLE 8: Results of Heuristic Measures (- indicates indeterminate due to small class size, * indicates clash with *FTP-BruteForce*, † indicates clash with *LOIC*, ‡ indicates clash with *Background*). We provide results for the primary class of each CTU-13 scenario, measured by packet volume, with more detail in the Appendix.

Dataset/Class	PDD_C	TC_C	WL_C	HDF_C	UGT_C
UNSW NB15					
Generic	**0.98**	0.92	0.0	**1.0**	**0.98**
Exploits	0.89	0.41	**0.10**	0.88	**0.37**
Fuzzers	0.93	0.6	**0.52**	0.66	**0.51**
DoS	0.91	0.48	**0.12**	0.90	**0.78**
Recon.	**0.95**	0.76	**0.38**	**0.95**	**0.84**
Analysis	0.93	0.54	**0.21**	0.89	**0.77**
Shellcode	**0.97**	**0.95**	**0.57**	0.69	0.0
Backdoor	0.91	0.48	0.0	0.91	**0.82**
Worms	0.94	0.46	**0.60**	0.78	0.0
ToN_IoT					
scanning	**0.97**	**0.95**	0.0	**0.99**	0.01
dos	**0.99**	**0.97**	0.0	**0.98**	0.03
ddos	**0.99**	**0.98**	0.0	**0.97**	**0.12**
mitm	0.83	0.85	**0.27**	0.73	**0.56**
xss	0.84	0.86	0.0	**0.97**	**0.27**
backdoor	**1.0**	**1.0**	**0.31**	**1.0**	0.0
injection	**0.95**	0.92	0.0	**0.98**	0.03
passwords	0.89	**1.0**	0.0	**0.99**	0.0
ransomware	0.83	0.91	0.05	0.84	0.0
Bot IoT					
DDoS	0.86	0.5	0.0	**0.98**	0.0
DoS	0.87	0.37	0.0	**0.99**	0.0
Recon.	**0.93**	0.53	0.01	**0.98**	0.0
Theft	0.89	0.67	0.06	**1.0**	0.0
CTU-13					
Neris 1	0.84	0.36	0.0	0.83	0.0
Neris 2	0.85	0.36	0.0	**0.96**	0.0
Rbot 1	**0.98**	**0.99**	**0.18**‡	**1.0**	0.0
Rbot 2	0.87	0.7	0.0	0.0	**0.98**
Virut 1	0.94	0.67	0.0	0.8	0.0
Donbot	**0.97**	**1.0**	0.0	**1.0**	0.0
Sogou	-	-	-	-	-
Murlo	**0.96**	0.84	**1.0**	0.87	0.0
Neris 3	0.9	**0.97**	0.0	0.87	0.0
Rbot 3	**0.99**	**0.99**	0.0	**1.0**	0.0
Rbot 4	**0.98**	**0.96**	0.01	**1.0**	0.0
NSIS	0.78	0.68	0.02	0.88	0.0
Virut 2	0.88	**0.95**	0.0	0.9	0.0
CIC-IDS 2017					
Portscan	**0.99**	**0.99**	0.0	**0.98**	0.0
DoS Hulk	**0.98**	**0.98**	0.0	**1.0**	0.0
FTP-Patator	**0.98**	**0.98**	0.0	**0.99**	0.0
SSH-Patator	**1.0**	**1.0**	0.0	**0.98**	0.0
DDoS	**0.98**	0.94	0.0	**0.99**	0.0
Bot	**0.98**	**1.0**	0.01	**1.0**	0.0
Slowloris	**0.97**	**1.0**	0.0	**0.98**	0.0
Slowhttptest	0.88	0.53	0.0	**0.96**	0.0
GoldenEye	**0.95**	0.68	0.0	**1.0**	0.0
Infil.	0.92	0.65	**0.81**	0.75	0.0
Brute Force	**0.99**	0.93	0.06	0.91	0.0
XSS	0.78	0.48	**0.35**	0.93	0.0
SQL	-	-	**0.63**	-	0.0
Heartbleed	-	-	**0.18**	-	0.0
CIC-IDS 2018					
Infil.	0.67	0.32	**0.65**	0.63	**0.32**
Bot	**0.99**	**0.99**	0.0	**0.99**	0.0
Hulk	**0.98**	**0.99**	0.0	**1.0**	0.0
Slowloris	0.83	0.89	0.0	**0.99**	0.0
SSH-Bruteforce	**0.99**	**0.99**	0.0	**1.0**	0.0
FTP-BruteForce	**0.99**	**1.0**	0.0	**0.99**	0.0
LOIC	**0.96**	**0.99**	0.0	**1.0**	0.0
LOIC-UDP	**0.96**	0.82	**0.16**†	**0.99**	0.0
HOIC	**0.98**	0.88	0.0	**1.0**	0.0
GoldenEye	0.93	**0.99**	0.0	**1.0**	0.0
SlowHTTPTest	**0.99**	**1.0**	**0.56***	**1.0**	0.0
XSS	0.90	**1.0**	0.05	0.83	0.0
Web	0.77	**1.0**	**0.21**	0.79	0.04
SQL	0.86	0.85	**0.21**	0.77	0.0
ICSX 2012					
BruteForce	**0.99**	**0.99**	0.02	**0.96**	0.0
SSH	**0.98**	**1.0**	0.0	0.93	0.0
nmap	**0.93**	**1.0**	0.04	0.78	0.02
IRC	**0.96**	**1.0**	0.0	0.70	0
Other	0.75	0.37	**0.35**	0.56	**0.13**

the extent of this issue, exemplified by *UNSW NB15*'s *Reconnaissance* class. The primary protocol in this class is Portmap at roughly 80.5% with very little variation between flows. Similarly, 97.8% of malicious flows in *CTU-13*'s *Scenario 5* are related to a basic nmap scan.

Classifying volumetric attacks is straightforward. Without exception, DoS attacks were launched against static targets. This design choice produce millions of near identical flows. This is particularly noteworthy in *Bot IoT*, as only 0.00013% of traffic is benign, and the overwhelming majority of attack traffic is volumetric. In Section 6.1, we demonstrate how lack of diversity rewards overfitting models without evaluating their generalisability.

When this smell coexists with other smells, it can be masked, potentially misleading researchers into overestimating the classification challenge presented by a dataset. For instance, in *CIC 17*, the minority classes *SQL Injection* and *Heartbleed* suffer from the *mislabelled* and *artificial diversity* smells respectively. Remediating these issues, it is possible to achieve perfect accuracy classifying these attacks even with simple models.

Often, data was not adequately audited. Launching attacks against closed ports is common, including the *Ton_IoT backdoor*, *Bot IoT Theft* and *CIC 18 FTP-Bruteforce* classes. Attacks were also launched against secure services, such as *CIC 17*'s *DoS Goldeneye* and *CIC 18*'s *XSS* classes. Although detecting failed attacks is potentially good, these should be explicitly labelled as failures. During our analysis in Section 3 we did not encounter a single paper that appeared to be aware that they were working with failed attack data, leading to overly optimistic interpretations of their results.

5.2. Automated Analysis

In Table 8, we report the results of our heuristic measures for 68 classes, bolding any results that we feel indicate serious data design issues. We also run our measures on the multi-dimensional ODDS collection [82], a set of tabular benchmark datasets for anomaly detection (including data taken from *KDD Cup*). The ODDS datasets are mostly non-synthetic, providing a comparison between synthetic network data and real-world data. We present these results in Table 9.

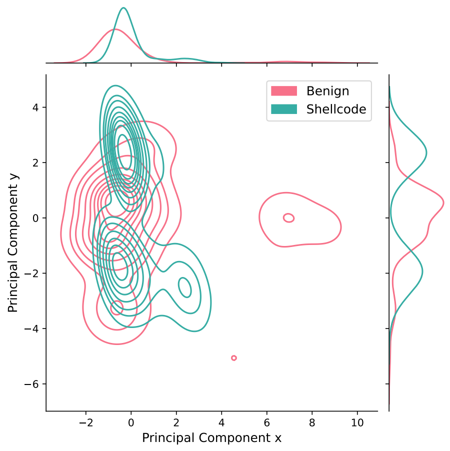

Figure 2: Overlap of Shellcode class with Benign traffic in UNSW NB15. We perform principal component analysis to represent each class in two dimensions.

Mislabelled. Table 8 shows that a minority of classes exhibit significant labelling issues, according to our WL_C and UGT_C measures. Although complex data could cause points to lie on the decision boundary and lead to high scores, our manual analysis refutes this. Instead, mislabelled and contextually benign data cause this overlap, demonstrated in Figure 2, where *UNSW NB15*'s *Shellcode* and *Benign* classes coincide heavily due to unrelated DNS traffic. This also occurs in *CIC 17*'s *SQL injection* class, where many flows consist of simple, benign GET requests. Even simple tests, such as our UGT_C, highlight severe issues with unclear attack classes. For instance, in *UNSW NB15*, common sense checks would expose the *Generic* class's issues, which also predominantly consists of DNS records. However, we also note that our set of *BG Ports* generalised poorly to *CTU-13*. As a result, naive application of our UGT_C measure produced a false positive rate of approximately 20%, as *CTU-13* contains malicious DNS and ICMP traffic.

Highly Repetitive. The results of PDD_C show that the attack traffic of these datasets have low diversity. Worryingly, many classes achieve extremely high TC_C scores, implying that classification is equivalent to identifying a small number of flows or, potentially, a single flow. Figure 3 shows an example of this problem.

Such patterns severely degrade the ubiquitous train/validation/test pipeline, preventing the formation of meaningful holdout sets and favoring models that overfit. Consequently, the generalization abilities of ML classifiers — a crucial aspect of NIDS research — are not effectively examined.

Simulation Artefacts. According to our HDF_C measure, *highly dependent features* are ubiquitous across the datasets tested. Our one-feature baseline frequently separated attack classes from the background traffic perfectly. Near-perfect scores were also common. Upon examining

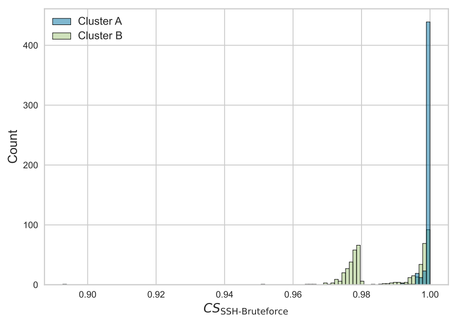

Figure 3: CS_C of two clusters of *CIC 18 SSH-BruteForce* class, partially launched against a closed port. Almost all sampled flows are identical in Cluster A.

these features, we observe few connections to the attack's underlying behavior. In *CIC 18*, our baseline achieves a near-perfect F1 score classifying *DoS Hulk* traffic using the *FWD Init Win Bytes* feature. This attack and feature also appears in *CIC 17*. However, here, our *FWD Init Win Bytes* random forest fails to correctly classify a single *DoS Hulk* flow. This discrepancy highlights the arbitrary nature of the connection between the *DoS Hulk* class and the *FWD Init Win Bytes* feature.

Measuring the impact of artefacts that increase diversity, such as our *artificial diversity* smell, is difficult. A notable example exists in *Bot IoT* and *CIC 17*, where the rate of TCP retransmissions exceeds 34% and 15%, respectively. In standard traffic, this rate typically does not exceed 3% [37]. This disparity introduces unrealistic variability in the flow's features, artificially complicating the data.

ODDS Dataset Collection. For comparison, we run our non-network specific metrics — WL_C, PDD_C, TC_C and HDF_C — on the ODDS collection. We limit our analysis to datasets with more than 2500 background samples[7]. The complexity of these datasets varies widely. State-of-the-art methods achieve F1 scores between ~ 0.2 and 1.0 [45], [64], [93]. Our results are collected in Table 9.

With the exception of the *shuttle* dataset and some WL_C measures, none of the ODDS datasets attain the extreme scores of our NIDS datasets. We emphasise that these measures are heuristics and, without the original data to analyse, we can't draw conclusions based on features alone. For instance, our results for WL_C correlate inversely with anomaly detection score [93] and the *mammography* or *speech* datasets may simply be challenging benchmarks. However, contrasting the scores attained on these non-synthetic datasets and our synthetic NIDS datasets, we see a marked difference: on a class-by-class basis, our metrics flags issues at a rate three times higher on the NIDS datasets than the ODDS collection. Strikingly, the best scores attained by our metrics occur on ODDS datasets.

6. Recommendations

We conclude with some recommendations for using NIDS data, in Sections 6.1–6.2, and developing NIDS datasets, in Section 6.3. Whilst building a 'perfect' dataset is ambitious, these suggestions could improve data quality and minimise design issues.

7. We also exclude the *Mulcross* dataset (as the official link was dead at the time of this experiment) as well as the *KDD Cup* based datasets.

TABLE 9: Results of Heuristic Measures on ODDS

Dataset/Measure	PDD_C	TC_C	WL_C	HDF_C
annthyroid	0.89	0.51	**0.6**	0.88
cardio	0.81	0.37	0.12	0.81
cover	0.91	0.31	0.0	0.92
mammography	0.92	0.67	**0.3**	0.78
mnist	0.53	0.0	0.08	0.77
optdigits	0.86	0.06	0.0	0.49
pendigits	0.93	0.68	0.02	0.56
satellite	**0.95**	0.81	0.06	0.8
satimage	0.88	0.75	0.07	0.94
shuttle	**0.98**	**0.95**	0.0	**0.98**
speech	0.80	0.0	**0.3**	0.53
thyroid	0.85	0.3	**0.19**	0.92

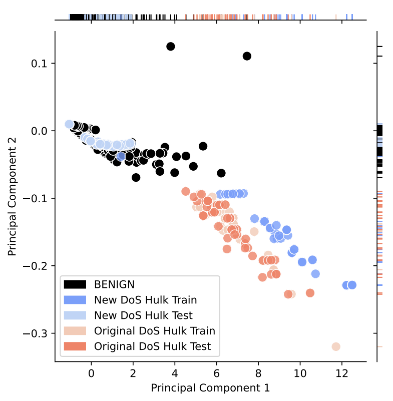

Figure 4: Overlap between train/test sets on *CIC 17* via standard evaluation pipeline (Original) vs. our bespoke data (New). Note that the train/test sets of *CIC 17* overlap almost entirely.

6.1. Testing for Generalisation Explicitly

The oft repeated advantage of machine learning in security is that models can generalise to unseen attacks. Whilst somewhat straightforward in other domains [4], [104], this goal is poorly defined in intrusion detection: along what axes should models generalise? What does it mean for an attack to be ‘different’ to another? If generalisation is the goal, papers should explicitly define success conditions and datasets should be used in manner that supports this aim. Instead, most work relying on these datasets measure generalisation via a typical train/test pipeline. Given the design choices of the analysed NIDS datasets, this is often equivalent to training and evaluating models on flows from the **same** attack, a form of data leakage. For the reasons outlined in this paper and others [23], this is not enough to demonstrate model generalisation.

Given existing NIDS data design, we recommend avoiding using training and test attack data from the same class and dataset, as recommended previously [8], [23], [83] and undertaken by some work [11]. Evaluating models based on their cross-class/dataset performance aligns more closely with the intended use case of machine learning-based NIDS: generalising to unseen attack data. Even better, by using prior work in synthetic NIDS data generation [28], [29], [58], it is possible to test whether model architectures can generalise to *arbitrary* attack traffic. Using DetGen [28], we generate malicious data to show how synthetic data can be used like this.

Consider detecting DoS traffic to demonstrate generalisation between different network bandwidths and web page sizes. Based on our analysis of *CIC 17*, the *DoS Hulk* class had *poor data diversity* with *highly dependent features* due to the fixed network conditions and target web page. We generate arbitrary DoS data by randomising webpage size — between 1 and 10MB — and attacker bandwidth — between 1Mbps and 50Mbps. In contrast to *CIC 17*’s single example, we collect volumetric DoS attack data for 60 combinations of bandwidth limit and web page size, which we inject into *CIC 17*. Although this process is artificial, it is similar to domain randomisation in computer vision, where arbitrary synthetic data has been leveraged to improve model generalisation [103]. The increase in diversity is reflected in our heuristics: compared with *CIC 17*’s *DoS Hulk* class, this process produces data that fares considerably better, scoring 0.89 and 0.62 on our PDD_C and TC_C measures, respectively.

By producing such traffic, researchers can query a far larger breadth of an attack’s possible data distributions during their testing and evaluation. Critically, this greater test data diversity allows for stronger generalisation claims, better probing of model failures — as shown by Clausen et al. [27] — and weakens the efficacy of naive application of flow statistics, as we show in Figure 4. Figure 4 also shows the inability of a standard train/test split on an unmodified *CIC 17* to evaluate generalisation. In contrast, by considering training and test data from different runs of our generation process, naive application of flow statistics results in considerable less overlap between the sets.

6.2. Improving Feature Selection

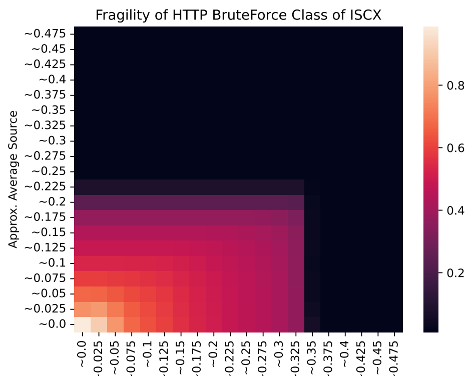

Figure 5: Heatmap of baseline model’s F1 scores as simple perturbations are applied to the *Total Source Bytes* and *Total Destination Bytes* features. Perfect score is achieved in bottom left (unperturbed). Feature values normalised between $[0, 1]$.

In general, we caution against directly using the feature sets provided with these datasets. It is widely understood that feature engineering is a vital step in the machine learning pipeline. However, NIDS research often completely avoids this process, instead using the provided features by default. These features rarely capture dependencies between flows, necessitating sequential machine learning methods, such as those investigated by Corsini et al. [31]. As far as we are aware, there is no established methodology for associating features with attacks. Even recent attempts to standardise the feature sets amongst NIDS datasets typically focus on the feasibility of collecting such features, rather than the relationship between the chosen features and attacks [11], [85].

As *highly dependent features* can easily occur via poor feature selection, high F1 scores are not enough to validate a NIDS model's performance, as highlighted by Jacobs et al. [52]. We recommend ***justifying*** model performance — using techniques described by Nadeem et al. [77] — connecting important features to an attack's properties across multiple diverse interactions to validate model efficacy. Papers should explicitly state this connection. If a feature is inexplicably highly discriminative then perhaps it should be discarded.

Following Geirhos et al.'s suggestion that adversarial samples are useful for understanding such shortcut features [42], we recommend researchers visualise egregiously dependent features by analysing a baseline model's 'fragility' — i.e., the ease with which unguided perturbations cause samples to cross the decision boundary. We demonstrate this for the *HTTPWeb* class of *ISCX 2012* in Figure 5. A simple random forest achieves a near-perfect F1 score separating *HTTPWeb* from the background. However, the most important features — *Total Source Bytes* and *Total Destination Bytes* — are attacker controlled, unrelated to the attack specification and, thus, potentially shortcut artefacts. Unguided perturbations on these features quickly reduce the model's F1 score without changing the semantic properties, indicating shortcut learning has taken place.

6.3. Towards Better Dataset Design

It is an ongoing process to consider how our smells should be best addressed when creating a dataset. However, based on our analysis, we can recommend some improvements.

Whilst the advice in Sections 6.1 & 6.2 is primarily geared towards dataset users, we believe they are also relevant for dataset creators. Namely, creators should include multiple diverse test scenarios for each class, whilst documenting their differences. For instance, tests sets may differ according to spacial or temporal factors, as in TESSERACT [4]. Creators should also highlight potential shortcut artefacts via analysis of baseline models trained naively on flow statistics.

A well-formed dataset should focus on a specific class of attacks or tactics, drawing on vulnerability and adversary analysis [46], [67], [98]. Many published classifiers have lofty ambitions, with model architectures trained and evaluated on all attacks or anomalies within a dataset, instead of a more focused goal. We believe this is heavily influenced by the design of current benchmark datasets, which often contain several disconnected attacks. We also recommend that datasets should include the normal usage of the service being attacked. Otherwise, models may simply be learning incidental aspects of the attack generation process, rather than distinguishing normal and abnormal behaviours.

In NIDS, there is a large gap between the feature and problem spaces. We believe this contributes heavily to labelling issues. For, say, images, converting features into their original source data is trivial and researchers can inspect data easily. In comparison, locating flows in PCAPs is painful. Berkeley Packet Filters [72] can filter PCAPs in a flexible manner and useful filters that isolate malicious from background traffic or separate network services would allow researchers to bridge this gap and should be a critical part of a NIDS dataset, alongside documentation detailing the source of the traffic. We note that CTU-13 appeared to be labelled similarly, simplifying our analysis process massively.

7. Related Work

Investigating issues with NIDS data is a well-established area and we build on many prior works. There are several NIDS surveys and overviews that highlight issues: Kenyon et al. [53] critique poor data provenance and simplistic synthetic models; Catillo et al. [24] catalogue several questionable practises in NIDS data, including shortcut artefacts and labelling issues; Apruzzese et al. [9] et al. provide a pragmatic assessment of machine learning for NIDS and recommend using multiple datasets for evaluation; Cordero et al. [29] provide a dataset overview when presenting their synthetic generation framework. These works typically discuss issues at a high level, rarely highlighting specific problems with specific datasets or quantitatively measuring issues as our work does. In contrast, Jacobs et al. [52] identify specific shortcut artefacts in *CIC 17* and *UNSW* via a process similar to HDF_C.

Other works provide more specific but limited NIDS dataset analysis. Analyses of single datasets exist for old, outdated datasets, such as Tavalee et al.'s criticism of *KDD Cup 1999* [3], [101]. In recent criticism, Liu and Engelen et al. [39], [66] provide analyses of *CIC 17/18* and discuss their shortcomings, however, the primary contribution of their work is the discovery of miscalculated flow statistics and labelling issues. Peterson et al. [81] provide a limited overview of pitfalls in *Bot IoT* and Catillo et al. [23] critique public datasets, centering their discussion on the poor transferability of classifiers between datasets.

Despite this prior research, to our knowledge, this is the first paper to present a methodology for uncovering design problems with NIDS datasets, to provide indicators of such issues and to provide a systematic overview of the pitfalls and problems that may be encountered by those who use these datasets. Previously, dataset evaluation frameworks such as that proposed by Gharib et al. [43] consist of simple checks, with no guarantee of the quality of the data that satisfy these criteria. NIDS datasets surveys often propose ways of mitigating problems, such as Ring et al. [83], who suggest using multiple datasets to evaluate performance. Again, these surveys rarely highlight specific problems as our work does.

Similar analyses do exist in other domains, such as time-series anomaly data [110] and image classification [104], as well as more general analyses of data quality in machine learning pipelines [84] or machine learning applied to security [12]. We also note that the term 'data smell' [95] is already used to refer to minor problems with datasets, such as formatting issues.

8. Caveats & Limitations

Despite these critiques, we emphasise that imperfect NIDS datasets are still useful tools for researchers and can be completely sufficient for certain tasks. For example, based on their usage of *CIC 17*, we see little reason why the covert communication channels of Chen et al. [26] or the OS fingerprinting of Holland et al. [51] would be impacted by bad data design smells. For NIDS specifically, these datasets do contain valid attack traces which can be properly utilised by research. However, we maintain that thorough understanding is needed to assess a dataset's suitability for a given task.

Whilst we try to be as thorough as possible, our analysis may be limited. Some of our criteria are inherently subjective, reflecting the heterogeneity of these datasets. It is difficult to formulate strict criteria that generalise across all current NIDS datasets, and our suggested heuristics must be applied sensibly to prevent false positives. Whilst we assigned multiple authors to cross-reference our findings, we did not contact the dataset authors for verification, performing this analysis ourselves.

9. Conclusion

Via an in-depth analysis of seven popular network intrusion datasets, we identify six data design smells. We find that insufficient data auditing is general across the field of network intrusion research and that these smells could severely bias downstream research.

We hope that the research community takes these issues seriously. Across many facets, synthetic data can provide advantages over real-world data such as complete ground truths, high generative control and repeatability. None of these are utilised by the current crop of datasets and static datasets are still the de facto standard. Further research into producing quality benchmark datasets via synthetic data generation techniques is one potential approach to ameliorate these issues.

Our goal was to inform researchers about dataset contents and limitations. Moreover, our work provides insight into the choices necessary to improve NIDS dataset design. By shedding light on these issues, we hope to stimulate discussions and encourage the community to reassess the suitability and reliability of these datasets for network intrusion research. The paucity of quality, public data is a major problem for the research community as a whole and continued critical evaluation of network data can only be good.

Data Availability

Our heuristics tooling, alongside our experiment notebooks, are available at https://github.com/DataBadSmells.

Acknowledgements

We gratefully acknowledge support of the UKRI under grant EP/T027037/1, ARM Limited, the Research Fund KU Leuven, and the Cybersecurity Research Program Flanders. This work has benefited from feedback from several people, including Gudmund Grov, Vera Rimmer, Henry Clausen, Marc Juarez, Stuart Wray and especially the EuroS&P reviewers.

References

[1] CICFlowMeter Repo. https://github.com/ahlashkari/CICFlowMeter. Accessed: 2021-12-15.

[2] IBM QRadar. https://www.ibm.com/qradar. Accessed: 2023-06-02.

[3] KDD Cup 1999 Dataset. http://kdd.ics.uci.edu/\databases/kddcup99/kddcup99.html. Accessed: 2022-03-07.

[4] Tesseract: Eliminating experimental bias in malware classification across space and time.

[5] Maged Abdelaty, Sandra Scott-Hayward, Roberto Doriguzzi-Corin, and Domenico Siracusa. GADoT: GAN-based Adversarial Training for Robust DDoS Attack Detection. In *2021 IEEE Conference on Communications and Network Security (CNS)*, pages 119–127. IEEE, 2021.

[6] Bushra A Alahmadi, Enrico Mariconti, Riccardo Spolaor, Gianluca Stringhini, and Ivan Martinovic. BOTection: Bot Detection by Building Markov Chain Models of Bots Network Behavior. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, pages 652–664, 2020.

[7] Huda Ali Alatwi and Charles Morisset. Realism versus Performance for Adversarial Examples Against DL-based NIDS. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, pages 1549–1557, 2023.

[8] Giuseppina Andresini, Feargus Pendlebury, Fabio Pierazzi, Corrado Loglisci, Annalisa Appice, and Lorenzo Cavallaro. Insomnia: Towards Concept-drift Robustness in Network Intrusion Detection. In *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, pages 111–122, 2021.

[9] Giovanni Apruzzese, Pavel Laskov, and Johannes Schneider. SoK: Pragmatic Assessment of Machine Learning for Network Intrusion Detection. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 592–614. IEEE, 2023.

[10] Giovanni Apruzzese, Pavel Laskov, and Aliya Tastemirova. SoK: The Impact of Unlabelled Data in Cyberthreat Detection. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pages 20–42. IEEE, 2022.

[11] Giovanni Apruzzese, Luca Pajola, and Mauro Conti. The Cross-evaluation of Machine Learning-based Network Intrusion Detection Systems. *IEEE Transactions on Network and Service Management*, 19(4):5152–5169, 2022.

[12] Daniel Arp, Erwin Quiring, Feargus Pendlebury, Alexander Warnecke, Fabio Pierazzi, Christian Wressnegger, Lorenzo Cavallaro, and Konrad Rieck. Dos and Don'ts of Machine Learning in Computer Security. In *Proc. of the USENIX Security Symposium*, 2022.

[13] Bradley Ashmore and Lingwei Chen. HOVER: Homophilic Oversampling via Edge Removal for Class-Imbalanced Bot Detection on Graphs. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 3728–3732, 2023.

[14] Dina Ayesha S and Siddique AB. FS3: Few-Shot and Self-Supervised Framework for Efficient Intrusion Detection in Internet of Things Networks. In *Proceedings of the 39th Annual Computer Security Applications Conference*, pages 138–149, 2023.

[15] Maximilian Bachl, Fares Meghdouri, Joachim Fabini, and Tanja Zseby. Sparseids: Learning Packet Sampling with Reinforcement Learning. In *2020 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9. IEEE, 2020.

[16] Kent Beck, Martin Fowler, and Grandma Beck. Bad Smells in Code. *Refactoring: Improving the design of existing code*, 1(1999):75–88, 1999.

[17] Lucas Beyer, Olivier J Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we Done with ImageNet? *arXiv preprint arXiv:2006.07159*, 2020.

[18] Siddharth Bhatia, Arjit Jain, Pan Li, Ritesh Kumar, and Bryan Hooi. Mstream: Fast Anomaly Detection in Multi-aspect Streams. In *Proceedings of the Web Conference 2021*, pages 3371–3382, 2021.

[19] Siddharth Bhatia, Arjit Jain, Shivin Srivastava, Kenji Kawaguchi, and Bryan Hooi. Memstream: Memory-based Streaming Anomaly Detection. In *Proceedings of the ACM Web Conference 2022*, pages 610–621, 2022.

[20] Siddharth Bhatia, Mohit Wadhwa, Kenji Kawaguchi, Neil Shah, Philip S Yu, and Bryan Hooi. Sketch-based Anomaly Detection in Streaming Graphs. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 93–104, 2023.

[21] Tim M Booij, Irina Chiscop, Erik Meeuwissen, Nour Moustafa, and Frank TH den Hartog. ToN_IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Data Sets. *IEEE Internet of Things Journal*, 9(1):485–496, 2021.

[22] Valentín Carela-Español, Tomasz Bujlow, and Pere Barlet-Ros. Is our Ground-truth for Traffic cClassification Reliable? In *International Conference on Passive and Active Network Measurement*, pages 98–108. Springer, 2014.

[23] Marta Catillo, Andrea Del Vecchio, Antonio Pecchia, and Umberto Villano. A Critique on the Use of Machine Learning on Public Datasets for Intrusion Detection. In *International Conference on the Quality of Information and Communications Technology*, pages 253–266. Springer, 2021.

[24] Marta Catillo, Antonio Pecchia, and Umberto Villano. Machine Learning on Public Intrusion Datasets: Academic Hype or Concrete Advances in NIDS? In *2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S)*, pages 132–136. IEEE, 2023.

[25] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: Synthetic Minority over-Sampling Technique. *J. Artif. Int. Res.*, 16(1):321–357, jun 2002.

[26] Olga Chen, Aaron D Jaggard, Catherine Meadows, and Michael C Shlanta. NExtSteP: An Extensible Testbed for Network Covert Channels. In *2020 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9. IEEE, 2020.

[27] Henry Clausen and David Aspinall. Examining Traffic Microstructures to Improve Model Development. In *2021 IEEE Security and Privacy Workshops (SPW)*, pages 19–24. IEEE, 2021.

[28] Henry Clausen, Robert Flood, and David Aspinall. Traffic Generation using Containerization for Machine Learning. In *2019 Workshop on DYnamic and Novel Advances in Machine learning and Intelligent Cyber Security*, pages 1–12, 2019.

[29] Carlos Garcia Cordero, Emmanouil Vasilomanolakis, Aidmar Wainakh, Max Mühlhäuser, and Simin Nadjm-Tehrani. On Generating Network Traffic Datasets with Synthetic Attacks for Intrusion Detection. *ACM Transactions on Privacy and Security (TOPS)*, 24(2):1–39, 2021.

[30] Andrea Corsini and Shanchieh Jay Yang. Are Existing Out-Of-Distribution Techniques Suitable for Network Intrusion Detection? In *2023 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9. IEEE, 2023.

[31] Andrea Corsini, Shanchieh Jay Yang, and Giovanni Apruzzese. On the Evaluation of Sequential Machine Learning for Network Intrusion Detection. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, pages 1–10, 2021.

[32] Alec F Diallo and Paul Patras. Adaptive Clustering-based Malicious Traffic Classification at the Network Edge. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2021.

[33] Alec F Diallo and Paul Patras. Sabre: Cutting through Adversarial Noise with Adaptive Spectral Filtering and Input Reconstruction. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 76–76. IEEE Computer Society, 2023.

[34] Yutao Dong, Qing Li, Kaidong Wu, Ruoyu Li, Dan Zhao, Gareth Tyson, Junkun Peng, Yong Jiang, Shutao Xia, and Mingwei Xu. HorusEye: A Realtime IoT Malicious Traffic Detection Framework using Programmable Switches. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 571–588, 2023.

[35] R. Doriguzzi-Corin. LUCID Source Code. https://github.com/doriguzzi/lucid-ddos, 2020.

[36] Roberto Doriguzzi-Corin, Stuart Millar, Sandra Scott-Hayward, Jesus Martinez-del Rincon, and Domenico Siracusa. LUCID: A Practical, Lightweight Deep Learning Solution for DDoS Attack Detection. *IEEE Transactions on Network and Service Management*, 17(2):876–889, 2020.

[37] Nandita Dukkipati, Matt Mathis, Yuchung Cheng, and Monia Ghobadi. Proportional Rate Reduction for TCP. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 155–170, 2011.

[38] Laurens D'hooge, Miel Verkerken, Bruno Volckaert, Tim Wauters, and Filip De Turck. 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, 2022.

[39] Gints Engelen, Vera Rimmer, and Wouter Joosen. Troubleshooting an Intrusion Detection Dataset: the CICIDS2017 Case Study. In *2021 IEEE Security and Privacy Workshops (SPW)*, pages 7–12. IEEE, 2021.

[40] Filipe Falcão, Tommaso Zoppi, Caio Barbosa Viera Silva, Anderson Santos, Baldoino Fonseca, Andrea Ceccarelli, and Andrea Bondavalli. Quantitative Comparison of Unsupervised Anomaly Detection Algorithms for Intrusion Detection. In *Proceedings of the 34th ACM/SIGAPP symposium on applied computing*, pages 318–327, 2019.

[41] Sebastian Garcia, Martin Grill, Jan Stiborek, and Alejandro Zunino. An Empirical Comparison of Botnet Detection Methods. *computers & security*, 45:100–123, 2014.

[42] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut Learning in Deep Neural Networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[43] Amirhossein Gharib, Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. An Evaluation Framework for Intrusion Detection Dataset. In *2016 International Conference on Information Science and Security (ICISS)*, pages 1–6. IEEE, 2016.

[44] Mateusz Gniewkowski, Henryk Maciejewski, Tomasz Surmacz, and Wiktor Walentynowicz. Sec2vec: Anomaly Detection in HTTP Traffic and Malicious URLs. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, pages 1154–1162, 2023.

[45] Parikshit Gopalan, Vatsal Sharan, and Udi Wieder. Pidforest: Anomaly Detection via Partial Identification. *Advances in Neural Information Processing Systems*, 32, 2019.

[46] William G Halfond, Jeremy Viegas, Alessandro Orso, et al. A Classification of SQL-injection Attacks and Countermeasures. In *Proceedings of the IEEE international symposium on secure software engineering*, volume 1, pages 13–15. IEEE, 2006.

[47] Dongqi Han, Zhiliang Wang, Wenqi Chen, Ying Zhong, Su Wang, Han Zhang, Jiahai Yang, Xingang Shi, and Xia Yin. Deepaid: Interpreting and Improving Deep Learning-based Anomaly Detection in Security Applications. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3197–3217, 2021.

[48] Zijun Hang, Yuliang Lu, Yongjie Wang, and Yi Xie. Flow-MAE: Leveraging Masked AutoEncoder for Accurate, Efficient and Robust Malicious Traffic Classification. In *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*, pages 297–314, 2023.

[49] Christoph Hardegen, Mike Petersen, Chukwuebuka Ezelu, Timo Geier, Sebastian Rieger, and Ulrich Buehler. A Hierarchical Architecture and Probabilistic Strategy for Collaborative Intrusion Detection. In *2021 IEEE Conference on Communications and Network Security (CNS)*, pages 128–136. IEEE, 2021.

[50] Tin Kam Ho and Mitra Basu. Complexity measures of supervised classification problems. *IEEE transactions on pattern analysis and machine intelligence*, 24(3):289–300, 2002.

[51] Jordan Holland, Paul Schmitt, Nick Feamster, and Prateek Mittal. New Directions in Automated Traffic Analysis. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3366–3383, 2021.

[52] Arthur S Jacobs, Roman Beltiukov, Walter Willinger, Ronaldo A Ferreira, Arpit Gupta, and Lisandro Z Granville. AI/ML for Network Security: The Emperor has no Clothes. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 1537–1551, 2022.

[53] Anthony Kenyon, Lipika Deka, and David Elizondo. Are Public Intrusion Datasets Fit for Purpose Characterising the State of the Art in Intrusion Event Datasets. *Computers & Security*, 99:102022, 2020.

[54] Keysight. PerfectStorm Data Generation Tool. https://www.keysight.com/gb/en/products/network-test/network-test-hardware/perfectstorm.html, 2024.

[55] Isaiah J King, Xiaokui Shu, Jiyong Jang, Kevin Eykholt, Taesung Lee, and H Howie Huang. EdgeTorrent: Real-time Temporal Graph Representations for Intrusion Detection. In *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*, pages 77–91, 2023.

[56] Nickolaos Koroniotis, Nour Moustafa, Elena Sitnikova, and Benjamin Turnbull. Towards the Development of Realistic Botnet Dataset in the Internet of Things for Network Forensic Analytics: Bot-IoT Dataset. *Future Generation Computer Systems*, 100:779–796, 2019.

[57] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (Canadian Institute for Advanced Research). *URL http://www. cs. toronto. edu/kriz/cifar. html*, 5(4):1, 2010.

[58] Max Landauer, Maximilian Frank, Florian Skopik, Wolfgang Hotwagner, Markus Wurzenberger, and Andreas Rauber. A Framework for Automatic Labeling of Log Datasets from Model-driven Testbeds for HIDS Evaluation. In *Proceedings of the 2022 ACM Workshop on Secure and Trustworthy Cyber-Physical Systems*, pages 77–86, 2022.

[59] Maxime Lanvin, Pierre-François Gimenez, Yufei Han, Frédéric Majorczyk, Ludovic Mé, and Eric Totel. Towards Understanding Alerts Raised by Unsupervised Network Intrusion Detection Systems. In *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*, pages 135–150, 2023.

[60] Yann LeCun. The MNIST Database of Handwritten Digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[61] Laetitia Leichtnam, Eric Totel, Nicolas Prigent, and Ludovic Mé. Forensic analysis of network attacks: Restructuring security events as graphs and identifying strongly connected sub-graphs. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 565–573. IEEE, 2020.

[62] Laetitia Leichtnam, Eric Totel, Nicolas Prigent, and Ludovic Mé. Sec2graph: Network attack detection based on novelty detection on graph structured data. In *Detection of Intrusions and Malware, and Vulnerability Assessment: 17th International Conference, DIMVA 2020, Lisbon, Portugal, June 24–26, 2020, Proceedings 17*, pages 238–258. Springer, 2020.

[63] Yangmin Li, Xinhang Yuan, and Wengen Li. An extreme semi-supervised framework based on transformer for network intrusion detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4204–4208, 2022.

[64] Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. Copod: copula-based outlier detection. In *2020 IEEE international conference on data mining (ICDM)*, pages 1118–1123. IEEE, 2020.

[65] Richard Lippmann, Joshua W Haines, David J Fried, Jonathan Korba, and Kumar Das. The 1999 DARPA Off-line Intrusion Detection Evaluation. *Computer networks*, 34(4):579–595, 2000.

[66] Lisa Liu, Gints Engelen, Timothy Lynar, Daryl Essam, and Wouter Joosen. Error Prevalence in NIDS Datasets: A Case Study on CIC-IDS-2017 and CSE-CIC-IDS-2018. In *2022 IEEE Conference on Communications and Network Security (CNS)*, pages 254–262. IEEE, 2022.

[67] Miao Liu, Boyu Zhang, Wenbin Chen, and Xunlai Zhang. A Survey of Exploitation and Detection Methods of XSS Vulnerabilities. *IEEE access*, 7:182004–182016, 2019.

[68] Gabriel Maciá-Fernández, José Camacho, Roberto Magán-Carrión, Pedro García-Teodoro, and Roberto Therón. UGR '16: A New Dataset for the Evaluation of Cyclostationarity-based Network IDSs. *Computers & Security*, 73:411–424, 2018.

[69] Matthew V Mahoney and Philip K Chan. An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection. In *International Workshop on Recent Advances in Intrusion Detection*, pages 220–237. Springer, 2003.

[70] David E Mann and Steven M Christey. Towards a Common Enumeration of Vulnerabilities. In *2nd Workshop on Research with Security Vulnerability Databases, Purdue University, West Lafayette, Indiana*, 1999.

[71] Robert A Martin, Steven M Christey, and Joe Jarzombek. The Case for Common Flaw Enumeration. In *Proceedings of Workshop on Software Security Assurance Tools, Techniques, and Metrics*, number 500-265, 2005.

[72] Steven McCanne and Van Jacobson. The BSD Packet Filter: A New Architecture for User-level Packet Capture. In *USENIX winter*, volume 46, 1993.

[73] Nour Moustafa. A New Distributed Architecture for Evaluating AI-based Security Systems at the Edge: Network TON_IoT Datasets. *Sustainable Cities and Society*, 72:102994, 2021.

[74] Nour Moustafa and Jill Slay. 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. IEEE, 2015.

[75] Nour Moustafa and Jill Slay. The Evaluation of Network Anomaly Detection Systems: Statistical Analysis of the NB15 Data Set and the Comparison with the KDD99 Data Set. *Information Security Journal: A Global Perspective*, 25(1-3):18–31, 2016.

[76] Nicolas M Müller and Karla Markert. Identifying Mislabeled Instances in Classification Datasets. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[77] Azqa Nadeem, Daniël Vos, Clinton Cao, Luca Pajola, Simon Dieck, Robert Baumgartner, and Sicco Verwer. SoK: Explainable Machine Learning for Computer Security Applications. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 221–240. IEEE, 2023.

[78] Guansong Pang, Chunhua Shen, Huidong Jin, and Anton van den Hengel. Deep Weakly-Supervised Anomaly Detection. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1795–1807, 2023.

[79] Guansong Pang, Chunhua Shen, and Anton van den Hengel. Deep Anomaly Detection with Deviation Networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 353–362, 2019.

[80] CS Penrod and TJ Wagner. Another Look at the Edited Nearest Neighbor Rule. Technical report, TEXAS UNIV AT AUSTIN DEPT OF ELECTRICAL ENGINEERING, 1976.

[81] Jared M Peterson, Joffrey L Leevy, and Taghi M Khoshgoftaar. A Review and Analysis of the Bot-IoT Dataset. In *2021 IEEE International Conference on Service-Oriented System Engineering (SOSE)*, pages 20–27. IEEE, 2021.

[82] Shebuti Rayana. ODDS Library. http://odds.cs.stonybrook.edu, 2016. Accessed: 2023-06-02.

[83] Markus Ring, Sarah Wunderlich, Deniz Scheuring, Dieter Landes, and Andreas Hotho. A Survey of Network-based Intrusion Detection Data Sets. *Computers & Security*, 86:147–167, 2019.

[84] Nithya Sambasivan, Shivani Kapania, Hannah Highfill, Diana Akrong, Praveen Paritosh, and Lora M Aroyo. "Everyone wants to do the model work, not the data work": Data Cascades in High-Stakes AI. In *proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2021.

[85] Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. Towards a Standard Feature Set for Network Intrusion Detection System Datasets. *Mobile networks and applications*, pages 1–14, 2022.

[86] HyungBin Seo and MyungKeun Yoon. Generative Intrusion Detection and Prevention on Data Stream. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 4319–4335, 2023.

[87] Giorgio Severi, Simona Boboila, Alina Oprea, John Holodnak, Kendra Kratkiewicz, and Jason Matterer. Poisoning Network Flow Classifiers. In *Proceedings of the 39th Annual Computer Security Applications Conference*, pages 337–351, 2023.

[88] Iman Sharafaldin, Amirhossein Gharib, Arash Habibi Lashkari, and Ali A Ghorbani. Towards a Reliable Intrusion Detection Benchmark Dataset. *Software Networking*, 2018(1):177–200, 2018.

[89] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. *ICISSp*, 1:108–116, 2018.

[90] Ryan Sheatsley, Blaine Hoak, Eric Pauley, and Patrick McDaniel. The Space of Adversarial Strategies. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 3745–3761, 2023.

[91] Ryan Sheatsley, Nicolas Papernot, Michael Weisman, Gunjan Verma, and Patrick McDaniel. Adversarial Examples in Constrained Domains. *arXiv preprint arXiv:2011.01183*, 2020.

[92] Ryan Sheatsley, Nicolas Papernot, Michael J Weisman, Gunjan Verma, and Patrick McDaniel. Adversarial Examples for Network Intrusion Detection Systems. *Journal of Computer Security*, (Preprint):1–26, 2022.

[93] Tom Shenkar and Lior Wolf. Anomaly Detection for Tabular Data with Internal Contrastive Learning. In *International Conference on Learning Representations*, 2022.

[94] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, and Ali A Ghorbani. Toward Developing a Systematic Approach to Generate Benchmark Datasets for Intrusion Detection. *computers & security*, 31(3):357–374, 2012.

[95] Arumoy Shome, Luis Cruz, and Arie Van Deursen. Data Smells in Public Datasets. In *Proceedings of the 1st International Conference on AI Engineering: Software Engineering for AI*, pages 205–216, 2022.

[96] Ankush Singla, Elisa Bertino, and Dinesh Verma. Preparing Network Intrusion Detection Deep Learning Models with Minimal Data using Adversarial Domain Adaptation. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, pages 127–140, 2020.

[97] Robin Sommer and Vern Paxson. Outside the Closed World: On using Machine Learning for Network Intrusion Detection. In *2010 IEEE symposium on security and privacy*, pages 305–316. IEEE, 2010.

[98] Blake E Strom, Andy Applebaum, Doug P Miller, Kathryn C Nickels, Adam G Pennington, and Cody B Thomas. Mitre ATT&CK: Design and Philosophy. In *Technical report*. The MITRE Corporation, 2018.

[99] Girish Suryanarayana, Ganesh Samarthyam, and Tushar Sharma. Refactoring for Software Design Smells. *ACM SIGSOFT Software Engineering Notes*, 40, 2015.

[100] Wesley Joon-Wie Tann, Jackie Jin Wei Tan, Joanna Purba, and Ee-Chien Chang. Filtering DDoS Attacks from Unlabeled Network Traffic Data Using Online Deep Learning. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 432–446, 2021.

[101] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. 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. Ieee, 2009.

[102] Robert L Thorndike. Who Belongs in the Family. In *Psychometrika*. Citeseer, 1953.

[103] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[104] Antonio Torralba and Alexei A Efros. Unbiased Look at Dataset Bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011.

[105] Andrea Venturi, Matteo Ferrari, Mirco Marchetti, and Michele Colajanni. ARGANIDS: A Novel Network Intrusion Detection System based on adversarially Regularized Graph Autoencoder. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, pages 1540–1548, 2023.

[106] Kai Wang, Zhiliang Wang, Dongqi Han, Wenqi Chen, Jiahai Yang, Xingang Shi, and Xia Yin. BARS: Local Robustness Certification for Deep Learning based Traffic Analysis Systems. In *NDSS*, 2023.

[107] Xian Wang. ENIDrift: A Fast and Adaptive Ensemble System for Network Intrusion Detection under Real-world Drift. In *Proceedings of the 38th Annual Computer Security Applications Conference*, pages 785–798, 2022.

[108] Xiaolei Wang, Lin Yang, Dongyang Li, Linru Ma, Yongzhong He, Junchao Xiao, Jiyuan Liu, and Yuexiang Yang. MADDC: Multi-scale Anomaly Detection, Diagnosis and Correction for Discrete Event Logs. In *Proceedings of the 38th Annual Computer Security Applications Conference*, pages 769–784, 2022.

[109] Dennis L Wilson. Asymptotic Properties of Nearest Neighbor Rules using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics*, (3):408–421, 1972.

[110] Renjie Wu and Eamonn Keogh. Current Time Series Anomaly Detection Benchmarks are Flawed and are Creating the Illusion of Progress. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[111] Yang Wu, Xurui Li, Xuhong Zhang, Yangyang Kang, Changlong Sun, and Xiaozhong Liu. Community-Based Hierarchical Positive-Unlabeled (PU) Model Fusion for Chronic Disease Prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 2747–2756, 2023.

[112] Junchi Xing and Chunming Wu. Detecting Anomalies in Encrypted Traffic via Deep Dictionary Learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 734–739. IEEE, 2020.

[113] Limin Yang, Wenbo Guo, Qingying Hao, Arridhana Ciptadi, Ali Ahmadzadeh, Xinyu Xing, and Gang Wang. CADE: Detecting and Explaining Concept Drift Samples for Security Applications. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2327–2344, 2021.

[114] Lun-Pin Yuan, Peng Liu, and Sencun Zhu. Recompose Event Sequences vs. Predict Next Events: A Novel Anomaly Detection Approach for Discrete Event Logs. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 336–348, 2021.

[115] Zili Zha, An Wang, Yang Guo, Doug Montgomery, and Songqing Chen. BotSifter: an SDN-based Online Bot Detection Framework in Data Centers. In *2019 IEEE Conference on Communications and Network Security (CNS)*, pages 142–150. IEEE, 2019.

[116] J. Zhou. Security Conference Ranking. http://jianying.space/conference-ranking.html, 2023. Accessed: 2024-03-21.

A. Treatment of *CTU-13*

Due to the granular labelling of *CTU-13*, treating each label as a separate class, as we do for the other datasets, is infeasible. Instead, we combine conceptually similar labels based on their protocol and purpose, determined via the provided labels. For instance, for *Neris 2*, we combine all malicious, established TCP connections into a single class, reducing 73 labels — many associated with only one flow — to a single class. The exact combinations chosen are available in the accompanying GitHub repo. In Table 8, we provide our results for the primary attack associated with a scenario, based on volume. We describe the associated malicious behaviours in Table 10.

CTU-13 provides a large amount of *Background* traffic, a subset of which has been filtered and labelled as *Normal* traffic. In the main body, we use this *Normal* traffic for our comparative measures, WL_C and HDF_C. However, in some cases, based on *CTU-13*'s granular labelling, it is apparent that the *Background* traffic could provide a more challenging classification task, impacting these measures. Where applicable, we rerun our comparative tests using the *Background* flows. We present our results in Table 11. Our other measures are not affected by this change.

Class	Attack	Class	Attack
Neris 1	Injected Ad Traffic	Murlo	UDP C&C
Neris 2	Injected Ad Traffic	Neris 3	Injected Ad Traffic
Rbot 1	PortScan	Rbot 3	ICMP Flood
Rbot 2	UDP Flood	Rbot 4	ICMP Flood
Virut 1	SMTP Proxy	NSIS	UDP C&C
Donbot	Attempted TCP Spam	Virut 2	SMTP Proxy

TABLE 10: Scenario and associated malicious behaviour.

Class	WL_C	HDF_C	Class	WL_C	HDF_C
Neris 1	**1.0**	0.73	Murlo	**1.0**	0.5
Neris 2	**0.95**	0.74	Neris 3	**0.89**	0.69
Rbot 1	**0.18**	**1.0**	Rbot 3	**0.95**	**1.0**
Rbot 2	0.03	**1.0**	Rbot 4	0.01	**1.0**
Donbot	0.01	**1.0**	NSIS	**1.0**	0.76

TABLE 11: Comparative measures with *Background* traffic.

B. Paper Overview

For our paper overview selection, we employed the following criteria:

- The paper uses at least one of the 7 datasets that we have covered here. Note that we looked through papers that cited the dataset's official paper, potentially missing incorrect citations.
- The aim of the paper is to detect or classify malicious activity found in one of the 7 datasets, or perform some kind of adversarial attack (e.g. adversarial examples, evasion attacks) against a model trained on one of the 7 datasets. For this reason, we excluded systematisations of knowledge [9], [10], [77] and papers that criticise other aspects of Machine Learning approaches for Network Intrusion Detection [7], [38]
- The paper was published at one of the venues listed in Table 12. As mentioned at the beginning of Section 3, we expanded our initial list of 7 top security target venues to increase dataset coverage.

Acronym	Venue Full Name
USENIX	USENIX Security Symposium
S&P	IEEE Symposium on Security and Privacy
CCS	ACM SIGSAC conference on computer and communications security
NDSS	Network and Distributed System Security Symposium
EuroS&P	IEEE European symposium on security and privacy
ACSAC	Annual Computer Security Applications Conference
AsiaCCS	ACM Asia conference on computer and communications security
SAC	ACM/SIGAPP Symposium on Applied Computing
CNS	IEEE Conference on Communications and Network Security
DIMVA	Detection of Intrusions and Malware, and Vulnerability Assessment
InfoCOM	IEEE International Conference on Computer Communications
WWW	World Wide Web Conference
CIKM	Conference on Information and Knowledge Management
KDD	ACM SIGKDD Conference on Knowledge Discovery and Data Mining

TABLE 12: List of venues considered for our selection of papers that use one of the 7 NIDS datasets treated in this work.

We evaluated dataset(s) usage as follows:

- *Feature Variation (FV)* - We marked the assumption as *present* if the paper does not mention any discussion or analysis of the variability (or lack thereof) of features in a class. This also extends to discussion or analysis on whether the distribution of features within a certain malicious class is, due to a lack of intra-class variation, significantly different from that of benign traffic, rendering the classification task trivial.
- *Attack Variation (AV)* - We marked the assumption as *present* if the paper does not mention any discussion or analysis of the number of distinct interactions in a class, whether the attack was repetitive in nature, or whether the attack was set up in a very simplistic way.
- *Highly Dependent Features (HDF)* - We marked this category as *present* if the paper does not perform any *semantic* post-hoc analysis on the most important features according to their trained model, namely whether these features are realistically able to characterise an attack or whether the feature's value is merely spuriously correlated to a certain attack. In order to mark this category as *not present*, we required that the paper's semantic analysis is built upon expert knowledge, which usually necessitates some level of manual PCAP analysis.
- *Wrong labels or Unclear Ground Truth (W/U)* - We marked the assumption as *present* if papers accepted the dataset's labels without questioning the validity of the ground truth. We marked it as *not present* if the paper performs PCAP analysis to verify the ground truth (e.g. inspecting their model's false positives and false negatives).

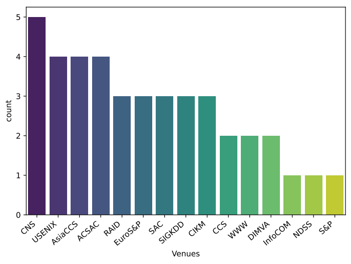

Figure 6: Plot of venue counts in paper overview.

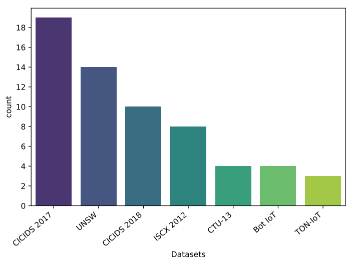

Figure 7: Plot of dataset citation counts in paper overview.