Measuring the Complexity of Benchmark NIDS Datasets via Spectral Analysis

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

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

***Abstract*—Due to the difficulties in obtaining real-world network traffic, lab-based datasets are commonplace in ML-based NIDS research. However, despite their popularity, whether these datasets conform to the expected properties of real-world traffic is unclear. To elucidate this, we present a novel measure of *input complexity* based on spectral clustering analysis. Via this measure, we evaluate two popular benchmark NIDS datasets, contrasting their input complexity with both real-world network traffic and benchmark image datasets. We find that the volumetric attack classes contain minimal data diversity, potentially limiting researchers' abilities to evaluate their methodologies. Following this, we produce our own attack traffic with considerably higher input complexity, demonstrating that, despite their repetitive nature, low data diversity is not a universal property of volumetric attacks. Our work highlights likely limitations of synthetic benchmark datasets for evaluating machine-learning based NIDS as well as the importance of input complexity in network intrusion dataset design.**

1. Introduction

Despite the ubiquity of synthetic datasets in machine learning-based network intrusion research, their suitability as benchmarks relative to real-world data is unknown. To our knowledge, no work establishes a general baseline for the classification complexity of network anomalies that could be used to evaluate lab-based datasets. Due to their synthetic nature, the *input complexity* of these benchmarks is vitally important to understand their relationship with real-world traffic. Using a novel metric, we aim to bridge this gap in knowledge, demonstrating that these datasets often have minimal data diversity that fails to represent the breadth of even repetitive, volume-based attacks.

Motivation: Lab-based datasets are entrenched in network intrusion research. Machine learning research investigating the detection of network anomalies relies heavily on datasets such as CIC IDS 2017 and CSE-CIC IDS 2018 [17], which have thousands of citations. These datasets are made by scripting interactions across a laboratory environment, simulating the internal network of an organisation. Often, the exact traffic, protocols and interactions within these datasets are poorly documented, raising significant issues regarding the complexity of the data: primarily, do these scripted processes accurately reflect the variability of real-world traffic? Although carefully generated synthetic data may produce sufficiently complex traffic, recent criticism of these particular datasets suggests that they might not reach this standard [6], [12]. As a result, these may be considerably simplified compared to a real-world network, potentially leading to experimental bias or limiting researchers' abilities to reason about their methodologies. Given this issue, a researcher developing, say, an anomaly detection system, may want to know which dataset most complex, relying on reasonable heuristics such as number of hosts, the size of the dataset or some statistical analysis, such as feature correlations. However, there is no guarantee that these heuristics exhaustively capture data complexity. Thus, machine learning NIDS techniques may depend on simulation artefacts specific to these datasets, failing to generalise to real-world data.

The realism of a network intrusion dataset is difficult to quantify. The paucity of data from equivalent, real-world networks prevents easy comparison between synthetic and real-world data. Furthermore, there is likely not a single fixed notion of what a 'realistic' network intrusion dataset looks like, with numerous dissimilar datasets being admissible, depending on the network topology, services, reliability etc. That said, even given the difficulty of defining 'realistic' data, it is unlikely that some lab-based datasets are valid representatives of real-world networks. For instance, in CIC IDS 2017, FTP traffic consists of downloading a `.txt` file of the *Encryption* Wikipedia page several thousand times [5]. Whilst these connections might be individually realistic, the lack of data *diversity* does not capture the breadth of the FTP protocol, resulting in data with high inter-flow and inter-feature dependencies. Noting this, we aim to evaluate synthetic NIDS datasets by measuring *input complexity*, an expected aspect of real-world data.

Whilst this shift moves us closer to evaluating NIDS datasets empirically, measuring input complexity is still difficult. Whilst numerous measures exist for *classification complexity* between classes, selecting an objective complexity metric is challenging as the Kolmogorov complexity [10] of a dataset, a reasonable starting point, is uncomputable. Compression-based metrics [16] have been used to approximate input complexity, but these fail to capture inter-feature dependencies. Moreover, NIDS datasets are increasingly being used outside of security contexts as general anomaly or classification datasets. Thus, it would be highly beneficial to compare input complexity of NIDS datasets with those of different fields, for instance, benchmark image datasets, which, again, can not be captured by compression-based metrics.

By measuring inter-feature relationships directly, via,

say, feature correlation, we can gain some intuition about the input complexity of a NIDS dataset. However, this approach is limited. For an 80 feature dataset, such as CIC IDS 2017, this requires 6320 comparisons, which is onerous and difficult to contrast across datasets. However, by comparing features on a pairwise basis like this, we have implicitly defined a weighted graph, where the weight of each edge depends on the relationship between features, reflecting the dataset's input complexity. Spectral clustering theory is a natural choice for analysing these complexity graphs, and has previously been used to define classification complexity metrics [2].

Idea: We define an input complexity metric, the *Spectral Input Complexity* or SIC, based on this insight. By embedding datasets into a latent space of fixed size, d, and defining a sensible measure of inter-feature dependency, we can build a weighted graph where weakly dependent features have a higher weighting and strongly dependent features have a lower weighting. We can then compute a $d \times d$ Laplacian matrix and calculate its spectrum, $\{\lambda_0, ..., \lambda_{d-1}\}$. These λ_i will be lower for 'simpler' datasets and higher for 'complex' datasets and, as each λ_i is bounded according to d, we can encapsulate input complexity as a single, normalised number. This measure directly captures feature dependencies and can be used to compare datasets across different fields. Our contributions include:

- **Novelty:** We present a novel metric, SIC, based on spectral analysis to measure input complexity of network intrusion datasets for machine learning. This metric directly captures inter-feature dependencies and can order samples by their contribution to overall data complexity.
- **Comparison:** We compare the input complexity of popular, benchmark NIDS datasets to real world traffic as well as benchmark datasets for other tasks. Based on SIC, we find these NIDS datasets to be considerably less complex than datasets that are understood to be simple, such as MNIST.
- **Dataset Selection:** SIC demonstrates that, despite the recency and expanded size of CSE-CIC IDS 2018, the input complexity of its background traffic is lower than that of its predecessor, CIC IDS 2017, suggesting, for some purposes it is a simpler benchmark dataset.
- **Improvement:** Due to the poor input complexity of volumetric attacks in lab-based datasets, we generate attack traffic to maximise input diversity, demonstrating that this is not a flaw of volumetric attack data generally. We use SIC to quantify this increase in complexity and demonstrate the impact this has on classification difficulty when injected into real-world data.

2. Background

2.1. Complexity Measures

Existing work on complexity in machine learning primarily focuses on the classification complexity of data, defined as the Kolmogorov complexity of the classification boundary, i.e., the minimal length of a computer program

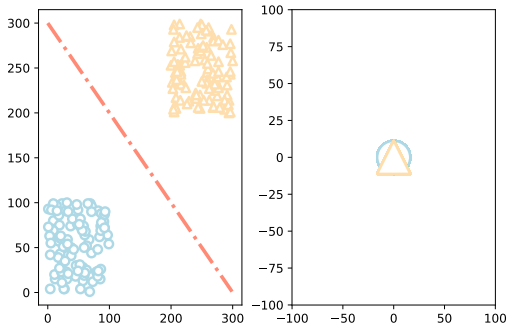

Figure 1: A toy example demonstrating the difference between input complexity and classification complexity. On the left, the two classes have high input complexity, but trivial classification complexity whilst the opposite is true on the right.

that describes the boundary, with more complex boundaries requiring longer programs. However, Kolmogorov complexity is uncomputable. Thus, practical measures of classification complexity are instead based on geometrical estimates of the decision boundary. Twelve of these geometrical descriptors are outlined by Ho and Basu [8], who characterise measures belonging to three categories: feature overlap, class separability and geometry, topology & density of manifolds. Measures of class separability have been successful: Branchaud et al. [2] measure classification complexity via spectral analysis of a graph with weights representing the overlap between classes of a dataset. In this work, we follow this general approach of forming a sensible complexity measure across multiple nodes, condensing this to a single number via spectral clustering.

However, classification complexity is distinct from the complexity of the data within a dataset, such as the visual complexity of a images or the inter-feature correlations of tabular data, or the *input complexity* of the dataset, which we focus on. Figure 1 demonstrates the difference for a toy example. Lloyd [13] present a large list of categories of complexity measures, including entropy-based, dimension-based, correlation-based and grammatical-based, reflecting the difficulty of defining complexity for a specific problem. Furthermore, certain methods, including entropy-based approaches, are difficult to calculate for high-dimensional data. Instead, investigating the relationship between input complexity and out-of-distribution detection, Serrà et al. [16] show that compression-based complexity measures are good indicators of input complexity for benchmark machine learning datasets.

2.2. NIDS Datasets

Several well-cited, benchmark NIDS datasets do not contain data collected from the real-world. Instead, datasets are sometimes *lab-based*, consisting of data generated via scripted procedures on virtual machines. We do not claim that synthetic network datasets *must* have lower input complexity than real-world datasets. However, these scripted interactions could produce 'simple' data if they are not sufficiently heterogeneous, which we investigate.

In this work, we focus on CIC IDS 2017 and CSE-CIC IDS 2018 — collectively the *CIC* datasets [17]. Both datasets were constructed in a similar manner; the main difference is size. CSE-CIC IDS 2018 contains traffic from over 500 hosts whilst CIC IDS 2017 contains approximately 10. We use the datasets' accompanying features, generated using *CICFlowMeter* [1], a set of 80 flow statistics including measures such as the *Total Backwards Flow Length*. Attack classes are largely *volumetric* in nature, such as denial of service or password bruteforce attacks. As both of these datasets have been known to have flaws, we use a partially amended version of these statistics released by Liu and Engelen et al. [6], [12]

2.3. Spectral Analysis

Consider an undirected, weighted similarity graph $G = (V, E)$. We denote the weight of edge E_{ij} as $w_{ij} \geq 0$. The weight of an edge relates to the 'closeness' of two adjacent nodes with closer nodes having a higher weights. A weight of 0 suggests no relationship between nodes. We summarise G via an adjacency matrix A where $A_{ij} = w_{ij}$. Note that A is a symmetric, square matrix of size n, where n is the number of nodes.

Spectral clustering provides us with the machinery to divide G into a series of subgraphs of minimal weight by 'cutting' the graph along its edges and incurring a greater 'cost' when cutting highly weighted edges. To do this, we first calculate the Laplacian as $L = D - A$ where D is the degree matrix with $D_i = \sum_j w_{ij}$. As L is symmetric and positive semi-definite, we can calculate n eigenvalues $\{\lambda_0, ..., \lambda_{n-1}\}$. All λ_i are real and greater than 0, except for λ_0, which equals 0 for a well-formed graph. These eigenvalues are known as the *spectrum* of L, and they provide inside into the cost associated the cuts necessary to form each subgraph. These eigenvalues are ordered by increasing size, and the size of a given λ_i is proportional to the cost of subdividing G into i subgraphs.

3. Methodology

3.1. Our Complexity Measure, SIC

Let x be an input sample and $\phi(x) \in \mathbb{R}^d$ be some embedding of that sample. To measure input complexity via spectral analysis, we first need to produce a graph where more complex relationships between nodes result in edges with higher weights. For our metric, we produce a weighted graph using the inter-feature conditional expectation of $\phi(x)$, i.e., $E(F_j|F_i)$, where $\phi(x) = \bigoplus_{n=0}^{d} F_n$.

Intuitively, given data with high input complexity, it is difficult to extract meaningful features from the data, and conditioning the expected value of $E(F_j)$ on F_i provides little information. For data with low input complexity, the opposite is true. $E(F_j|F_i)$ will yield significant improvement over $E(F_j)$. Our metric relies on the quality of $E(F_j|F_i)$ to measure complexity.

Once we have a low-dimensional embedding $\phi(x) \in \mathbb{R}^d$, $E(F_j|F_i)$ can be estimated quickly, either in a symmetric[1] manner using, say, linear regression or a non-symmetric manner using, say, a simple random forest trained and tested on a small subset of the embedding. We measure the goodness of each test fit by calculating R^2, the coefficient of determination [2]. Doing so leads to a $d \times d$ matrix $\hat{S}$. Next, $\hat{S}$ can be converted to a similarity matrix S by subtracting from the matrix of all ones: $S = 1 - \hat{S}$. Because S is symmetric, up to small deviations, it is thus an adjacency matrix. If we use a non-symmetric method to calculate $E(F_j|F_i)$ we have two weights for each edge in $\hat{S}$. We prioritise the smaller weight, calculating S as $S = 1 - min(\hat{S}, \hat{S}^\intercal)$.

Intuitively, a class with little variation and high correlation between features will produce a highly connected graph. For this graph, any cut will be expensive. We expect higher eigenvalues for graphs representing complex data structures and vice versa for simpler data structures. Thus, we measure input complexity via the normalised sum of eigenvalues:

$$SIC = \sum_i \frac{\lambda_i}{d(d-1)} \in [0, 1] \tag{1}$$

Effectively, spectral complexity analysis allows us to reduce unwieldy feature relationships to a single, normalised measure. Although dataset agnostic, SIC is particularly applicable to lab-based NIDS datasets due to two common weaknesses: low data diversity and high inter-feature dependency — with flow statistics often containing highly correlated measures (such as 'Total Forwards Packet Length', 'Total Backwards Packet Length' and 'Total Packet Length'). As each entry contributes individually to SIC, we can order entries by their predictability and infer such correlations or patterns. We apply SIC to tabular, numerical datasets as the pre-calculated flow statistics of NIDS datasets are commonly used off-the-shelf in this format.

Whilst our approach is limited by only considering the relationship between pairs of features, rather than, say, arbitrary numbers of features, we show that this is a good approximation of input complexity in Section 4.1. Furthermore, measuring input complexity in this way can provide additional insight into the structure of the data via closer examination of the eigenvalues and eigenvectors, which we hope to explore in future work.

3.2. Finding a Good Embedding ϕ

For our embedding $\phi(x)$, we consider the latent vector of a fully-connected autoencoder trained using a modified loss function detailed in Section 3.2.1.

3.2.1. Loss Function. $\phi(x)$ produces two outputs: our latent vector z and y such that $y \approx x$. We could train our autoencoder just using a Mean Squared Error loss:

$$\sum_n^i (y_i - x_i)^2 \tag{2}$$

However, relying exclusively on Eq. 2 gives us few guarantees about the structure of the embedding and, as a

1. $E(F_j|F_i) = E(F_i|F_j)$

2. As extremely inaccurate fits cause R^2 to be negative, we bound it from below by 0.

Dataset/Class	0/A	1/B	2/C	3/D	4/E	5/F	6/G	7/H	8/I	9/J
MNIST	0.559	0.229	0.546	0.483	0.476	0.464	0.532	0.393	0.586	0.432
notMNIST	0.658	0.721	0.65	0.646	0.604	0.646	0.679	0.613	0.508	0.542
	Port 80	Port 443	FTP-BF	SSH-BF	DoS Hulk	DoS-GE	DoS-SL	nmap	DDoS	DoS-SHT
CIC IDS 2017	0.143	0.284	**0.031**	**0.02**	**0.032**	**0.000**	**0.022**	**0.000**	**0.000**	**0.043**
CSECIC-IDS 2018	**0.026**	0.265	**0.000**	**0.000**	**0.039**	**0.066**	**0.001**	**0.000**	**0.000**	-
Port No.	80	443 (TCP)	443 (UDP)	22	25/537	53	30303	33**	-	-
MAWI	0.195	0.21	0.257	0.111	0.142	**0.072**	0.133	**0.0249**	-	-

TABLE 1: SIC calculated across several benchmark datasets. **NB**: A SIC score of zero does not imply that all entries are identical. However, it does indicate that all entries can be separated into a small number of tight, non-overlapping clusters. Port 30303 is Ethereum network traffic. Ports 3300-3399 appear to be P2P torrenting traffic.

result, SIC can be quite volatile. To remedy this, we train ϕ with an additional *cross-covariance loss* term that aims to maximise the covariance between latent features, encouraging smaller values of SIC.

Given z of dimension $batch\_size \times d$, we calculate the cross covariance matrix of z as $\bar{z}^\intercal \bar{z}$ where $\bar{z} = \frac{z-\mu_z}{\sigma_z}$. This cross-covariance matrix has diagonals of value $(batch\_size - 1)$, and each off-diagonal element is bounded by $[-(batch\_size-1), (batch\_size-1)]$. We normalise $\bar{z}^\intercal \bar{z}$ by $(batch\_size-1)$, and calculate the MSE between it and a matrix that maximises (or minimises) the covariance of $\bar{z}$, i.e., $sign(\bar{z}^\intercal \bar{z})$. Our additional loss term is then:

$$\epsilon \sum_{n}^{i} \Big(\frac{\bar{z}^\intercal \bar{z}}{batch\_size - 1} - sign(\bar{z}^\intercal \bar{z}) \Big)^2 \tag{3}$$

where ϵ is a hyperparameter controlling the influence of this term. Our full loss function is:

$$\sum_{n}^{i} (y_i - x_i)^2 + \epsilon \sum_{n}^{i} \Big(\frac{\bar{z}^\intercal \bar{z}}{batch\_size - 1} - sign(\bar{z}^\intercal \bar{z}) \Big)^2 \tag{4}$$

4. Experiments & Results

For all of our experiments, we use a fully connected autoencoder with three layers of size $(512, 256, 128)$ with $ReLU$ activations and a latent space of size $d = 16$ as ϕ. As overly large values of ϵ cause the model to learn meaningless embeddings, distorting SIC, we choose ϵ for each dataset such that Eq. 3 is bounded by approximately $0.01 * $ Eq. 2, as we found this ratio to produce consistent values of SIC whilst not impacting the mean squared error between x and $\phi(x)$ when trained for 400 epochs.

4.1. Relationship with Complexity

Although we aim to measure input complexity, the relationship between our metric and input complexity is not immediate. To demonstrate that SIC accurately reflects input complexity, we perform two simple comparisons.

As an initial common sense check, we create a toy dataset with 128 features, all perfectly linear dependent on one another. Using this as a base, we produce 9 additional datasets, $D_i,\ i \in [1, 9]$. For a given D_i, we perturb all features with random, normal noise $\sim N(0, i * m)$, where m is a multiplicative factor. Thus, higher values of i result in weaker correlations between features and $i < j$ should imply that $SIC(D_i) < SIC(D_j)$.

Secondly, we demonstrate how SIC behaves on a standard benchmark dataset: MNIST [11]. As an approximation to measure the complexity of each MNIST class, we modify the approach of Serrà et al [16]: given a set of inputs x, we calculate the mean Normalised Compression Distance [4] (NCD) between randomly sampled pairs of input, i.e., $NCD(x_1, x_2)$, using `lzma` as our compression algorithm. By using a compression-based distance metric, we encapsulate both the average image complexity as well as similarities in complexity between images. However, unlike SIC, this NCD procedure does not capture inter-feature dependencies between images. We then order the MNIST digits by input complexity by calculating the mean score produced by this NCD procedure, which we compare with *SIC* for each digit.

4.1.1. Results. For both tests, we find that SIC accurately reflects the input complexity of the data, measured by our NCD procedure. On our toy dataset, SIC monotonically increases as the correlations between features become weaker, as in Figure 2. On MNIST, we find a Spearman correlation between SIC and NCD of 0.952, indicating an extremely high correlation. With the exception of the digit *4*, the complexity ordering of SIC is identical to that of NCD, as demonstrated in Figure 3.

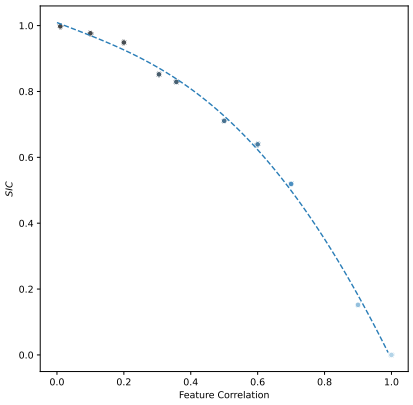

Figure 2: Inter-feature correlation vs. SIC on our toy data.

4.2. Benchmark Comparisons

Although recent criticism of the CIC datasets [9], [12] may raise questions about their input complexity, it is

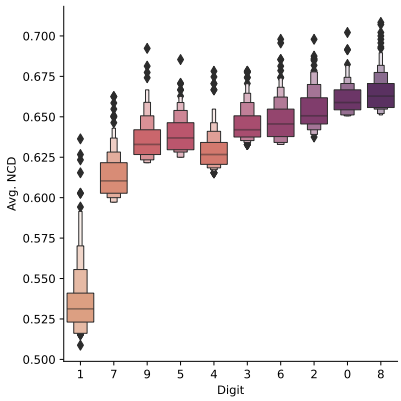

Figure 3: Box plot of the NCD metric. Digits are ordered according to their SIC score. With the exception of digit *4*, NCD increases monotonically as SIC increases.

unclear how simple they are relative to other benchmark datasets. SIC allows us to measure the input complexity of many datasets in a directly comparable manner.

Placing the input complexity of synthetic NIDS datasets in context requires real-world traffic from an internal network. However, this data is extremely difficult to obtain. Instead, we provide two juxtapositions: common benchmark image datasets and backbone network traces. First, we use two benchmark image classification datasets: alongside MNIST, we use notMNIST, a dataset containing 28×28px images of the letters A through J in a variety of typefaces. Second, we calculate SIC on network traces from the MAWI Traffic Data Repository [7] between 2017 and 2023. This dataset repository contains daily network traces of backbone internet traffic between the USA and Japan. Whilst this is not directly comparable to the traffic of an internal network, it does provide some indication into the breadth of data diversity possible for a network dataset. As CICFlowMeter unfortunately crashes when run on MAWI datasets due to their size, we use our own tooling to extract 41 flow statistics. For all datasets, we calculate SIC on a class-by-class basis. We limit our investigation to 10 classes with over 1000 samples in each CIC dataset, including benign traffic directed towards ports 80 and 443. As the MAWI dataset does not have classes, we categorise flows based on destination port.

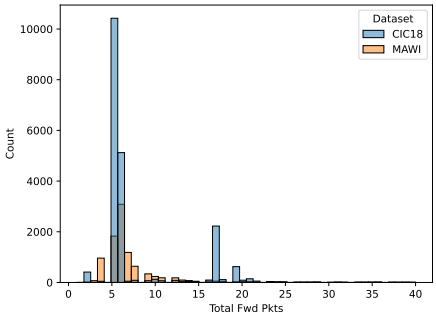

Figure 4: Histogram of Total Fwd Packet feature for CIC IDS 18 vs. MAWI Lab data.

4.2.1. Results. We report our results in Table 1. Some values stand out, such as the extremely low input complexity of the benign HTTP data in CSE-CIC IDS 2018. Investigating samples that contribute minimally to SIC, we see that, despite benign data having generally higher complexity, CSE-CIC IDS 2018 disproportionately contains flows with 5 forward packets and 5/6 backward packets, as seen in Figure 4. The statistics of such flows are highly homogeneous, resulting in low input complexity.

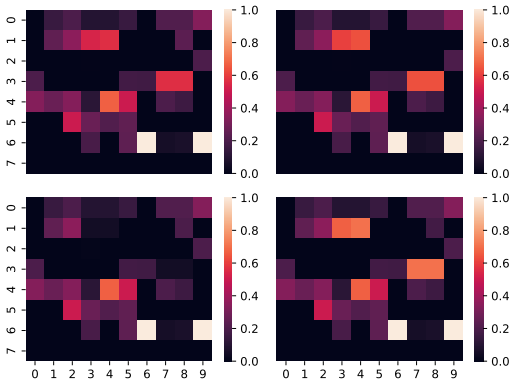

Figure 5: Heatmap of normalised features of four random, distinct CSE-CIC IDS 2018 FTP-Bruteforce samples.

Worryingly, according to SIC, all analysed attack classes in CIC IDS 2017 and CSE-CIC IDS 2018 have minimal complexity stemming from trivial data diversity, an example of which can be seen in Figure 5, providing numerical support for Engelen et al.'s claim that launching attacks using fixed tooling leads to poor data quality [6]. In contrast, the minority classes of our MAWI data score higher, with the exception of DNS and torrenting traffic. Furthermore, the complexity of network features are considerably lower than that of image datasets, with very few exceptions. Whilst the homogeneity of lab-based data contributes to this, redundancy between features is likely also a factor.

4.3. Improving NIDS Dataset Complexity

Whilst Table 1 suggests that the input diversity of attack traffic in the CIC datasets is low, this may be a natural consequence of volumetric attacks. After all, these attacks are intentionally repetitive, and it may be incorrect to say they could have more diverse feature distributions.

To test this, we attempt to rectify this poor diversity by generating our own volumetric attack traffic, using SIC to demonstrate the improvement in input complexity. We utilise the synthetic data generation tool DetGen [5], allowing us to simulate traffic with specific network conditions in a controlled and fine-grained manner. Like the CIC datasets, we collect denial of service traffic by repeatedly visiting webpages using malicious tooling. However, we aim to maximise both spacial and temporal variation in our generated traffic. We increase spacial diversity by attacking randomly generated webpages between 0MB and 12MB in size. Similarly, we increase temporal diversity by artificially limiting host bandwidth to five rates: 50Mb, 25Mb, 10Mb, 5Mb and 1Mb. Overall, we capture 125 combinations of bandwidth limit and webpage size. As small webpages combined with high bandwidth limits produce highly imbalanced data, we downsample

Dataset	F1 Score
CIC IDS 2017 (Hulk)	1.0
MAWI/DetGen	0.681

TABLE 2: F1 score of a simple DoS classifier trained on Packet Size Features, CIC IDS 2017 vs. MAWI Labs data combined with our data.

flows generated under these conditions. For a like-with-like comparison with the CIC datasets, we calculate flow statistics using CICFlowMeter.

4.3.1. Results. Following the above procedure, we generate valid DoS data with considerably greater SIC complexity than its CIC equivalents: **0.141**, achieving near-parity with traffic from the Mawi Labs Repository and demonstrating that volumetric attack data is not necessarily devoid of input complexity.

Importantly, generating traffic whilst deliberately increasing input complexity results in challenging test sets that contain flows dissimilar to the training data. In other words, by increasing input complexity, we increase classification complexity in parallel. This likely better reflects the real-world challenges of intrusion detection, such as identifying out-of-distribution attacks, and provides non-trivial benchmarks. We quickly show the impact of this in Table 2 by injecting our DoS traffic into the MAWI Labs data and training a simple random forest-based IDS, contrasted with CIC IDS 2017.

5. Related Work

As well as general techniques such as KL-divergence or information entropy, several works investigate the input or classification complexity of machine learning datasets. As mentioned previously, Branchaud et al. [2] use a similar spectral clustering-based metric to measure the classification complexity of various image datasets. In their analysis of the relationship between input and classification complexity and model capacity, Mei et al. [14] propose an input complexity measure based on the relative entropy of randomly sampled subsets of a dataset. However, like other measures, this fails to directly capture inter-feature relationships. Recently, Schmidt et al. [15] and Carmon et al. [3] have investigated the relationship between sample complexity and adversarial robustness.

The evaluated datasets have been criticised before in literature reviews and dataset overviews [9], [18]. However, these critiques tend to be high-level with little accompanying evidence. More specific critiques have been raised by Liu and Engelen et al. [6], [12]. Whilst not the focus, their results point towards the low classification complexity of the CIC datasets, with simple random forests achieving extremely high F1 scores.

6. Conclusion

We present an analysis of the input complexity of two highly-cited benchmark intrusion detection datasets via a novel metric, SIC, experimentally justifying its efficacy. Using this measure, we contrast the input complexity of benchmark NIDS datasets with benchmarks from image classification as well as internet backbone traffic. We consistently found these NIDS datasets to be generatively trivial compared to other benchmarks, particularly the volumetric attack traffic which makes up the majority of the malicious data. Following this, we briefly investigate whether this simplicity is an inherent property of volumetric attacks. Using DetGen, we generate attack traffic with higher complexity, pointing towards better methods for generating synthetic attack data. Altogether, our results have potentially significant consequences for the utility of the examined datasets as benchmarks, the realism of their traffic and their suitability for developing machine leaning-based intrusion detection methods.

The measure has some limitations. It is sensitive to hyperparameter selection and comparisons between datasets are easiest when ϕ is fixed. Large outliers in the data must be dropped or normalised carefully as they can impact SIC disproportionately when scaled. For future work, we would like to utilise additional information related to SIC, such as the eigengaps or the eigenvectors, as a similarity measure to compare disparate network statistics. We would also like to investigate other measures of feature relations to determine graph weightings, including entropy-based approaches, and apply the measure to other domains.

Acknowledgements

We gratefully acknowledge support of the UKRI under grant EP/T027037/1 and ARM Limited. This work has benefited from feedback from Gudmund Grov and the WTMC reviewers.

References

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

[2] Frederic Branchaud-Charron, Andrew Achkar, and Pierre-Marc Jodoin. Spectral metric for dataset complexity assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3215–3224, 2019.

[3] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. *Advances in neural information processing systems*, 32, 2019.

[4] Rudi Cilibrasi and Paul MB Vitányi. Clustering by compression. *IEEE Transactions on Information theory*, 51(4):1523–1545, 2005.

[5] Henry Clausen, Robert Flood, and David Aspinall. Traffic generation using containerization for machine learning. In *Proceedings of the 2019 Workshop on DYnamic and Novel Advances in Machine Learning and Intelligent Cyber Security*, pages 1–12, 2019.

[6] 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.

[7] MAWI Working Group et al. Traffic archive. *http://tracer. csl. sony. co. jp/mawi/*.

[8] 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.

[9] 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.

[10] Andrei N Kolmogorov. On tables of random numbers. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 369–376, 1963.

[11] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[12] 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.

[13] Seth Lloyd. Measures of complexity: a nonexhaustive list. *IEEE Control Systems Magazine*, 21(4):7–8, 2001.

[14] Shibin Mei, Chenglong Zhao, Shengchao Yuan, and Bingbing Ni. Towards bridging sample complexity and model capacity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1972–1980, 2022.

[15] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. *Advances in neural information processing systems*, 31, 2018.

[16] Joan Serrà, David Álvarez, Vicenç Gómez, Olga Slizovskaia, José F Núñez, and Jordi Luque. Input complexity and out-of-distribution detection with likelihood-based generative models. *arXiv preprint arXiv:1909.11480*, 2019.

[17] 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.

[18] Ankit Thakkar and Ritika Lohiya. A review of the advancement in intrusion detection datasets. *Procedia Computer Science*, 167:636–645, 2020.