Local Intrinsic Dimensionality of IoT Networks for Unsupervised Intrusion Detection

Matt Gorbett(✉), Hossein Shirazi, and Indrakshi Ray

Colorado State University, Fort Collins, CO 80523, USA
{Matt.Gorbett,Hossein.Shirazi,Indrakshi.Ray}@colostate.edu

Abstract. The Internet of Things (IoT) is revolutionizing society by connecting people, devices, and environments seamlessly and providing enhanced user experience and functionalities. Security and privacy issues remain mostly ignored. Attackers can compromise devices, inject spurious packets into an IoT network, and cause severe damage. Machine learning-based Network Intrusion Detection Systems (NIDS) are often designed to detect such attacks. Most algorithms use labeled data for training the classifiers, which is difficult to obtain in a real-world setting.

In this work, we propose a novel unsupervised machine learning approach that uses properties of the IoT dataset for anomaly detection. Specifically, we propose the use of Local Intrinsic Dimensionality (LID), a theoretical complexity measurement that assesses the local manifold surrounding a point. We use LID to evaluate three modern IoT network datasets empirically, showing that for network data generated using IoT methodologies, the LID estimates of benign network packets fit into low LID estimations. Further, we find that malicious examples exhibit higher LID estimates. We use this finding to propose a new unsupervised anomaly detection algorithm, the *Weighted Hamming Distance LID Estimator*, which incorporates an entropy weighted Hamming distance into the LID Maximum Likelihood Estimator algorithm. We show that our proposed approach performs better on IoT network datasets than the Autoencoder, KNN, and Isolation Forests. We test the algorithm on ToN IoT, NetFlow Bot-IoT (NF Bot-IoT), and Aposemat IoT-23 (IoT-23) datasets, using leave-one-out validation to compare results.

Keywords: Intrusion detection systems · IoT · Anomaly detection

1 Introduction

The proliferation of IoT systems has introduced new, and emerging security vulnerabilities [2,5,13,49] which can be readily exploited to cause harm.

This work was supported in part by funding from NSF under Award Number CNS 1822118, NIST, ARL, Statnett, AMI, Cyber Risk Research, NewPush, and State of Colorado Cybersecurity Center.

Published by Springer Nature Switzerland AG 2022
S. Sural and H. Lu (Eds.): DBSec 2022, LNCS 13383, pp. 143–161, 2022.
https://doi.org/10.1007/978-3-031-10684-2_9

Network Intrusion Detection Systems (NIDS) play a critical role in security of IoT networks [14]. Modern-day NIDS' depend in part on machine learning algorithms' ability to detect malicious actors. However, heterogeneity and non-standardized protocol of IoT networks have been posited as critical challenges for enhancing the security of IoT systems [11,23,37] – networks with diverse devices ranging from single-purpose machines to robust servers, each with varied communication structures, are cumbersome to protect.

1.1 Limitations of Prior Attempts

Various machine learning solutions have been proposed for NIDS in IoT networks. Such solutions incorporate deep learning such as autoencoders and classifiers [10,38,44,45], as well traditional machine learning algorithms [32]. While these solutions are each impactful in their own right, most are supervised learning solutions requiring a fully annotated dataset to train – a costly, time-intensive task. Moreover, with constantly evolving attack vectors (malicious actors acting in novel ways), supervised NIDS solutions trained on datasets with particular attack types become vulnerable [33,47]. Further, IoT networks are fundamentally different from standard networks – devices will be added to the network, and existing devices may have software/firmware updates with greater frequency. Supervised algorithms are unable to detect new devices or updates in these situations without expensive retraining or reconfiguration [9].

To mitigate the problems of supervised learning algorithms in IoT, Haefner and Ray [19] take the novel approach to intrusion detection in IoT from the perspective of device traffic complexity. The authors measure the complexity of network traffic on a per device basis to tune an (unsupervised) Isolation Forest algorithm. They find that several single-purpose IoT devices contain simple (non-malicious) network traffic, enabling us to assume trust of the device based on its low network packet variability. However, the tuning procedure used for the Isolation Forest assumes a particular contamination rate: for more complex devices, they assume a more significant and fixed percentage of network packets are anomalies, an assumption that could lead to false positives and not perform well in real-world scenarios.

1.2 Proposed Approach

We take a different approach by using complexity measurements to detect malicious data. Gorbett et al. [17] showed that heterogeneous IoT networks have a lower complexity than regular non-IoT datasets using Intrinsic Dimensionality (ID). ID is a property that has been proposed to measure the complexity of a data set as a whole [50]. Gorbett et al. [17] measured IoT datasets at the network level and device level. We instead show that at the sample-level (network packets), IoT traffic can be categorized as benign or malicious using a device-independent unsupervised model. We do this using LID, which estimates the intrinsic dimension around an individual data point, and show that malicious activity exhibits higher LID values than benign samples.

1.3 Problem Statement

We focus on the question of heterogeneity and complexity in IoT networks and their effects on detecting malicious activity via NIDS. Specifically, we ask the following questions:

1. Do the properties of multi-device heterogeneous IoT networks exhibit fundamentally more complex behavior?
2. Can we detect malicious activity at the IoT network level in an unsupervised manner, without the need to label each device and attack?
3. Which attacks are networks more susceptible to?

In this work, we measure the complexity of IoT network traffic using the novel perspective of LID. Using this metric, we focus on the problem of detecting malicious actors in IoT networks. We measure the complexity of network packets by formulating an entropy-weighted Hamming distance calculation on top of the LID measurement to construct a novel anomaly detection algorithm. The results of the algorithm show that benign network data in IoT datasets exhibit a lower LID measurement compared to malicious actors, which provides us the opportunity to threshold this measurement during test time. The unsupervised algorithm uses benign IoT network data as a training set and assumes any test sample under threshold τ is benign network behavior. If the LID estimate is above this threshold, we can flag the example as malicious.

Contributions

- We propose a novel algorithm for unsupervised anomaly detection using a combination of Hamming distance and the Hill Maximum Likelihood Estimator (MLE) LID, and show that the algorithm performs competitively with several state-of-the-art algorithms.
- To the best of our knowledge, this is the first work that uses LID to measure IoT datasets. We show the potential utility of this measurement in security by outlining the theoretical basis for the approach.
- We measure the complexity of IoT traffic at the network level, and show that, even for networks with several devices, benign data generally fits into low LID estimates. In contrast, malicious data exhibits higher LID values.

2 Related Work

2.1 Intrusion Detection Systems

Intrusion Detection Systems (IDS) can broadly be classified based on 1) where the detection is placed (network or host) and 2) the detection method that is employed (Machine Learning (ML) anomaly-detection algorithms or traditional signature-based detection where attack patterns are defined in a database). In this work, we concentrate on network-based intrusion detection where anomalies are classified based on an ML algorithm. ML based NIDS are employed in

production into a key point within a network to monitor traffic to and from all devices connected to the network. Network features are extracted from the entire subnet about the passing traffic and scored by the ML detection algorithm. NIDS can operate on-line (real-time detection) or off-line (batch detection). Real-time detection offers more robust security and beneficial results as long as it does not impair the overall speed of the network. Our study aims to characterize network data in real-time.

2.2 IoT

IoT is a rapidly evolving field, with research being done at dozens of institutions across industry and academia [11,23,25,37]. It is postulated that IoT increases the vulnerability of networks because the *attack surface* has increased, with many new entry and exit points with new devices available on networks [30,39]. A heterogeneous IoT network is typically made of various sub-devices within a distributed network. It includes resource-constrained devices, such as a smart light bulb or garage door opener, and more powerful devices such as embedded and regular computers.

Existing research notes that IoT networks and devices have multiple intrusion sources: IoT backends, cloud services supporting an IoT device, and other hubs within the IoT system [1,35], which makes it difficult to implement traditional intrusion detection approaches such as rule-based and signature-based methods.

2.3 State-of-the-Art Network Intrusion Models

Several works propose new and existing algorithms for intrusion detection on commonly used datasets. Nour [32] released dataset *TON_IoT* (*TON_IoT*), as a baseline result that uses many supervised learning algorithms. Sahu et al. [41] proposed a hybrid deep learning model which uses a CNN/LSTM framework to achieve 96% accuracy on the IoT-23 dataset generated by [15] and outperformed several proposed deep learning-based attack detection. Kozik et al. [26] use hybrid time window embeddings with a transformer neural network to classify IoT-23 data. This model achieves between 93% and 95% accuracy on attacks in IoT-23 and does better than three other proposed deep learning models: HaddadPajouh et al. [18] use an LSTM trained on IoT devices execution operation codes (OpCodes), Roy and Cheung [40] use a bi-directional LSTM for detecting attacks on UNSW-NB15, and Azmoodeh et al. [7] use OpCodes to train a deep Eigenspace model to detect attacks. Moustafa and Slay released UNSW-NB15 dataset [33] that was generated using IXIA PerfectStorm. The dataset has been widely used as a benchmark for comparison. MStream [10] is an online neural network-based anomaly detection algorithm. It uses multi-aspect streams, such as multiple features, continuous, categorical. This tool achieves 0.90 AUROC on UNSW-NB15 and is considered state-of-the-art according to PapersWithCode.com[1]. The Edge-detect model [45] is another neural network-based framework that proposes a lightweight model to detect anomalies on edge and was

[1] https://paperswithcode.com/dataset/unsw-nb15.

tested on UNSW-15 and is also considered state-of-the-art. Meftah et al. [29] performed a similar approach to [22], using Recursive Feature Elimination and Random Forests to select features, achieving up to 86% F1 accuracy. It should be noted that UNSW-NB15 is only used in this paper to measure dataset complexity, not anomaly detection. We specifically concentrate our anomaly detection on IoT datasets, especially since the high complexity of UNSW-NB15 makes it a poor choice for our algorithm.

Several other papers are published using alternative datasets that propose different machine learning models for intrusion detection. Rezvy et al. [38] proposed a deep learning framework for intrusion classification and prediction in 5G and IoT networks. They propose an autoencoder neural network for detecting intrusion or attacks in 5G and IoT networks, evaluating the model on the Aegean Wi-Fi Intrusion dataset. Their results showed an overall detection accuracy of 99.9% for different types of attacks. Kasongo and Sun [22] argue that feature selection is essential for the performance of ML models in intrusion detection since model accuracy decreases with more high-dimensional datasets. They apply a filtering technique on features and train several ML models using this technique, showing strong performance. They relate the feature selection to IoT devices with limited capacity, showing that less robust modeling techniques are favorable in limited-capacity systems such as small IoT devices.

2.4 Complexity and Anomaly Detection in Deep Learning

Neural networks trained with back propagation provide diverse structures and objectives to learn from high-dimensional data. Despite their incredible power, anomaly detection remains an open research problem, even in state-of-the-art models. Notably, several works in computer vision have shown that classification, generative, and unsupervised deep neural networks are all susceptible to anomalous data [16,20,31,34,43]. For example, one common computer vision experiment involves training a deep learning model on CIFAR-10, a dataset with 60,000 images labeled into ten classes. It is expected that the likelihood of a CIFAR-10 test image will be higher than images from other datasets during test time. However, several papers have shown that the examples from the dataset SVHN produces a higher likelihood when passed through the model trained on CIFAR-10 [31,34,43]. Recently, Serra et al. showed that anomalous high-likelihood data could be linked to complexity [43]. They find that the simplicity of SVHN data compared to CIFAR-10 data causes the deep learning model to exercise a higher likelihood on SVHN examples than the complex CIFAR-10 data. They use image compression scores as a complexity metric (likelihood ratio) to determine whether the high likelihood can be attributed to lower complexity.

2.5 Complexity and Anomaly Detection in Security

Interestingly, a similar complexity finding to [43] was found in a recent security paper [19]. Using data from various IoT devices, they find that each device has varying complexity. They formalize a complexity measure (IP Spread/IP

Depth) per device in order to fine-tune an Isolation Forest anomaly detection algorithm. Their architecture, ComplexIoT, measures network traffic on a device level, which can be used in Host Intrusion Detection Systems. This work is similar to ComplexIoT [19] in that we propose a complexity measurement; however, there are several key differences: (i) We analyze IoT datasets both from the point of view of network-level and the device level, while ComplexIoT only looks at device level complexity. (ii) ComplexIoT proposes a device complexity score to moderate the contamination rate of an Isolation Forest. This is problematic as it assumes $x\%$ of a device's traffic will be malicious given a complexity score and lead to false positives. (iii) The ComplexIoT complexity score is based on IP spread and IP depth and does not consider other network features to compute its complexity estimate. (iv) The efficacy of the ComplexIoT approach has not been measured via binary classification metrics on benign and malicious examples. In this paper, we measure the results of the weighted Hamming LID estimator on common IoT network Intrusion datasets.

3 Proposed Approach

We begin by explaining concepts and mathematics behind ID, in order to gain an intuitive understanding of the approach. We will then use this to detail LID. Next, we describe our approach to measuring LID in IoT datasets. Finally, we summarize the Weighted Hamming Distance LID Estimator, including the algorithm details, baseline models we compare against, and our experimental protocol.

3.1 Intrinsic Dimensionality

The ID of a dataset can be thought of as the minimum number of variables needed to retain a full approximation of the data [8]. It is based on the observation that high-dimensional data can often be described by a lower number of variables. The utility of lower dimensional representations is apparent throughout ML research, from data compression (such as autoencoders [21]) to dimensionality reduction (PCA). ID is akin to autoencoders and PCA, however quite distinct in that its an estimate of the lowest possible dimension of a dataset (e.g. the lowest possible bottleneck size in an autoencoder), and not a reduction technique in itself. ID can be thought of as a geometric property to measure complexity of a dataset as a whole [50].

Formally, the ID of dataset $\mathcal{X} \in \mathbb{R}^{m \times n}$, with m samples and n features, lies on a lower dimensional manifold $\mathcal{M}$, where $ID = dim(\mathcal{M})$, i.e. ID is the dimension of the manifold $\mathcal{M}$ of the data. Usually, the ID measurement is significantly less than extrinsic dimension n, or number of features.

The main approach to estimate ID involves examining the neighborhood around a reference point x_i for each x in $\mathcal{X}$. A common equation used in existing research was proposed by Levina and Bickel [27]:

$$ID(\mathcal{X}) = \left(\frac{1}{m(k-1)} \sum_{i=1}^{m} \sum_{j=1}^{k-1} log \frac{T_k(x_i)}{T_j(x_i)} \right)^{-1} \quad (1)$$

where m is the number of samples, x_i is a sample in the dataset, and k and j are integer values representing the k^{th} and j^{th} nearest neighbors from x_i. $T_k(x_i)$ is the distance between x_i and x_k, similarly, $T_j(x_i)$ is the distance between x_i and x_j. Intuitively, Eq. 1 measures the rate that new neighbors are encountered as we move out from the reference point x_i.

Recently, ID has been gaining relevance in the machine learning community [4,6,36]. Pope et al. [36] showed that common computer vision datasets exhibit very low intrinsic dimension relative to their number of pixels. They also showed that the intrinsic dimension greatly impacts learning: the higher the intrinsic dimension of a dataset, the harder it is to learn from it. In addition, they showed that the extrinsic dimension of the dataset, *i.e.*the total number of pixels per image in a dataset, did not effect learning and generalization, indicating that sample complexity only depends on the intrinsic dimension rather than the total dimension of the dataset. Ansuini et al. [6] showed that neural networks exhibit low intrinsic dimensionality at deep layers of the models. Outside of deep learning, intrinsic dimensionality has been used in applications such as anomaly detection[46], clustering, similarity search, and deformation in complex materials.

3.2 Local Intrinsic Dimensionality

In contrast to ID, LID estimates individual data samples, rather than the full dataset. It is based on the observation that individual data points in a dataset often fit within a specific lower-dimensional structure when only considering a subset of the nearby data. As a result, these values can vary greatly within a dataset. Intuitively, the LID measurement can be interpreted as the dimension immediately surrounding a data point.

LID has been proposed for anomaly/out-of-distribution detection [48] as well as detection of adversarial examples in deep neural nets [28]. Theoretically, examples within a dataset should have lower LID values than anomalous examples generated from an alternative source.

Amsaleg et al. [4] propose several estimators for the LID, though they note that these are theoretical quantities and only estimates of the true local dimension. We use their Maximum Likelihood Estimator in Sect. 3.3. Their equation provides a strong balance between efficiency and complexity:

$$\widehat{LID}(x) = - \left(\frac{1}{k} \sum_{i=1}^{k} log \frac{r_i(x)}{r_k(x)} \right)^{-1} \quad (2)$$

where r_i is the distance of data point x to the i^{th} closest neighbor and r_k is the distance to the k^{th} neighbor. Additionally, it has been shown that hyperparameter k is sensitive and must be experimentally tuned. Equation 2 is a theoretical

quantity, and it should be noted that $\widehat{LID}(x)$ is an *estimation*. Further, the dimension estimate is usually not an integer value, except in idealized distributions and datasets.

Ma et al. [28] used LID estimates to characterize adversarial subspaces in deep learning. They showed how traditional density measures can fail to detect adversarial examples in the final layers of deep learning models, while LID measurements can better characterize these subspaces. This is because traditional measures only measure the density of neighboring points surrounding an example, whereas LID measures the rate at which new neighbors occur.

Algorithm 1. Weighted Hamming Distance LID Estimator

Require: $\mathcal{X}_{train}, \mathcal{X}_{test}$, nearest neighbors k, threshold τ
Require: $\mathcal{X}_{train}$ contains only benign examples
  for x_i in $\mathcal{X}_{test}$ **do**
    $\{\mathcal{H}_1 ... \mathcal{H}_m\} \leftarrow \mathcal{H}(x_i, \{\mathcal{X}_{train}\})$
    $\{\mathcal{H}_1 ... \mathcal{H}_n\} \leftarrow$for all distinct $\mathcal{H}_i \in \{\mathcal{H}_1 ... \mathcal{H}_m\}$
    if 0 is in $\{\mathcal{H}_1 ... \mathcal{H}_n\}$ **then**
      x_i is benign (exact match)
    else
      $LID(x_i) \leftarrow LID(\{\mathcal{H}_1 ... \mathcal{H}_m\}, k)(Eq.2)$
      if $LID(x_i) \leq \tau$ **then**
        x_i is benign
      else
        x_i is malicious
      end if
    end if
  end for

3.3 Weighted Hamming Distance LID Estimator

LID is typically measured on a sample using distances from its neighbors in a dataset, and can be thought of as the *rate of growth* between a point and its neighbors. In this work, we use Eq. 2 for LID estimation, using the training data $\mathcal{X}_{train}$ as neighboring points.

Distance Metric. For the distance metric required in Eq. 2, we use Hamming Distance to compute similarity between both categorical and continuous feature points. Hamming distance is computed for continuous features by setting them to mismatching if they don't match exactly. While Euclidean Distance is typically used in Eq. 2 to measure LID, Ma et al. [28] suggested not using Euclidean Distance as the underlying distance metric. Choosing the Hamming Distance metric over Euclidean Distance for continuous variables showed better experimental results. Effectively, this turns each pairwise feature distance into a binary metric: 0 for same, 1 for different. We compute Hamming Distance using the Python SciPy library as:

$$\mathcal{H}(x_i, x_j) = \frac{\text{Number of mismatching features}}{\text{Total Features}} \tag{3}$$

Entropy. We calculate entropy of each feature and set it as the weight. In a dataset with n features, we set weight w_i for feature i to $n/Entropy(i)$, where the entropy of a feature i is:

$$-\sum_{j=1}^{m} p_j log_2(p_j) \tag{4}$$

and n is the total number of features and j are specific classes in the feature. p_j is calculated by getting the counts of each class within feature i. For example, the protocol feature may have TCP and UDP classes, we compute the counts for each to calculate entropy. We find that features with low entropy should be weighted more since they are stable properties of benign samples. For example, if benign examples come from TCP protocol 99% of the time, we can theorize that new examples matching the TCP protocol may be similar to a benign example.

After computing Hamming distance between x_i and the set of $\mathcal{X}_{train}$, we have a set of $\mathcal{H}_1 ... \mathcal{H}_m$ Hamming distances the size of $\mathcal{X}_{train}$. We filter all duplicate distances where $\mathcal{H}_i = \mathcal{H}_j$, to include only a single distance value for each cluster of distances, leaving a set of $\mathcal{H}_1 ... \mathcal{H}_n$ distances where $n \leq m$. In other words, when examples have the same distance $\mathcal{H}$ to a reference point x_i, we filter them into a single distance in order to gather a unique set of distances. From here, we are able to compute the final LID score using Eq. 2.

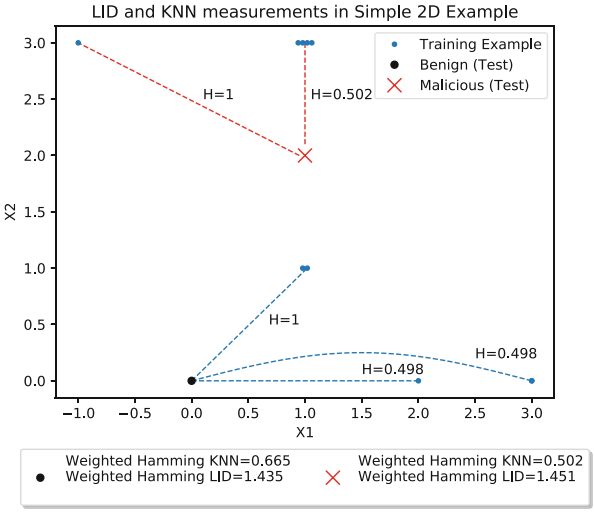

Fig. 1. A visual explanation of how the Weighted Hamming Distance LID estimator can detect anomalous examples where the traditional KNN algorithm will fail. See the end of Sect. 3.3 for more details.

Figure 1 depicts a visual example of how the Weighted Hamming LID Estimator can classify an anomalous example where the KNN algorithm will fail. For example, the traditional KNN algorithm can discriminate test examples as benign or malicious by measuring a samples average distance to K training samples, where lower average distances indicate a benign example and higher distances indicate a malicious example. Figure 1 shows how KNN can fail on a simple two dimensional problem: the red X is closer to several training examples, however, the black dot matches the more relevant feature, X2. We show how the LID estimate corrects this issue, yielding a lower value for the benign example compared to the malicious example. We weight the Hamming distances in the KNN and LID estimates with 2/Entropy of each feature. Entropy for X1 is 1.99 and 2.01 for X2. In the image, results are presented with a weighted Hamming distance, however Euclidean distance yields the same results on KNN when $K = 3$. Here, the LID is calculated as $-1/ln(0.502)$ and $-1/ln(0.498)$.

3.4 Baseline Models

In the previous section, we explain our proposed Weighted Hamming Distance LID Estimator model. We compare our proposed model with several models as baselines in the tasks of detecting attacks in the IoT networks. These models include include several modern and classic algorithms: KNN, Isolation Forest, and Autoencoder. In the following, we briefly explain each of these models.

KNN Algorithm. K-Nearest Neighbors (KNN) classification is an unsupervised machine learning model that measures the distance between a sample point and its neighbors. It takes an arbitrary distance measurement and measures the average distance between a reference point and its neighbors using this distance. In our experiments, we take the average of these nearest neighbor distances and use them to threshold scores. Theoretically, reference points with lower KNN averages should belong to the normal, non-malicious examples in the dataset, and malicious examples should have higher KNN averages.

Isolation Forest. Isolation Forest is an efficient algorithm to determine anomalies in an unsupervised manner. It does not need a profile of what is normal and not normal and identifies anomalies independent of labels. The algorithm relies on the tendency of anomalies to be easier to separate from the rest of the sample compared to normal points.

Notably, an Isolation Forest was used in [19], where they tune the contamination parameter based on the complexity of an IoT device. They argue that devices with low complexity should have contamination values close to zero because their expected network traffic should fit certain patterns, hence the device should never receive anomalous traffic. This means that, should the device be compromised, the algorithm would likely not classify the attack as anomalous because of the low expected contamination. As a result, the algorithms assumption that a certain percentage of examples are contaminated makes it vulnerable to changes in the number of contaminated records.

Autoencoder. Autoencoder is a neural network based model commonly used for unsupervised anomaly detection, such as in [38]. The model compresses the training set into a bottleneck representation before reconstructing it. We train the autoencoder on benign examples, and in theory, the reconstruction loss should be smaller for all benign examples compared to malicious examples.

The objective function is the reconstruction loss: given example x and continuous feature x_c we use mean-squared error:

$$\mathcal{L}(x_c, x'_c) = ||x_c - x'_c||^2 \tag{5}$$

where x' is the reconstructed output of the neural network. For discrete categorical features, we encode the categories into embedding layers to input them into the model. The outputs of the autoencoder for categorical variables are one-hot vectors, denoted x_d, and we use cross entropy for the objective function:

$$\mathcal{L}(x_d, x'_d) = x_d log(x'_d) + (1 - x_d) log(1 - x'_d) \tag{6}$$

We sum the loss of the continuous and categorical variables to obtain the full reconstruction loss. We use Adam optimization with the default learning rate.

3.5 Experimental Setup

To train each model in an unsupervised manner, we first take all clean examples ($\mathcal{X}_{benign}$). For each experiment, we run the algorithm on $\mathcal{X}_{benign}$ using leave-one-out cross validation,*i.e.*, calculate distances $\mathcal{H}$ from x_i on each member of the training set but x_i. We also pass it the entire set of malicious data, $\mathcal{X}_{malicious}$. The result of each sample is either 0 distance, exact match, or a weighted Hamming LID estimate. Zero's are automatically classified as benign, while lower LID estimates are also classified as benign.

A proper threshold τ can be determined based on the desired accuracy rate. In other words, if its important to classify all malicious samples, we can set a lower τ for higher detection, though this may lead to some benign examples being classified as malicious (false positive).

To measure results, we use Receiver Operating Characteristic curve (ROC) and Precision-Recall curve (PR). ROC plots the True Positive Rate (TPR) against False Positive Rate (FPR). PR plots the precision versus the recall calculated by following formulas:

$$precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{7}$$

$$recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{8}$$

4 Datasets

This section summarizes the datasets we use in our experiments. We use three IoT related datasets: (*TON_IoT*, NF Bot-IoT, IoT-23,).

4.1 TON IoT

TON_IoT [3,32], published in 2020, comprises heterogeneous IoT data across several devices. The work uses several data source types, including sensor, raw, and log data. Additionally, it includes several infrastructure layers in the testbed architecture, such as the edge, fog, and cloud layers with nine types of generated attacks: Distributed Denial-of-Service (DDoS), Scanning, Ransomware, Backdoor, and Injection attacks. The dataset has 41 total features; however, the authors recommend not to use source IP/port and destination IP/port. The dataset simulates traffic from seven IoT sensors: weather, smart garage door, smart fridge, smart TCP/IP Modbus, GPS tracker, motion-enabled light, and a smart thermostat.

4.2 NF Bot-IoT

NF Bot-IoT [42] is a dataset based on the BotNet IoT dataset [24,25]. Botnets are an important attack vector to protect against as they have been the source of several breaches over the past few years [24]. NF Bot-IoT converts four common network NIDS datasets into network flow datasets using the commonly deployed NetFlow [12] protocol for network traffic collection. Authors argue NetFlow's features are easier to extract compared to the complex features used in the original NIDS datasets since NetFlow's features are usually extracted from packet headers. The dataset includes several attacks, including DDoS, Denial-of-Service (DoS), OS and Service Scan, Keylogging, and Data exfiltration attacks.

4.3 IoT-23

IoT-23 [15] was released in 2020. The dataset has 23 different scenarios, of which three are benign traffic scenarios captured on real IoT devices. The dataset contains almost 11 million total records; however, with the difficulty of modeling this much data, we sample a million records with the following logic: From the entire dataset, we sample 500K malicious records and 500K benign examples from simulated files that contain a source IP or destination IP with an internal IP address and have at least 50 samples belonging to that specific IP address. We find that 99.99% of internal IPs have at least 50 samples. We also included all samples from three real devices (Philips HUE smart LED lamp, Amazon Echo, and a Somfy smart door lock) with 1,634 total packets.

We categorize these devices similar to IoT-Sense as a light (Philips HUE smart LED lamp), Smart Controller (Amazon Echo), and Appliance (Somfy smart door lock). Philips HUE light is in both datasets so that it can be used for comparison device-specific ID measurements.

This dataset also captures 20 simulated scenarios of both benign and malicious traffic. It offers several attack examples: DDoS, FileDownload (to infected device), HeartBeat (indicates packets sent on the connection are used to keep track of infected host by CC server), Mirai, Torii, and Okiru BotNets (new common attacks), and HorizontalPortScan (used to gather information for further attacks).

Table 1. Dataset summaries including total number of samples, percentage of benign samples, percentage of malicious samples, percent duplicates, number of features, and number of attacks.

Name	Samples	Ben(%)	Mal(%)	Dup (%)	Feats(#)	Attk(#)
UNSW-NB15	82K	45%	55%	12.5%	42	-
KDD Cup	24K	45.8%	54.2%	0%	41	-
TON_IoT	461K	65%	35%	62%	38	9
IoT-23	1M	50%	50%	2%	19	7
IoT Sense	54K	100%	0	63%	21	-
NF Bot-IoT	599K	21.7%	78.3	0%	12	4

4.4 IoT Dataset Features

We use the available features for each dataset, except we exclude source and destination IP and port as well as any ID or timestamp columns for *TON_IoT* and IoT-23. We include IPs and ports in NF Bot-IoT because of its small number of available features to provide more discriminability. Features among the datasets include protocol, source, and destination bytes, connection state, service, duration, missed bytes, number of packets, window size, payload, entropy, DNS, SSL, and HTTP properties.

5 Anomaly Detection Results

In this section, we summarize our findings of LID measurements on IoT networks, showing that we can use the localized complexity measurement of LID to detect anomalous behavior.

5.1 Algorithm Comparison Results

Gorbett et al. [17] showed that benign IoT network datasets and devices exhibit low ID measurements. In this section, we extend this finding to the sample level, showing that malicious examples have higher LID. We use this finding for the task of anomaly detection, using three public IoT datasets with benign and malicious examples. Results are compared against each other in Table 2.

Table 2 reports results of our proposed Weighted Hamming LID versus other baseline models of KNN and Weighted Hamming KNN, Isolation Forest, and the Autoencoder. We run experiments with four different K values of 3, 5, 10, and 20 for the KNN and Weighted Hamming algorihtms, and report the best findings of each. In all three cases, the same K value performed the best between KNN, Weighted KNN, and Weighted Hamming LID algorithms for the given dataset. For NF Bot-IoT $K = 5$, *TON_IoT* $K = 10$, and IoT-23 $K = 3$.

Our algorithm outperforms baseline models in all but one case (*TON_IoT* PR performs best with the Autoencoder). The most notable results were with the IoT-23 and NF Bot-IoT datasets, where the algorithm outperformed other unsupervised learning algorithms in both ROC and PR scores.

Table 2. Results of the Weighted Hamming LID algorithm on 3 datasets, plus the results of four baseline models. We experiment with K values of $3, 5, 10$, and 20 for the KNN, KNN (Weighted Hamming), and Weighted Hamming LID estimators. We choose the K with the best result for each experiment, with K = 5, 10, and 3 for NF Bot-IoT, *TON_IoT*, and IoT-23 respectively. The best ROC and PR for each dataset is in bold.

	NF Bot-IoT		***Ton-IoT***		**IoT-23**	
Test Type	ROC	PR	ROC	PR	ROC	PR
Isolation Forest	0.957	0.999	0.567	0.364	0.492	0.594
KNN	0.961	0.999	0.973	0.943	0.990	0.970
Weighted Hamming KNN	0.955	0.998	0.973	0.944	0.990	0.918
Autoencoder	0.944	0.998	0.981	**0.985**	0.981	.972
Weighted Hamming LID (Ours)	**0.970**	**0.999**	**0.985**	0.983	**0.998**	**0.994**

Table 3. Anomaly Detection Results specific attack type available in each dataset. Varied results indicate networks may be more prone specific attacks in IoT networks.

Dataset	**Attack**	**Sample Size**	**ROC**	**PR**
IoT-23	Horiz.PortScan	199,283	0.998	0.989
	DDoS	54,750	0.999	0.892
	Okiru	53,959	0.999	0.701
	C&C	271	0.999	0.632
NF Bot-IoT	Theft	1,909	0.990	0.902
	DDoS	56,844	0.999	0.999
	DoS	56,833	0.998	0.999
	Recon.	470,655	0.963	0.999
TON_IoT	Scanning	19,995	0.993	0.812
	DoS	19,994	0.987	0.755
	Injection	19,930	0.995	0.932
	DDoS	19,790	0.966	0.838
	Password	17,428	0.970	0.746
	XSS	8,914	0.952	0.727
	Ransomware	7,221	0.983	0.511
	Backdoor	19,908	0.983	0.621
	MITM	1,041	0.988	0.523

Notably, algorithms that performed distance computations to their closest neighbors exhibited the best results (KNN, weighted Hamming KNN, and LID algorithms). Isolation Forest had low results for both *TON_IoT* and IoT-23. The Autoencoder did well on two out of the three datasets, but did not perform as well on NF Bot-IoT.

The Weighted Hamming LID algorithm showed the most consistent results across all three datasets, highlighting its efficacy and promise as a robust approach.

These results indicate it is a strong alternative to classic anomaly detection algorithms such as the Autoencoder, Isolation Forest, and K-nearest neighbors. The algorithm shows promising results across three unique IoT datasets, indicating it generalizes well to several types of attacks and network datasets.

5.2 Attacks Specific Results

In this experiment, we break down our results by attack in each IoT dataset using the Weighted Hamming Distance LID estimator. We group some IoT-23 attacks based on their broad category; for example, we group File Download and Heartbeat attacks as C&C, because these attacks both come from a known C&C server and they have a small overall sample size. The results of this experiment is reported in Table 3,

Results are varied for IoT-23 and *TON_IoT*; stronger results for NF Bot-IoT. While some attacks are easily recognizable by our algorithm, such Distributed Denial-of-Service (DDoS) and DoS attacks, others do very poorly, such *TON_IoT*'s Ransomware, Backdoor, and Man-in-the-middle attack (MITM) attacks.

The first two datasets in Table 3, IoT-23 and NF Bot-IoT, performed well on all ROC metrics for each attack. Precision-Recall metrics performed slightly worse than ROC in IoT-23, with lower values for Okiru and C&C attacks. We found that both attacks had network packets that were largely indistinguishable from benign network. For example, Okiru attacks were generally TCP protocol with packet count of 1 and low byte count, similar to many benign network packets.

While the results of attack detection using the Weighted Hamming Distance LID estimator are stronger than other algorithms, results on specific attack types show that IoT networks may still be vulnerable in certain cases. In particular, *TON_IoT* did not perform as well, leading us to further analyze the data. We found that of 300K benign examples, 61.8% were packets with an exact match with another benign packet, *i.e.*, all 38 features had the same value. Further, more than 26K malicious examples had an exact match with at least one benign sample. These factors indicate that the data generated for *TON_IoT* has low discriminability when excluding source and destination IPs and ports, as we did. We note in Sect. 4.4 that the authors of the original papers [3,32] performed supervised classification on *TON_IoT*, yielding high accuracy using both source and destination IPs and ports. They recommended removing source and destination IPs and ports for further experimentation, however, the remaining 38 features contained identical or close to identical values for benign and malicious examples.

6 Conclusion

In this work, we view several network datasets through the lens of complexity and show that IoT datasets exhibit a lower ID complexity estimate than standard network collections. This finding extends to the point-wise estimation of complexity, where individual samples in (benign) IoT datasets contain low LID measures. We show that benign examples can be identified by either 1) exactly

matching the features of a training set sample or 2) by a low LID estimate. We propose a novel algorithm for detecting malicious actors in an unsupervised manner, providing the ability to deploy a model into production with only two hyperparameters needed (k value for distance measurements and threshold value τ). The algorithm is based on the theoretical LID estimation using the Hill MLE estimator, using an entropy weighted Hamming distance for measuring distances between points and features.

Future Work. Several avenues can be explored to extend this work in future research. First, evaluating our model with a broader set of attacks would enable a more robust picture of LID's strengths and weaknesses. Second, this work focused on regular IoT networks. In the future, we want to investigate the applicability of our approach on specialized IoT networks, such as industrial IoT. Lastly, evaluating the LID metric in more complex domains, such as more complex network architectures, would provide a broader picture of the algorithms capabilities.

References

1. Ahmad, R., Alsmadi, I.: Machine learning approaches to IoT security: a systematic literature review. Internet Things. **14**, 100365 (2021). https://doi.org/10.1016/j.iot.2021.100365, https://www.sciencedirect.com/science/article/pii/S2542660521000093
2. Alaba, F.A., Othman, M., Hashem, I.A.T., Alotaibi, F.: Internet of Things security: a survey. J. Netw. Comput. App. **88**, 10–28 (2017). https://doi.org/10.1016/j.jnca.2017.04.002, https://www.sciencedirect.com/science/article/pii/S1084804517301455
3. Alsaedi, A., Moustafa, N., Tari, Z., Mahmood, A., Anwar, A.: TON_IoT telemetry dataset: a new generation dataset of IoT and IIoT for data-driven intrusion detection systems. IEEE Access **8**, 165130–165150 (2020). https://doi.org/10.1109/ACCESS.2020.3022862
4. Amsaleg, L., et al.: Estimating local intrinsic dimensionality. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 29–38. KDD 2015, Association for Computing Machinery (2015). https://doi.org/10.1145/2783258.2783405
5. Andrea, I., Chrysostomou, C., Hadjichristofi, G.: Internet of things: security vulnerabilities and challenges. In: IEEE Symposium on Computers and Communication (ISCC), pp. 180–187 (2015). https://doi.org/10.1109/ISCC.2015.7405513
6. Ansuini, A., Laio, A., Macke, J.H., Zoccolan, D.: Intrinsic dimension of data representations in deep neural networks. arXiv:1905.12784 [cs, stat] (2019)
7. Azmoodeh, A., Dehghantanha, A., Choo, K.K.R.: Robust malware detection for internet of (battlefield) things devices using deep eigenspace learning. IEEE Trans. Sustain. Comput. **4**(1), 88–95 (2019). https://doi.org/10.1109/TSUSC.2018.2809665
8. Bernal, D.: 3 - Analytical techniques for damage detection and localization for assessing and monitoring civil infrastructures. In: Wang, M.L., Lynch, J.P., Sohn, H. (eds.) Sensor Technologies for Civil Infrastructures, vol. 56, pp. 67–92. Woodhead Publishing (2014). https://doi.org/10.1533/9781782422433.1.67, https://www.sciencedirect.com/science/article/pii/B978178242242650003X

9. Bezawada, B., Bachani, M., Peterson, J., Shirazi, H., Ray, I., Ray, I.: IoTSense: behavioral fingerprinting of IoT devices. arXiv:1804.03852 [cs] (2018)
10. Bhatia, S., Jain, A., Li, P., Kumar, R., Hooi, B.: MSTREAM: fast anomaly detection in multi-aspect streams.In: Proceedings of the Web Conference 2021, pp. 3371–3382 (2021). https://doi.org/10.1145/3442381.3450023, http://arxiv.org/abs/2009.08451
11. Choudhary, D.: Security challenges and countermeasures for the heterogeneity of IoT applications. J. Autonom. Intell. **1**, 16 (2019). https://doi.org/10.32629/jai.v1i2.25, http://en.front-sci.com/index.php/JAI/article/view/25
12. Claise, B.: Cisco Systems NetFlow Services Export Version 9. Request for Comments RFC 3954, Internet Engineering Task Force (2004). https://doi.org/10.17487/RFC3954, https://datatracker.ietf.org/doc/rfc3954
13. Conti, M., Dehghantanha, A., Franke, K., Watson, S.: Internet of Things security and forensics: challenges and opportunities. Future Gener. Comput. Syst. **78**, 544–546 (2018). https://doi.org/10.1016/j.future.2017.07.060, http://arxiv.org/abs/1807.10438
14. Elrawy, M.F., Awad, A.I., Hamed, H.F.A.: Intrusion detection systems for IoT-based smart environments: a survey. J. Cloud Comput. **7**(1), 21 (2018). https://doi.org/10.1186/s13677-018-0123-6
15. Garcia, S., Parmisano, A., Erquiaga, M.J.: IoT-23 dataset: a labeled dataset of malware and benign IoT traffic. (version 1.0.0) [data set] zenodo. https://www.stratosphereips.org/datasets-iot23
16. Gorbett, M., Blanchard, N.: Utilizing network properties to detect erroneous inputs. arXiv:2002.12520 [cs] (2020)
17. Gorbett, M., Shirazi, H., Ray, I.: The intrinsic dimensionality of IoT networks. In: Proceedings of the 2022 ACM Symposium on Access Control Models and Technologies (SACMAT) (2022)
18. HaddadPajouh, H., Dehghantanha, A., Khayami, R., Choo, K.K.R.: A deep Recurrent Neural Network based approach for Internet of Things malware threat hunting. Future Gener. Comput. Syst. **85**, 1-9 (2018). https://doi.org/10.1016/j.future.2018.03.007, https://www.sciencedirect.com/science/article/pii/S0167739X1732486X
19. Haefner, K., Ray, I.: ComplexIoT: Behavior-based trust for IoT networks. In: 2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), pp. 56–65 (2019). https://doi.org/10.1109/TPS-ISA48467.2019.00016
20. Hendrycks, D., Gimpel, K.: A Baseline for detecting misclassified and out-of-distribution examples in neural networks. arXiv:1610.02136 [cs] (2018)
21. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (Jul 2006). https://doi.org/10.1126/science.1127647, https://www.science.org/doi/10.1126/science.1127647
22. Kasongo, S.M., Sun, Y.: Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. J. Big Data **7**, 105 (2020). https://doi.org/10.1186/s40537-020-00379-6
23. Kollolu, R.: A Review on wide variety and heterogeneity of IoT platforms. SSRN Scholarly Paper ID 3912454, Social Science Research Network, Rochester, NY (2020). https://doi.org/10.2139/ssrn.3912454, https://papers.ssrn.com/abstract=3912454

24. Koroniotis, N., Moustafa, N., Sitnikova, E., Slay, J.: towards developing network forensic mechanism for botnet activities in the IoT based on machine learning techniques. In: Mobile Networks and Management. pp. 30–44. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-90775-83
25. Koroniotis, N., Moustafa, N., Sitnikova, E., Turnbull, B.: Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset. Future Gener. Comput. Syst. **100**, 779–796 (2019). https://doi.org/10.1016/j.future.2019.05.041, https://www.sciencedirect.com/science/article/pii/S0167739X18327687
26. Kozik, R., Pawlicki, M., Choraś, M.: A new method of hybrid time window embedding with transformer-based traffic data classification in IoT-networked environment. Pattern Anal. Appl. **24**(4), 1441–1449 (2021). https://doi.org/10.1007/s10044-021-00980-2, https://www.mendeley.com/catalogue/92cc3e51-9dc9-3c8e-9e05-aeea7382b93c/
27. Levina, E., Bickel, P.J.: Maximum Likelihood estimation of intrinsic dimension. In: Proceedings of the 17th International Conference on Neural Information Processing Systems, pp. 777–784. NIPS 2004, MIT Press, Cambridge, MA, USA (2004)
28. Ma, X.,et al.: Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality. arXiv:1801.02613 [cs] (2018)
29. Meftah, S., Rachidi, T., Assem, N.: Network based intrusion detection using the UNSW-NB15 dataset. Int. J. Comput. Digital Syst. **8**, 478–487 (2019). https://doi.org/10.12785/ijcds/080505, https://journal.uob.edu.bh:443/handle/123456789/3580
30. Mohsin, M., Anwar, Z., Husari, G., Al-Shaer, E., Rahman, M.A.: IoTSAT: a formal framework for security analysis of the internet of things (IoT). In: 2016 IEEE Conference on Communications and Network Security (CNS), pp. 180–188 (2016). https://doi.org/10.1109/CNS.2016.7860484
31. Morningstar, W., Ham, C., Gallagher, A., Lakshminarayanan, B., Alemi, A., Dillon, J.: Density of states estimation for out of distribution detection. In: Proceedings of The 24th International Conference on Artificial Intelligence and Statistics, pp. 3232–3240. PMLR (2021). https://proceedings.mlr.press/v130/morningstar21a.html
32. Moustafa, N.: A new distributed architecture for evaluating AI-based security systems at the edge: network TON_iot datasets. Sustain. Cities Soc. **72**, 102994 (2021). https://doi.org/10.1016/j.scs.2021.102994, https://www.sciencedirect.com/science/article/pii/S2210670721002808
33. Moustafa, N., Slay, J.: 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), pp. 1–6 (2015). https://doi.org/10.1109/MilCIS.2015.7348942
34. Nalisnick, E., Matsukawa, A., Teh, Y.W., Gorur, D., Lakshminarayanan, B.: Do Deep Generative Models Know What They Don't Know? (2018). https://openreview.net/forum?id=H1xwNhCcYm
35. Tapping AI for Intrusion Detection Systems, October 2021. https://www.iotworldtoday.com/2021/10/18/tapping-ai-for-intrusion-detection-systems/
36. Pope, P., Zhu, C., Abdelkader, A., Goldblum, M., Goldstein, T.: The Intrinsic Dimension of Images and Its Impact on Learning (2020). https://openreview.net/forum?id=XJk19XzGq2J

37. Rashma, B.M., Macherla, S., Jaiswal, A., Poornima, G.: Handling heterogeneity in an IoT infrastructure. In: Patnaik, S., Yang, X.-S., Sethi, I.K. (eds.) Advances in Machine Learning and Computational Intelligence. AIS, pp. 635–643. Springer, Singapore (2021). https://doi.org/10.1007/978-981-15-5243-4_60
38. Rezvy, S., Luo, Y., Petridis, M., Lasebae, A., Zebin, T.: An efficient deep learning model for intrusion classification and prediction in 5G and IoT networks. In: 2019 53rd Annual Conference on Information Sciences and Systems (CISS), pp. 1–6 (2019). https://doi.org/10.1109/CISS.2019.8693059
39. Rizvi, S., Orr, R., Cox, A., Ashokkumar, P., Rizvi, M.R.: Identifying the attack surface for IoT network. Internet Things. **9**, 100162 (2020). https://doi.org/10.1016/j.iot.2020.100162, https://www.sciencedirect.com/science/article/pii/S2542660520300056
40. Roy, B., Cheung, H.: A Deep Learning Approach for Intrusion Detection in Internet of Things using Bi-Directional Long Short-Term Memory Recurrent Neural Network, pp. 1–6 (2018). https://doi.org/10.1109/ATNAC.2018.8615294. ISSN: 2474-154X
41. Sahu, A.K., Sharma, S., Tanveer, M., Raja, R.: Internet of Things attack detection using hybrid Deep Learning Model. Comput. Commun. **176**, 146–154 (2021). https://doi.org/10.1016/j.comcom.2021.05.024, https://www.sciencedirect.com/science/article/pii/S0140366421002164
42. Sarhan, M., Layeghy, S., Moustafa, N., Portmann, M.: NetFlow datasets for machine learning-based network intrusion detection systems. In: Deze, Z., Huang, H., Hou, R., Rho, S., Chilamkurti, N. (eds.) BDTA/WiCON -2020. LNICST, vol. 371, pp. 117–135. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-72802-1_9
43. Serrà, J., Álvarez, D., Gómez, V., Slizovskaia, O., Núñez, J.F., Luque, J.: Input Complexity and Out-of-distribution Detection with Likelihood-based Generative Models (2019). https://openreview.net/forum?id=SyxIWpVYvr
44. Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q.: A deep learning approach to network intrusion detection. IEEE Trans. Emerg. Topics Comput. Intell. **2**, 41–50 (2018). https://doi.org/10.1109/TETCI.2017.2772792
45. Singh, P., Jaykumar, J., Pankaj, A., Mitra, R.: Edge-Detect: Edge-centric Network Intrusion Detection using Deep Neural Network. arXiv:2102.01873 [cs], February 2021
46. Stolz, B.J., Tanner, J., Harrington, H.A., Nanda, V.: Geometric anomaly detection in data. Proc. Natl. Acad. Sci. **117**(33), 19664–19669 (2020). https://doi.org/10.1073/pnas.2001741117, https://www.pnas.org/content/117/33/19664
47. Vasudevan, A., Harshini, E., Selvakumar, S.: SSENet-2011: a network intrusion detection system dataset and its comparison with KDD CUP 99 dataset. In: 2011 Second Asian Himalayas International Conference on Internet (AH-ICI), pp. 1–5 (2011). https://doi.org/10.1109/AHICI.2011.6113948
48. Wang, Q., Erfani, S.M., Leckie, C., Houle, M.E.: A Dimensionality-Driven Approach for Unsupervised Out-of-distribution Detection, p. 9 (2021)
49. Zhao, K., Ge, L.: A Survey on the Internet of Things Security, pp. 663–667 (2013). DOI: https://doi.org/10.1109/CIS.2013.145
50. Zhou, S., Tordesillas, A., Pouragha, M., Bailey, J., Bondell, H.: On local intrinsic dimensionality of deformation in complex materials. Sci. Rep. **11**(1), 10216 (2021). https://doi.org/10.1038/s41598-021-89328-8, https://www.nature.com/articles/s41598-021-89328-8