A Review and Analysis of the Bot-IoT Dataset

Jared M. Peterson*, Joffrey L. Leevy*, Taghi M. Khoshgoftaar*
*Florida Atlantic University
Email: jpeterson2019@fau.edu, jleevy2017@fau.edu, khoshgof@fau.edu

2021 IEEE International Conference on Service-Oriented System Engineering (SOSE) | 978-1-6654-3477-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/SOSE52839.2021.00007

***Abstract*—Machine learning is rapidly changing the cybersecurity landscape. The use of predictive models to detect malicious activity and identify inscrutable attack patterns is providing levels of automation that are desperately needed to level the playing field between malicious actors and network defenders. This has led to increased research at the intersection of machine learning and cybersecurity and also the creation of many publicly available datasets. This paper provides an in-depth, unique review and analysis of one of the newest datasets, Bot-IoT. The full dataset contains about 73 million instances (big data). Models trained on Bot-IoT are capable of detecting various botnet attacks in *Internet of Things* (IoT) networks. The purpose of this paper is to provide researchers with a fundamental understanding of Bot-IoT, its features, and some of its pitfalls. We also discuss data cleaning procedures and briefly summarize the use of the dataset in published research.**

***Index Terms*—Bot-IoT, data cleaning, feature analysis, machine learning, big data.**

I. Introduction

With the number of networked devices increasing every year, the challenge of securing these devices is also increasing. Organizations and businesses around the world are charged with protecting these new and vast networks but are overburdening security analysts with a massive influx of data. The most obvious solution has been the turn to automation, specifically machine learning, to identify and respond to attacks without the need for constant human supervision. Machine learning models can drastically improve the detection rate of traditional static signature-based detection systems while reducing human workload. To that end, there has been a large amount of research on machine learning applications for cybersecurity. Many researchers choose to investigate different machine learning models and thereby leverage publicly available datasets to train and test their models.

A significant part of the machine learning process is the analysis of data that is used to train a predictive model. This can be a daunting task, especially when a dataset requires additional knowledge of the problem domain. A clear understanding of a dataset is of critical importance in machine learning, and misunderstood or poorly generated features can lead to models that produce invalid results.

During data cleaning, it is important to understand the significance of a particular feature, its expected type and range of values [1]. Without this information, it would be difficult to discern between valid and invalid values. Understanding a feature's potential range of values ensures that the proper lower and upper bounds are used in the normalization process, even if the dataset does not contain them. This is important for a model to maintain its ability to generalize. The generalizability of a model can also be affected during the feature selection process. Features that contain environment- or dataset-specific information can cause models to overfit. This is especially true for network security data, which can contain features that are only relevant to a single network.

In this paper, we provide an in-depth analysis of the Bot-IoT dataset developed by Koroniotis et al. [2]. Bot-IoT was created in 2018 and published in 2019 by the *University of New South Wales* (UNSW) as a modernized and realistic dataset for training models to detect botnet attacks in *Internet of Things* (IoT) networks. IoT refers to a network of devices, not normally regarded as computers, that have Internet connectivity and limited computing capability [3].

The main contribution of this paper is to provide a clear understanding of Bot-IoT, its subsets, and its features. We also identify features of the dataset that may invalidate performance results, and to the best of our knowledge, we are the first to do this. Our work goes beyond the scope of a traditional review paper. In addition, we analyze published Bot-IoT works and rely on our empirical research to establish the validity of features used in those works.

The remainder of this paper is organized as follows: Section II discusses the composition of Bot-IoT; Section III analyzes the features contained within the dataset; Section IV discusses the data cleaning of Bot-IoT; Section V provides an overview of papers that have incorporated the dataset; and Section VI concludes with the main points of this paper.

II. Dataset Description

Bot-IoT was developed in a testbed consisting of multiple virtual machines with various *operating system*s (OSs), network firewalls, network taps, the Node-red tool [4], and the Argus network security tool [2], [5]. Bot-IoT is comprised of multiple sets and subsets that are different in file format, size, and feature count. A full listing of features, their definitions, and associated sets and subsets are shown in Table III (Appendix).

The first set, referred to as the Raw Set, contains roughly 70 GB of *packet capture* (PCAP) files. These files contain the network data that was intercepted in the testbed environment from network taps. Since this set is raw, it needs to be processed by a network analysis tool such as Wireshark [6], Argus [5], or Zeek [7] before it is usable for traditional machine learning models. This also means the features produced by the Raw Set can vary based on the tool utilized to process the PCAP.

2642-6587/21/$31.00 ©2021 IEEE
DOI 10.1109/SOSE52839.2021.00007

TABLE I
BOT-IOT: FULL SET

Category	Subcategory	No. of Instances
Normal	Normal	9,543
DoS	TCP	12,315,997
	UDP	20,659,491
	HTTP	29,706
DDoS	TCP	19,547,603
	UDP	18,965,106
	HTTP	19,771
Reconnaissance	OS Fingerprinting	358,275
	Service Scanning	1,463,364
Information Theft	Keylogging	1,469
	Data Exfiltration	118

The second set, referred to as the Full Set, contains *comma-separated values* (CSV) files that add up to roughly 73 million instances, which are generated by the Argus network security tool. It is important to note that each instance is representative of a network session. The features of a session represent an aggregation of all the bytes and packets related to a single communication session between two hosts. The Full Set has the fewest total features of any of the processed sets or subsets. It contains 26 independent features and 3 dependent features. The 26 independent features only include the network flow data from Argus and do not contain the 14 additional calculated features developed by Koroniotis et al. [2]. It is also worth noting that the CSV files for the Full Set do not contain a header row, and each file has six columns interspersed in the features that do not contain any data. These details can be important when attempting to convert the data into another format or importing the data into a tool for analysis. Table I shows categories and subcategories of network traffic for the Full Set.

The first subset of the Bot-IoT dataset, referred to as the 5% Subset, was provided and recommended by Koroniotis et al. [2] as a smaller and more manageable version. It contains 5% of the instances of the original set, roughly 3.6 million, and is a representative sample of the Full Set in terms of attack category. The 5% Subset contains the most features of any processed set or subset of Bot-IoT, with 43 independent features and 3 dependent features. The 43 independent features contain the Argus network flow features and the additional calculated features. The 5% Subset is divided into five CSV files, each containing its own header row with feature names. Table II shows categories and subcategories of traffic for the 5% Subset.

The final subset, referred to as the 10-Best Subset, contains the same number of instances as the original 5% Subset (see Table II) but contains only 10 of the original 43 independent features. Each file contains 3 dependent features and 16 independent features, of which 6 serve as session identifiers and 10 are considered by Koroniotis et al. [2] to be the best. The 10-Best features were derived through the mapping of the correlation coefficient and joint entropy of the 43 independent features in the 5% Subset. The 10-Best features were selected based on their rankings. This subset is contained within two CSV files that both have a header row in the feature names.

TABLE II
BOT-IOT: 5% SUBSET AND 10-BEST SUBSET

Category	Subcategory	No. of Instances
Normal	Normal	477
DoS/DDoS	TCP	1,593,180
	UDP	1,981,230
	HTTP	2,474
Reconnaissance	OS Fingerprinting	17,914
	Service Scanning	73,168
Information Theft	Keylogging	73
	Data Exfiltration	6

The Raw Set, Full Set, 5% Subset, and 10-Best Subset contain big data. Big data is defined by specific properties, such as volume, variety, variability, velocity, complexity, and value [8]. These properties may increase the difficulty of the classification task for learners trained on big data. With regard to binary classification of normal and attack instances, Tables I and II show that the datasets are severely imbalanced. Class imbalance arises from a disproportionate number of majority class instances (attack instances in Bot-IoT) and could potentially skew the results of big data analytics [9]. We point out that the class imbalance problem is compounded by big data.

III. FEATURE ANALYSIS

A common goal for any cybersecurity machine learning study is to produce generalizable models that can accurately identify malicious behavior. To this end, it is extremely important to understand the data and its features, so that the models can be trained and tested on valid information. We analyzed all 46 features present in the 5% Subset. This section provides information on the data type, meaning, and utility of the various features. The section is divided into several subsections, including Dependent Features, Independent Features, and Invalid Features.

A. Dependent Features

There are three dependent categorical features in the Bot-IoT dataset: *category*, *subcategory*, and *attack*. All three features are present in each of the sets and subsets with the exception of the Raw Set. *Category* is a string type feature with five potential values: Normal, DoS, DDoS, Reconnaissance, and Information Theft. These values represent the attack category for a given instance. *Subcategory* is also a string type feature but with eight potential values: Normal, *Transmission Control Protocol* (TCP), *User Datagram Protocol* (UDP), *Hypertext Transfer Protocol* (HTTP), OS Fingerprinting, Service Scanning, Keylogging, and Data Exfiltration. The *subcategory* values are used in conjunction with the *category* value to classify an instance with a more specific type of attack.

It is important to note that both *Denial-of-Service* (DoS) and *Distributed Denial-of-Service* (DDoS) categories share the same values for their subcategories. The final dependent feature is *attack*, which serves as a binary value indicating that an instance is either an attack, with a value of 1, or normal, with a value of 0.

B. Independent Features

The 5% Subset contains 43 independent features. We determined that 6 of these features are invalid, and in Section III-B3, we provide justification for invalidating these features. Of the remaining 37 features, there are 23 standard features and 14 calculated features. The standard features were generated by the Argus network security tool and are included in the Full Set or can be trivially derived from its features. The calculated features were introduced by Koroniotis et al. [2] as "additional features." These calculated features are present only in the 5% Subset and the 10-Best Subset and require some form of scripting to be calculated.

1) Standard Features: The standard features contain several categorical features. These include *flgs*, *proto*, and *state*, which are all string data types. Each of the three has a categorical integer equivalent, corresponding to *flgs_number*, *proto_number*, and *state_number*, respectively. The integer counterparts are not included in the Full Set but can be easily generated based on the values of the string features. Both sets of features contain the same categorical information, but with different data types. The features *sport* and *dport* are the only other categorical features, and they represent a unique port number that can range from 1 to 65,535. There is the possibility, depending on cleaning procedures, for the values 0 and -1 to also be present. Typically, these values represent protocols that do not use port numbers.

The standard features also include many ratio type features. These include features related to the network session of the instance, such as *pkts*, *bytes*, *dur*, *spkts*, *dpkts*, *sbytes*, *dbytes*, *rate*, *srate*, and *drate*. Each of these 10 features provides insight into the length and throughput of the session of an instance. The features *mean*, *stddev*, *sum*, *min*, and *max* contain aggregated session duration information that is calculated by the Argus network security tool. This data can help models compare the current instance duration with previous instances.

2) Calculated Features: The calculated features are all ratio type features that contain additional aggregated session information not included in the standard features. The features *TnBPSrcIP*, *TnBPDstIP*, *TnP_PSrcIP*, *TnP_PDstIP*, *TnP_PerProto*, and *TnP_Per_Dport* represent a total byte or packet count for a particular *Internet Protocol* (IP) address or protocol. Similarly, *N_IN_Conn_P_DstIP* and *N_IN_Conn_P_SrcIP* represent total connections for a source or destination IP address. The final two calculated totals are *Pkts_P_State_P_Protocol_P_DestIP* and *Pkts_P_State_P_Protocol_P_SrcIP*. These two features represent the total number of packets for sessions in the same connection state that have the same source or destination IP address. The other four calculated features are average rates: *AR_P_Proto_P_SrcIP*, *AR_P_Proto_P_DstIP*, *AR_P_Proto_P_Sport*, and *AR_P_Proto_P_Dport*. The average rate these features refer to is the number of packets divided by the duration of the session.

Since Koroniotis et al. [2] did not provide pseudo code for calculating the features, we performed our own calculations. This process gave us additional insight into the relevant algorithms. The features utilize a window of 100 instances that includes the current instance. The various totals and average rates are calculated and use the current instance as a filter. For example, *TnBPSrcIP* is the total number of bytes per source IP. It is calculated by summing the bytes feature from every instance within the current window that has the same *saddr* value as the current instance.

The inclusion of the calculated features makes it possible for a model utilizing only the valid features to properly categorize the DDoS and Reconnaissance attack categories. Without the inclusion of these features, it would be extremely difficult for any model, without some type of memory, to accurately identify these attacks. This is due to the nature of the two attacks. A DDoS attack requires multiple different hosts attacking at or near the same time. A DDoS instance would be impossible to differentiate from a DoS instance if the model has no way of knowing how many concurrent or near concurrent connections exist in proximity. The Reconnaissance category is similar, in that the primary ways of identifying service scanning or OS fingerprinting is by identifying a single host attempting connections on many ports. Without a way of determining the number of connections going to many ports that originate from a single host, it would be difficult, if not impossible, for a model to properly categorize this type of attack.

3) Invalid Features: We established that the following features are invalid: *pkSeqID*, *seq*, *stime*, *ltime*, *saddr*, and *daddr*. These features undermine the efficiency of a predictive model because they contribute to overfitting and limit generalization. We explain further in the next three paragraphs.

As *pkSeqID* and *seq* are row or sequence identifiers that only contain ordinal information, we consider them to be invalid. Unlike *pkSeqID*, *seq* has not been explicitly defined by Koroniotis et al. [2]. Furthermore, *seq* has been used in several studies as this feature appears in Koroniotis et al.'s 10-Best Feature set. After consulting the Argus developers and Argus documentation, we discovered *seq* is a monotonically increasing sequence number that provides synchronization and reliability between Argus readers. Apart from providing ordinal information about the network session, *seq* does nothing else.

The start packet time and last packet time for each instance are denoted by *stime* and *ltime*, respectively. Although using *stime* in combination with *ltime* enables other key features to be derived, most of the derived information is already available from *dur*, *rate*, *srate*, and *drate*, which are more generalizable features than *stime* and *ltime*. While *stime* and *ltime* might help improve detection of intense attacks, these timestamps may simultaneously degrade proper classification

of normal traffic during those same time windows. This can have very undesirable consequences for business operations. For example, if a major website is experiencing a severe DoS attack, it is not desirable for normal customer traffic to be incorrectly classified as attack traffic and dropped.

The source and destination IP addresses for each instance are designated by *saddr* and *daddr*, respectively. We excluded them for two reasons. Firstly, we recognize that private IP addresses are not globally unique and may vary between networks. If a trained model associates a particular activity with a specific private IP address, this association will most likely not generalize for the same IP address in another local network. The second reason relates to an inherent issue with Bot-IoT. When analyzing the 5% Subset, we observed that 100% of the attack traffic contains private IPs for both *saddr* and *daddr*, but only 36% of the normal traffic contains private IPs for both *saddr* and *daddr*. This creates a problem where 64% of normal instances can be identified because their *saddr* and *daddr* features display a public IP.

Although we consider the six features (*pkSeqID*, *seq*, *stime*, *ltime*, *saddr*, and *daddr*) to be invalid, this does not mean they are entirely unusable. Machine learning practitioners may find them beneficial in certain situations, such as for troubleshooting models, and cybersecurity analysts may decide the features are useful for forensics. There are many other examples capable of demonstrating the upside of keeping one or more of these features. However, for the purpose of our study, we believe the six features should be removed from Bot-IoT.

IV. Data Cleaning

In addition to the six invalid features that should be removed, we discovered other areas of the dataset that need to be cleaned. These additional discoveries are derived solely from working with the 5% Subset, but we believe the issues will be found in all the processed sets and subsets. First, many if not all the instances using *Internet Control Message Protocol* (ICMP) have a hexadecimal value for the sport and dport features. The ICMP port values should be changed to -1 or 0 to indicate that they do not have a valid port value. Changing the values to -1 would have an additional advantage of matching the values for instances that use *Address Resolution Protocol* (ARP). Our second observation involves the mislabeling of several instances. Specifically, we are referring to instances that use ARP and are not labeled as normal traffic. Based on the description of each of the attack categories, we determined that ARP does not contribute to the attacks. Therefore, mislabeled instances should be relabeled as normal traffic, or all instances that use ARP should be removed.

V. Literature Review

The primary focus of this section is to group the published works of research that have used the Bot-IoT dataset. Our search for papers concluded on June 20, 2021. The search revealed 47 studies that utilized the Bot-IoT dataset. To the best of our knowledge, these are all the published works that used the Bot-IoT dataset.

A. Studies with Invalid Features

There are several works that utilize the 5% Subset and its 43 independent features. As discussed previously, this set contains all six invalid features (*pkSeqID*, *seq*, *stime*, *ltime*, *saddr*, and *daddr*). However, we assume that these works did not include *pkSeqID*. We make this assumption because *pkSeqID* was clearly labeled as a row identifier in the Bot-IoT README file and the original paper [2]. In addition to the original paper, we discovered 12 studies that utilized the full independent feature set. Bhuvaneswari and Selvakumar [10], Alhowaide et al. [11], Abdel-Basset et al. [12], Wiyono and Cahyani [13], Ferrag et al. [14], Zhang et al. [15], Liaqat et al. [16], Pacheco and Sun [17], Aldhaheri et al. [18], Soe et al. [19], Ferrag et al. [20], and Alyasiri et al. [21] all mentioned using the 5% Subset with either an accompanying feature list or feature count. Of note, Bhuvaneswari and Selvakumar, Abdel-Basset et al., Wiyono and Cahyani, Zhang et al., and Aldhaheri et al. used both the 5% Subset and the 10-Best Subset. Since these papers would have used at least one if not all the invalid features, we assume their results are not valid and do not represent a good generalizable model.

Another commonly used feature set was the 10-Best Subset. This subset contains the invalid feature *seq*, and so we assume each of these papers utilized it for training and testing. We found nine studies that utilized the 10-Best Subset, five [10], [12], [13], [15], [18] of which were previously mentioned, since they also use the 5% Subset. Ibitoye et al. [22], Filus et al. [23], Lawal et al. [24], and Sriram et al. [25] all utilized the 10-Best Subset for training and testing. Since *seq* is not a valid feature that will produce generalizable models, we assume the results of these nine studies did not produce valid results.

We found only a single study utilized the 26 independent features present in the Full Set. This set includes all of the invalid features and if utilized in its entirety will cause a model to lose its ability to generalize. Just like the studies utilizing the 5% Subset, we assume that *pkSeqID* was not included. We found that Ferrag and Maglaras [26] utilized this set, and therefore, we assume their results are invalid.

Multiple studies did not use one of the provided sets of features but instead developed their own feature lists. These studies did, however, include at least one of the previously mentioned invalid features, which would have affected the validity of their conclusions. The affected studies are as follows: Oreški and Andročec [27], Alkadi et al. [28], Kumar et al. [29], Al-Zewairi et al. [30], Djenna et al. [31], Churcher et al. [32], Popoola et al. [33], Kumar et al. [34], Kumar et al. [35], and Biswas and Roy [36]. The number of invalid features varied from study to study, but the most common invalid feature was *seq*.

B. Studies with Unlisted Features

A sizeable portion of the studies captured in our search did not state or list the features used. These studies are Guizani

and Ghafoor [37], Alkadi et al. [38], Shafiq et al. [39], Cheema et al. [40], Mulyanto et al. [41], Bagui and Li [42], Susilo and Sari [43], Koroniotis et al. [44], Dwibedi et al. [45], Shafiq et al. [46], Huong et al. [47], Huong et al. [48], Venugopal et al. [49], Nimbalkar and Kshirsagar [50], and Jithu et al. [51]. Since the exact list of features used is unknown, we cannot make any conclusions about the validity of the results.

Another subset of the studies we reviewed did not utilize any of the processed sets or subsets. Instead, PCAP files from the Raw Set were used to generate features. The studies are Ge et al. [52], Costa et al. [53], and Ge et al. [54]. These studies did not use the Argus network security tool, which was used in the original study, and therefore generated completely different features. Because we do not have access to the data generated by these studies, we cannot make any conclusions about the validity of the results.

C. Studies with Only Valid Features

Two studies clearly used only valid features from Bot-IoT. These works are Demirpolat et al. [55] and Soe et al. [56]. Demirpolat et al. used 16 features comprised of the standard Argus network data with the exclusion of session identification data, timestamps, and the *seq* feature. Soe et al. used a smaller feature set, comprised of eight features, that consisted primarily of the additional features developed by Koroniotis et al. [2].

In their work, Demirpolat et al. evaluated an ensemble of prototypical networks [57] and *Support Vector Machine*s (SVMs) [58] against four other models: *Convolutional Neural Network* (CNN) [59], SVMs, Naive Bayes [60], and deep autoencoders [61]. The proposed ensemble model uses few-shot learning [62] to compensate for training machine learning models with limited data. All models were trained on three different datasets: Bot-IoT, UNSW-NB15 [63], and a software-defined networking [64] customized set. The models were implemented with Scikit-learn [65], Keras [66], and Pytorch [67]. To address class imbalance, the number of instances in each category of Bot-IoT, except the Normal and Information Theft categories, was down-sampled to 20,000. The instances for Normal and Information Theft were not down-sampled because their numbers are small in comparison. For the training set, Demirpolat et al. randomly selected instances of 100, 400, 800, and 1,000. Also, the researchers randomly selected 100 instances for the validation set and used the remaining instances from the down-sampled dataset as the test set. With regard to accuracy, precision, recall, and F-measure for Bot-IoT, the ensemble model outperformed the other models by roughly 20%. The highest Bot-IoT F-measure score for the ensemble was 96% for the DDoS category. One shortcoming of this work is the relatively small sizes of the training sets. A small sample size can lead to a model with high variance.

For their research, Soe et al. proposed a feature selection algorithm that was implemented on a Raspberry Pi device. The algorithm uses correlation-based feature selection [68] and for the calculation of the final feature set, relies on the gain-ratio [69] metric. Soe et al. utilized several tree-based classifiers, namely random forest [70], decision tree [71], Hoeffding tree [72], and logistic model tree [73]. All 477 normal instances from the 5% Subset were selected. From the attack categories, the researchers randomly selected 81,977 instances from DDoS, 82,060 from Reconnaissance, and 556 from Information Theft. The dataset was then split in a ratio of 66:34 for training and testing. In general, the results show that using the proposed selection algorithm significantly decreases the number of features without impacting classifier performance. Among the four models, decision tree performed the best, with top scores of 100% precision for DDoS and Reconnaissance, 99.99% F-measure for DDoS and Reconnaissance, 0% false positive rate for DDoS and Reconnaissance, and 99.40% true positive rate for DDoS. The fact that all the classifiers are tree-based is a limitation of this study. Classifier diversity adds credence to the results of a machine learning study.

VI. Conclusion

Data analysis is one of the most crucial aspects of the machine learning process. We emphasize that invalid features and/or poor data cleaning practices can lead to non-generalizable models and invalid performance scores. Bot-IoT is an intrusion detection dataset that trains models to detect various botnet attacks in IoT networks. Based on our data analysis of Bot-IoT, we discovered several invalid features. These features were highly utilized in several studies that we reviewed, with over 50% of the papers using one or more of the invalid features. For future studies that incorporate Bot-IoT, we recommend utilizing only valid features and adopting an effective data cleaning procedure.

Acknowledgments

We would like to thank the reviewers in the Data Mining and Machine Learning Laboratory at Florida Atlantic University. Additionally, we acknowledge partial support by the NSF (CNS-1427536). Opinions, findings, conclusions, or recommendations in this paper are the authors' and do not reflect the views of the NSF.

References

[1] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on cse-cic-ids2018 big data," *Journal of Big Data*, vol. 7, no. 1, pp. 1–19, 2020.

[2] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[3] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in internet of things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017.

[4] T. O. Foundation, "Node-red: Low-code programming for event-driven applications." https://nodered.org/.

[5] Argus, "Argus," https://openargus.org/.

[6] Wireshark, "Wireshark. go deep." https://www.wireshark.org/.

[7] T. Z. Project, "The zeek network security monitor," https://zeek.org//.

[8] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, "A survey on addressing high-class imbalance in big data," *Journal of Big Data*, vol. 5, no. 1, p. 42, 2018.

[9] R. Zuech, J. Hancock, and T. M. Khoshgoftaar, "Detecting web attacks using random undersampling and ensemble learners," *Journal of Big Data*, vol. 8, no. 1, pp. 1–20, 2021.

[10] B. A. NG and S. Selvakumar, "Anomaly detection framework for internet of things traffic using vector convolutional deep learning approach in fog environment," *Future Generation Computer Systems*, vol. 113, pp. 255–265, 2020.

[11] A. Alhowaide, I. Alsmadi, and J. Tang, "Pca, random-forest and pearson correlation for dimensionality reduction in iot ids," in *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*. IEEE, 2020, pp. 1–6.

[12] M. Abdel-Basset, V. Chang, H. Hawash, R. K. Chakrabortty, and M. Ryan, "Deep-ifs: Intrusion detection approach for iiot traffic in fog environment," *IEEE Transactions on Industrial Informatics*, 2020.

[13] R. T. Wiyono and N. D. W. Cahyani, "Performance analysis of decision tree c4. 5 as a classification technique to conduct network forensics for botnet activities in internet of things," in *2020 International Conference on Data Science and Its Applications (ICoDSA)*. IEEE, 2020, pp. 1–5.

[14] M. A. Ferrag, L. Maglaras, A. Ahmim, M. Derdour, and H. Janicke, "Rdtids: Rules and decision tree-based intrusion detection system for internet-of-things networks," *Future internet*, vol. 12, no. 3, p. 44, 2020.

[15] Y. Zhang, J. Xu, Z. Wang, R. Geng, K.-K. R. Choo, J. A. Pérez-Díaz, and D. Zhu, "Efficient and intelligent attack detection in software defined iot networks," in *2020 IEEE International Conference on Embedded Software and Systems (ICESS)*. IEEE, 2020, pp. 1–9.

[16] S. Liaqat, A. Akhunzada, F. S. Shaikh, A. Giannetsos, and M. A. Jan, "Sdn orchestration to combat evolving cyber threats in internet of medical things (iomt)," *Computer Communications*, vol. 160, pp. 697–705, 2020.

[17] Y. Pacheco and W. Sun, "Adversarial machine learning: A comparative study on contemporary intrusion detection datasets," in *Proceedings of the 7th International Conference on Information Systems Security and Privacy*, vol. 1, 2021, pp. 160–171.

[18] S. Aldhaheri, D. Alghazzawi, L. Cheng, B. Alzahrani, and A. Al-Barakati, "Deepdca: novel network-based detection of iot attacks using artificial immune system," *Applied Sciences*, vol. 10, no. 6, p. 1909, 2020.

[19] Y. N. Soe, P. I. Santosa, and R. Hartanto, "Ddos attack detection based on simple ann with smote for iot environment," in *2019 Fourth International Conference on Informatics and Computing (ICIC)*. IEEE, 2019, pp. 1–5.

[20] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102419, 2020.

[21] H. Alyasiri, J. A. Clark, A. Malik, and R. de Fréin, "Grammatical evolution for detecting cyberattacks in internet of things environments."

[22] O. Ibitoye, O. Shafiq, and A. Matrawy, "Analyzing adversarial attacks against deep learning for intrusion detection in iot networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[23] K. Filus, J. Domańska, and E. Gelenbe, "Random neural network for lightweight attack detection in the iot," in *Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*. Springer, 2020, pp. 79–91.

[24] M. A. Lawal, R. A. Shaikh, and S. R. Hassan, "An anomaly mitigation framework for iot using fog computing," *Electronics*, vol. 9, no. 10, p. 1565, 2020.

[25] S. Sriram, R. Vinayakumar, M. Alazab, and K. Soman, "Network flow based iot botnet attack detection using deep learning," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2020, pp. 189–194.

[26] M. A. Ferrag and L. Maglaras, "Deepcoin: A novel deep learning and blockchain-based energy exchange framework for smart grids," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1285–1297, 2019.

[27] D. Oreški and D. Andročec, "Genetic algorithm and artificial neural network for network forensic analytics," in *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*. IEEE, pp. 1200–1205.

[28] O. AlKadi, N. Moustafa, B. Turnbull, and K.-K. R. Choo, "Mixture localization-based outliers models for securing data migration in cloud centers," *IEEE Access*, vol. 7, pp. 114 607–114 618, 2019.

[29] P. Kumar, G. P. Gupta, and R. Tripathi, "Toward design of an intelligent cyber attack detection system using hybrid feature reduced approach for iot networks," *Arabian Journal for Science and Engineering*, vol. 46, no. 4, pp. 3749–3778, 2021.

[30] M. Al-Zewairi, S. Almajali, and M. Ayyash, "Unknown security attack detection using shallow and deep ann classifiers," *Electronics*, vol. 9, no. 12, p. 2006, 2020.

[31] A. Djenna, D. E. Saidouni, and W. Abada, "A pragmatic cybersecurity strategies for combating iot-cyberattacks," in *2020 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, 2020, pp. 1–6.

[32] A. Churcher, R. Ullah, J. Ahmad, F. Masood, M. Gogate, F. Alqahtani, B. Nour, W. J. Buchanan *et al.*, "An experimental analysis of attack classification using machine learning in iot networks," *Sensors*, vol. 21, no. 2, p. 446, 2021.

[33] S. I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, and H. Gacanin, "Hybrid deep learning for botnet attack detection in the internet of things networks," *IEEE Internet of Things Journal*, 2020.

[34] P. Kumar, G. P. Gupta, and R. Tripathi, "Tp2sf: A trustworthy privacy-preserving secured framework for sustainable smart cities by leveraging blockchain and machine learning," *Journal of Systems Architecture*, vol. 115, p. 101954, 2021.

[35] P. Kumar, R. Kumar, G. P. Gupta, and R. Tripathi, "A distributed framework for detecting ddos attacks in smart contract-based blockchain-iot systems by leveraging fog computing," *Transactions on Emerging Telecommunications Technologies*, p. e4112, 2020.

[36] R. Biswas and S. Roy, "Botnet traffic identification using neural networks," *Multimedia Tools and Applications*, pp. 1–25, 2021.

[37] N. Guizani and A. Ghafoor, "A network function virtualization system for detecting malware in large iot based networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1218–1228, 2020.

[38] O. Alkadi, N. Moustafa, B. Turnbull, and K.-K. R. Choo, "A deep blockchain framework-enabled collaborative intrusion detection for protecting iot and cloud networks," *IEEE Internet of Things Journal*, 2020.

[39] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "Corrauc: a malicious bot-iot traffic detection method in iot network using machine learning techniques," *IEEE Internet of Things Journal*, 2020.

[40] M. A. Cheema, H. K. Qureshi, C. Chrysostomou, and M. Lestas, "Utilizing blockchain for distributed machine learning based intrusion detection in internet of things," in *2020 16th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 2020, pp. 429–435.

[41] M. Mulyanto, M. Faisal, S. W. Prakosa, and J.-S. Leu, "Effectiveness of focal loss for minority classification in network intrusion detection systems," *Symmetry*, vol. 13, no. 1, p. 4, 2021.

[42] S. Bagui and K. Li, "Resampling imbalanced data for network intrusion detection datasets," *Journal of Big Data*, vol. 8, no. 1, pp. 1–41, 2021.

[43] B. Susilo and R. F. Sari, "Intrusion detection in iot networks using deep learning algorithm," *Information*, vol. 11, no. 5, p. 279, 2020.

[44] N. Koroniotis, N. Moustafa, and E. Sitnikova, "A new network forensic framework based on deep learning for internet of things networks: A particle deep framework," *Future Generation Computer Systems*, vol. 110, pp. 91–106, 2020.

[45] S. Dwibedi, M. Pujari, and W. Sun, "A comparative study on contemporary intrusion detection datasets for machine learning research," in *2020 IEEE International Conference on Intelligence and Security Informatics (ISI)*. IEEE, 2020, pp. 1–6.

[46] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and bot-iot attacks traffic identification for internet of things in smart city," *Future Generation Computer Systems*, vol. 107, pp. 433–442, 2020.

[47] T. T. Huong, T. P. Bac, D. M. Long, B. D. Thang, N. T. Binh, T. D. Luong, and T. K. Phuc, "Lockedge: Low-complexity cyberattack detection in iot edge computing," *IEEE Access*, vol. 9, pp. 29 696–29 710, 2021.

[48] T. T. Huong, T. P. Bac, D. M. Long, B. D. Thang, T. D. Luong, and N. T. Binh, "An efficient low complexity edge-cloud framework for security in iot networks," in *2020 IEEE Eighth International Conference on Communications and Electronics (ICCE)*. IEEE, 2021, pp. 533–539.

[49] S. Venugopal, G. W. Sathianesan, and R. Rengaswamy, "Cyber forensic framework for big data analytics using sunflower jaya optimization-based deep stacked autoencoder," *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, p. e2892, 2021.

[50] P. Nimbalkar and D. Kshirsagar, "Feature selection for intrusion detection system in internet-of-things (iot)," *ICT Express*, vol. 7, no. 2, pp. 177–181, 2021.

[51] P. Jithu, J. Shareena, A. Ramdas, and A. Haripriya, "Intrusion detection system for iot botnet attacks using deep learning," *SN Computer Science*, vol. 2, no. 3, pp. 1–8, 2021.

[52] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, "Towards a deep learning-driven intrusion detection approach for internet of things," *Computer Networks*, vol. 186, p. 107784, 2021.

[53] W. L. Costa, M. M. Silveira, T. de Araujo, and R. L. Gomes, "Improving ddos detection in iot networks through analysis of network traffic characteristics," in *2020 IEEE Latin-American Conference on Communications (LATINCOM)*. IEEE, 2020, pp. 1–6.

[54] M. Ge, X. Fu, N. Syed, Z. Baig, G. Teo, and A. Robles-Kelly, "Deep learning-based intrusion detection for iot networks," in *2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2019, pp. 256–25 609.

[55] A. Demirpolat, A. K. Sarica, and P. Angin, "Protédge: A few-shot ensemble learning approach to software-defined networking-assisted edge security," *Transactions on Emerging Telecommunications Technologies*, p. e4138, 2020.

[56] Y. N. Soe, Y. Feng, P. I. Santosa, R. Hartanto, and K. Sakurai, "Towards a lightweight detection system for cyber attacks in the iot environment using corresponding features," *Electronics*, vol. 9, no. 1, p. 144, 2020.

[57] T. Gao, X. Han, Z. Liu, and M. Sun, "Hybrid attention-based prototypical networks for noisy few-shot relation classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 6407–6414.

[58] R. A. Bauder and T. M. Khoshgoftaar, "The detection of medicare fraud using machine learning methods with excluded provider labels," in *The Thirty-First International Flairs Conference*, 2018.

[59] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[60] R. A. Bauder, T. M. Khoshgoftaar, A. Richter, and M. Herland, "Predicting medical provider specialties to detect anomalous insurance claims," in *2016 IEEE 28th international conference on tools with artificial intelligence (ICTAI)*. IEEE, 2016, pp. 784–790.

[61] Z. Salekshahrezaee, J. L. Leevy, and T. M. Khoshgoftaar, "A reconstruction error-based framework for label noise detection," *Journal of Big Data*, vol. 8, no. 1, pp. 1–16, 2021.

[62] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Computing Surveys (CSUR)*, vol. 53, no. 3, pp. 1–34, 2020.

[63] N. Moustafa and J. 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)*. IEEE, 2015, pp. 1–6.

[64] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[65] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[66] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[67] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019.

[68] M. A. Hall, "Correlation-based feature selection for machine learning," 1999.

[69] J. L. Leevy, J. Hancock, R. Zuech, and T. M. Khoshgoftaar, "Detecting cybersecurity attacks across different network features and learners," *Journal of Big Data*, vol. 8, no. 1, pp. 1–29, 2021.

[70] V. M. Herrera, T. M. Khoshgoftaar, F. Villanustre, and B. Furht, "Random forest implementation and optimization for big data analytics on lexisnexis's high performance computing cluster platform," *Journal of Big Data*, vol. 6, no. 1, pp. 1–36, 2019.

[71] N. Seliya and T. M. Khoshgoftaar, "The use of decision trees for cost-sensitive classification: an empirical study in software quality prediction," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 5, pp. 448–459, 2011.

[72] C. Manapragada, G. I. Webb, and M. Salehi, "Extremely fast decision tree," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1953–1962.

[73] M. Sumner, E. Frank, and M. Hall, "Speeding up logistic model tree induction," in *European conference on principles of data mining and knowledge discovery*. Springer, 2005, pp. 675–683.

APPENDIX

TABLE III
FEATURES, DESCRIPTIONS, AND SETS

Feature	Description	Full Set	5%	10-Best
pkSeqID	Row Identifier	YES	YES	YES[a]
stime	Record start time	YES	YES	NO
flgs	Flow state flags seen in transactions	YES	YES	NO
flgs_number	Numerical representation of feature flags	NO	YES	NO
proto	Textual representation of transaction protocol...	YES	YES	YES[a]
proto_number	Numerical representation of feature proto	NO	YES	NO
saddr	Source IP address	YES	YES	YES[a]
sport	Source port number	YES	YES	YES[a]
daddr	Destination IP address	YES	YES	YES[a]
dport	Destination port number	YES	YES	YES[a]
pkts	Total count of packets in transaction	YES	YES	NO
bytes	Total number of bytes in transaction	YES	YES	NO
state	Transaction state	YES	YES	NO
state_number	Numerical representation of feature state	NO	YES	NO
ltime	Record last time	YES	YES	NO
seq	Argus sequence number	YES	YES	YES
dur	Record total duration	YES	YES	NO
mean	Average duration of aggregated records	YES	YES	YES
stddev	Standard deviation of aggregated records	YES	YES	YES
sum	Total duration of aggregated records	YES	YES	NO
min	Minimum duration of aggregated records	YES	YES	YES
max	Maximum duration of aggregated records	YES	YES	YES
spkts	Source-to-destination packet count	YES	YES	NO
dpkts	Destination-to-source packet count	YES	YES	NO
sbytes	Source-to-destination byte count	YES	YES	NO
dbytes	Destination-to-source byte count	YES	YES	NO
rate	Total packets per second in transaction	YES	YES	NO
srate	Source-to-destination packets per second	YES	YES	YES
drate	Destination-to-source packets per second	YES	YES	YES
attack[b]	Class label: 0 for Normal traffic, 1 for Attac...	YES	YES	YES
category[b]	Traffic category	YES	YES	YES
subcategory[b]	Traffic subcategory	YES	YES	YES
TnBPSrcIP	Total Number of bytes per source IP	NO	YES	NO
TnBPDstIP	Total Number of bytes per Destination IP	NO	YES	NO
TnP_PSrcIP	Total Number of packets per source IP	NO	YES	NO
TnP_PDstIP	Total Number of packets per Destination IP	NO	YES	NO
TnP_PerProto	Total Number of packets per protocol	NO	YES	NO
TnP_Per_Dport	Total Number of packets per dport	NO	YES	NO
AR_P_Proto_P_SrcIP	Average rate per protocol per Source IP (calcu...	NO	YES	NO
AR_P_Proto_P_DstIP	Average rate per protocol per Destination IP	NO	YES	NO
N_IN_Conn_P_SrcIP	Number of inbound connections per source IP	NO	YES	YES
N_IN_Conn_P_DstIP	Number of inbound connections per destination IP	NO	YES	YES
AR_P_Proto_P_Sport	Average rate per protocol per sport	NO	YES	NO
AR_P_Proto_P_Dport	Average rate per protocol per dport	NO	YES	NO
Pkts_P_State_P_Protocol_P_DestIP	Number of packets grouped by state of flows an...	NO	YES	NO
Pkts_P_State_P_Protocol_P_SrcIP	Number of packets grouped by state of flows an...	NO	YES	NO

[a] Included for network session identification only
[b] Dependent feature