A Partial Approach to Intrusion Detection

John Sheppard(✉)

Waterford Institute of Technology, Waterford, Ireland
jsheppard@wit.ie

Abstract. The need for intrusion detection continues to grow with the advancement of new and emerging devices, the increase in the vectors of attack these bring, and their computational limitations. This work examines the suitability of a traditional data mining approach often overlooked in intrusion detection, partial decision trees, on the recent CICIDS 2017 dataset. The approach was evaluated against recent deep learning results and shows that the partial decision tree outperformed these deep learning techniques for the detection of DDoS and Portscan attacks. Further analysis of the complete dataset has been performed using this partial technique. The creation of a reduced feature version of the dataset is proposed using PCA and is evaluated using a partial decision tree. It shows that a ten feature version of the dataset can produce a detection rate of 99.4% across the twelve classes, with a 77% reduction in training time.

Keywords: IDS · Data mining · Partial decision trees · CICIDS · PCA

1 Introduction

The critical importance of Intrusion Detection Systems (IDS) and the suitability of anomaly detection classification models are highlighted in [1] and [2] and reinforced in recent surveys conducted by [3–6], and [7]. Recurring issues associated with intrusion detection include the problem of high dimensionality associated with intrusion detection datasets [8–10], along with the application and evaluation of data mining algorithms for intrusion detection [8,11].

Intrusion Detection is founded upon the belief that the actions of an intruder deviate from those of a normal user in a measurable manner. The measure of the deviation, however, may not be clear and there may exist some overlap between the two behavioural styles. An IDS is a system used for detecting such intrusions. An IDS attempts to detect illegitimate users gaining access to a network or system, and legitimate users misusing resources. In [12] Denning proposed four reasons for the development of such systems. Even with the advancement of technology these motivating factors are still valid. They are:

S. Goel et al. (Eds.): ICDF2C 2020, LNICST 351, pp. 78–97, 2021.
https://doi.org/10.1007/978-3-030-68734-2_5

1. the majority of Information Technology (IT) systems suffer from security vulnerabilities; it is not always feasible to identify and fix theses issues due to technological or financial constraints;
2. where vulnerabilities have been identified in these systems it is not easy to replace these systems due to either financial constraints or due to a trade-off in usability;
3. it is not possible to build a completely secure system especially considering the two very independent and non-integrated worlds of software development and security where it has been tradition to develop and release software as quickly as possible and patch problems later;
4. even if systems are secured externally, attacks can occur internally through avenues such as privilege escalation or misuse.

To tackle these problems there has been a wealth of work based on the application of Artificial Intelligence (AI) techniques to fabricated datasets. Models produced by researchers in the past have been weakened through the analysis of the datasets on which these models were based. In 2018 the CICIDS 2017 dataset was released, and an attempt was made to address the issues that had been discovered in earlier datasets. This work focuses on the performance of traditional data mining, and deep learning techniques on this dataset in terms of accuracy, false alarm rates and speed.

In this work a Partial Decision Tree approach has been used on the CICIDS 2017 dataset for the first time. It has been chosen for its speed and accuracy. Decision tree techniques have been commonly used on various different IDS datasets. Partial Decision Trees have been are applied here as they tend to produce higher accuracy rates than separate and conquer techniques, while being faster than divide and conquer approaches.

1.1 Contribution of This Work

The contribution of this work can be summarized as follows:

- An evaluation of a partial decision tree approach to the complete CICIDS 2017 dataset, and to a Principal Component Analysis (PCA) reduced feature version of this dataset.
- An evaluation of a partial decision tree versus a deep learning approach for the detection of Denial of Service (DoS), Distribute Denial of Service (DDoS) and Probing attacks using the CICIDS 2017 dataset.

1.2 Structure of This Work

Section 2 presents the related research in the area of IDS. Section 3 details the dataset used. Section 4 presents the methodology used. Section 5 presents the results of the approach taken. Section 6 discusses the findings. Section 7 summarises the conclusions of the paper, and identifies future work that needs to be addressed in the area.

2 Related Work

Intrusion detection systems can take several forms. In accordance with the National Institute for Standards and Technology (NIST) [13] these are Network Intrusion Detection Systems (NIDS), Host Intrusion Detection Systems (HIDS), Network Baseline Analysers (NBA) and Wireless Intrusion Detection Systems (WIDS). These systems are characterised by different monitoring and analysing approaches. These approaches can be described in terms of a generic process model. This model has three fundamental components, (i) information sources, (ii) analysis engine (iii) response [14].

For research purposes, the community has worked on the development of different datasets to replicate the information source phase, and to give researchers a benchmark for the analysis phase. A review of eleven of the most popular datasets used in intrusion detection highlighted eleven shortcomings with these datasets [15]. The datasets reviewed were:

- Darpa (Lincoln Laboratory 1998, 1999)
- KDD 99 (University of California, Irvine 1998, 99)
- Defcon Dataset 2000
- CAIDA 2002/2016
- LBNL (Lawrence Berkeley National Laboratory and ICSI - 2004/2005)
- CDX (United States Military Academy 2009)
- Kyoto (Kyoto University - 2009)
- Twente (University of Twente - 2009)
- UMASS (University of Massachusetts - 2011)
- ISCX2012 (University of New Brunswick - 2012)
- ADFA (University of New South Wales - 2013)

Following the review of these datasets, a new dataset was designed taking account of these shortcomings. The release of the CICIDS-17 dataset by the Canadian Institute for Cybersecurity [16] gave the research community a new dataset to work with. The dataset was produced over five days during July 2017. This dataset contained B profile data used to produce benign activity of 25 users across HTTP, HTTPS, FTP, SSH, and email protocols. The first day of data did not contain any attack data. The dataset also contained attacks conducted over the other four days. The implemented attacks include Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS. The dataset was made available as .pcap files and as .CSV files for research. Eighty features were constructed from the data. In 2018 the CSE-CICIDS 2018 dataset was released. It has been reviewed and compared with the 2017 version [17]. They found improvements in the sample sizes for some attacks, in particular for Botnet and Infiltration attacks, but that the number of Web attack samples were small at 928 instances. This paper focuses primarily on the detection of Portscan, DoS and DDoS attacks and so the CICIDS 2017 dataset has been used.

This dataset has been analysed using Support Vector Machines (SVM) and deep learning techniques for the detection of port scans by [18]. On a subset

of the data consisting of 286467 records, where 127537 were benign and 158930 were port scan attempts. The dataset was split with 67% used for training and 33% used for testing. This gave 191684 samples for training and 94412 for testing. The deep learning model was trained based on 30 passes. It was found that Port Scans could be detected with a deep learning machine with an accuracy of 97.8% while SVMs could only achieve 69.8% accuracy. On a separate version of the dataset containing 80 features, and 26,167 DDoS and 26,805 benign examples, [19] achieved accuracy rates of 60.69% for SVMs and 99.0% for a decision tree approach. However, the decision tree algorithm used is not specified.

Shallow and deep learning methods are an advancement on Artificial Neural Network (ANN) architectures. An ANN which consists of one or two layers is considered to be shallow learning. Deep learning consists of several hidden layers [20]. Deep learning techniques can be categorised into generative and discriminative architectures [21]. Deep learning techniques require larger amounts of data than traditional machine learning algorithms. They also require longer training times and higher performance machines. However deep learning does tend to have the advantage of higher accuracy rates and eliminate the need for the feature extraction phase by identifying high-level features in an incremental manner. Deep learning also tends to solve problems end-to-end.

Generative models depict independence/dependence for distribution graphically. Generative models can be classed as Recurrent Neural Network (RNN), Deep Auto-Encoders, Deep Boltzmann Machines (DBM) and Deep Believe Networks (DBN). Discriminative architectures, such as those used in Convolutional Neural Networks (CNNs) and some RNNs, use discriminative power for classification [31].

To date, much of the literature associated with deep learning in intrusion detection has focused on deep learning techniques evaluated against the KDDCup'99 and the NSL-KDD Dataset. Example of such work include [22–24] and [25]. However these datasets have been critiqued for issues such as a lack of real world data and for suffering from bias as was reported by [26].

A comparative study of traditional machine learning algorithms and deep learning algorithms, was conducted across several IDS datasets including KDDCup'99, NSL-KDD, UNSW-NB15 and CICIDS 2017 [27]. The CICIDS 2017 training dataset consisted of 93,500 instances, while the testing dataset contained 28,481. The approaches of Decision Tree, Ada Boost (AB) and Random Forest (RF) classifiers performed better than the other classifiers namely linear regression, Naive Bayes, K-Nearest Neighbour and SVMs. The performance of Decision Trees (DT), AB and RF classifiers remained in the same range across different datasets. The performance of Logistic Regresion (LR), NB, KNN, and SVM-rbf are varied across different datasets. They state that the DT, AB and RF classifiers can detect new attacks. In the binary classification experiment labelled as attack versus normal, the decision tree produced an accuracy rate of 92.9% on the KDDCup 99 dataset, 93% on the NSL-KDD dataset and 93.5% on the CICIDS 2017 dataset. The SVM technique produced accuracy rates 87.7%, 83.7% and 79.9% for the three datasets respectively. Depending on the number

of layers used, Deep Neural Networks produced accuracy rates between 92.7% and 93% for the KDDCup data, between 78.9% and 80.1% for the NSLKDD data and between 93.1% and 96.3% for the CICIDS data.

CICIDS 2017 DDoS attacks were examined using Random Forests for feature reduction followed by the use of a deep learning MLP [28]. The dataset consisted of 225,746 instances and an accuracy rate of 91% was achieved using all 80 features and the MLP approach. Using the Random Forest feature reduction technique, the dataset was reduced to the 10 most important features. The retrained MLP model could then achieve 89% accuracy. Training and testing times for the two models were not presented.

The CICDS 2018 dataset was examined using decision trees, Gaussian Naive Bayes, random forest, KNN, SVM and LTSM+AM. These produced accuracy rates of 92.8%, 55.4%, 94.2%, 94.2%, 74.7%, 96.2% respectively. LTSM+AM was also compared with MLP, producing 90.5%, and with LTSM without AM, producing 93.3% [17].

An ANN approach was compared with a Random Forest classifier in [45]. The ANN had an average accuracy score of 96.53% while the random Forest had a score of 96.24.

In [43] a deep fully connected feed forward neural network was employed on the full CICIDS 2017 dataset. This deep learning approach achieved an accuracy rate of 96.77%. When IP addresses were included as a feature in the dataset an accuracy of 99.93% was achieved. Manimurugan et al. applied a range of deep learning techniques to the CICDIS 2017 dataset. Using a Deep Belief Neural network they achieved an accuracy of 96.67% on the DoS and DDoS data and an accuracy rate of 97.71% for Portscan attacks. SVM, RNN, Spiking Neural Network (SNN) and FNN were also employed but found to perform poorer than the DBN [44].

Recently there has been a focus in terms of the application and deployment of IDS to alternate environment with different computational powers. Environments investigated include Internet of Things (IoT), Cloud-based and Fog-based [3]. The need for improved AI techniques in these areas in expected to grow in the future.

IoT IDS suffer from high false alarm rates due to overlapping and suspicious instances [34,35]. Sources of data with IoT IDS can include telemetry data of sensors and actuators connected to the internet. There is a need for further research into the development of new post-processing techniques for IoT networks for correlating NIDS alerts, reducing false alarm rates and for visualisation of network data [36]. [37] surveyed the challenges of security in IoT and presented a comprehensive review of current anomaly-based IoT IDSs. [38] proposed a lightweight machine learning NIDS for IoT environments using a combination of Fuzzy C-means Clustering (FCM) and PCA, while [39] presents a NIDS with low computational and resource requirements using decision trees. A data mining approach for an IoT NIDS using PCA and Suppressed Fuzzy Clustering (SFC) techniques proved to be well suited to high dimensional spaces producing high levels of accuracy [40].

Cloud IDS are used by organisations that send data and processing tasks to public cloud services. Existing NIDS are not capable of detecting internal malicious activities. The detection of internal malicious activities is a challenging task due to their complexity and the remotely located modules [41]. Mobile edge IDS, or Fog-based IDS, suffer from decentralised and distributed issues such as synchronisation, provider integration and VM access to host and context information [42].

3 Dataset

The Canadian Institute for Cybersecurity (CIC) has been developing more recent datasets with up to date attacks. For this work the CICIDS 2017 dataset was chosen for use based on the shortcomings of other datasets in the area. In [15] eleven of the most cited and studied datasets were evaluated, and eleven shortcomings of these datasets were identified. The CICIDS 2017 and CICIDS 2018 datasets were developed taking these considerations into account to improve the quality of the dataset. These shortcomings found that:

1. To represent the real world a completely configured network is needed.
2. A realistic looking mix of pseudo-realistic, or synthetic traffic in a dataset.
3. Attention should be given to whether the dataset is labelled, partial labelled or unlabelled.
4. All network interactions are needed, such as interactions between internal networks.
5. All traffic should be captured and should remain in the dataset. They found issues with other datasets where non-functional or unlabelled data was removed.
6. Normal and anomalous data needs to be present.
7. There is a need to be able to analyse and test IDSs against up to date attacks in an off-line environment.
8. Payloads are often removed due to privacy concerns. However this prevents research in to techniques such as deep packet inspection.
9. There is a need for datasets that use a single source, and datasets that use multiple combined sources of data such as network traffic and operating system logs.
10. One of the biggest issues in the creation of an IDS dataset is feature extraction and construction.
11. Most IDS datasets suffer from incomplete documentation.

Taking these considerations into account, in January 2018 CIC released CIC-IDS 2017 [16]. The dataset was developed over five days during July 2017. This dataset contained B profile data used to produce benign activity of 25 users across HTTP, HTTPS, FTP, SSH, and email protocols. The first day of data did not contain any attack data. The dataset also contained attacks conducted over the other four days. The attacks were categorised seven major groups based

on the 2016 McAfee report, Browser-based, Brute force, DoS, Scan or enumeration, Backdoors, DNS, and other attacks (e.g., Heartbleed, Shellshock, and Apple SSL library bug). The dataset was made available as .pcap files and as .CSV files for research. Eighty features were constructed from the data using the CICFlowMeter-V3.

CICIDS 2017 was followed by CSE-CICIDS 2018, an updated version with more instances. [17] presented the increase in traffic flows as seen in Table 1. As the download size of the CSE-CICIDS 2018 dataset is over 220 Gb in size it was decided to focus on the CICIDS 2017 dataset for this research.

Table 1. Statistics of redundant records in the testing data

	Normal	DDOS	PortScan	BOT	Inf	Web Attack	BF	DOS
CIC-IDS-2017	1743179	128027	158930	1966	36	2180	13835	252661
CSE-CIC-IDS-2018	6112151	687742	–	286191	161934	928	380949	654301

4 Methodology

The release of the CICIDS-17 dataset by the Canadian Institute for Cybersecurity [16] gave the research community a new dataset to work with. The experiments that were conducted were,

1. An evaluation of a partial decision tree approach for probing attack detection, using the data employed by [18] to test SVM and Deep Learning techniques. This dataset consisted of benign and probing activity.
2. An evaluation of a partial decision tree approach, using data as used for SVM and Deep Learning techniques in [28]. This dataset consisted of benign and DDoS activity.
3. An evaluation of a partial decision tree approach for the detection of DoS and DDoS attacks.
4. An evaluation of a partial decision tree approach, using the full dataset gathered over the 5 days, and the 12 classes, benign, PortScan, FTP-Patator, SSH-Patator, DoS, DDoS, Heartbleed, Web-Attack-Brute-Force, Web-Attack-XSS, Infiltration, Bot, Web-Attack-Sql-Injection.
5. An evaluation of a partial decision tree approach to 10 feature dataset created by applying Principal Component Analysis (PCA) to the dataset.

Each of these experiments was conducted in the Weka 3-9-1 environment using the default PART parameters. For the Portscan data, the training and testing data were split 67% to 33% to consistent with [18]. For the DDoS and full dataset 10 fold cross validation was used. The results of experiments on the CICIDS 2017 dataset are presented and discussed in Sect. 5.

Classification. One of the most common forms of data mining is classification. Classification differs from prediction in that classification is the forecasting of a discrete or categorical value whereas prediction forecasts a continuous value [29].

Given a database $D = \{t1, t2, ..., tn\}$ of tuples and a set of classes $C = \{C1, C2, ..., Cn\}$, the classification problem is to define a mapping $f: D \rightarrow C$ where each ti is assigned to one class. A class, Cj, contains precisely those tuples mapped to it; that is $Cj = ti|f(ti) = Cj, 1 \leq i \leq n$ and $ti \epsilon D$ [30].

Classification occurs in two phases

1. A model is created on a specific dataset by evaluating a training set in a supervised learning mode. This phase takes as input a training dataset and produces as output a definition of a model which classifies the training set as accurately as possible.
2. Apply the model created in stage one to a test dataset.

Classification algorithms are categorised as Statistical, Distance, Decision Tree, Neural Network and Rule-based.

Decision Tree Techniques. A Decision Tree is constructed from nodes, representing questions, and arcs that represent the possible answers to those questions. The first node is known as the root, and the set of internal nodes, based on the answers, leads to the solution at the end leaf node [32].

A Decision Tree Model consists of three parts;

- A decision tree consisting of edges or branches, and nodes or leaves
- An algorithm for creating the tree
- An algorithm that applies the tree to data to solve the problem being considered

[30] defines a decision tree as:

Given a database $D = \{t_1, ..., t_n\}$ where $t_i = <t_{i1}, ..., t_{ih}>$ and the database schema contains the following attributes $A_1, A_2, ..., A_h$. Also given is a set of classes $C_1, ..., C_m$. A decision tree (DT) or classification tree is a tree associated with D that has the following properties:

- Each internal node is labelled with attribute, A_i
- Each arc is labelled with a predicate that can be applied to the attribute associated with the parent
- Each leaf node is labelled with a class, C_j

Decision trees are easy to use and produce output that can easily be understood and hence easily translated into rulesets. DTs scale well with large datasets and can handle large numbers of attributes [30]. Continuous data needs to be divided into categories though in order to be used. Missing data can prove troublesome as a decision cannot be made as to which branch in the tree should be followed. DTs can also suffer from overfitting though this can be addressed by pruning. Correlations amongst attributes are ignored by DTs.

Attributes are represented as nodes in a DT structure and are referred to as splitting attributes. The arcs which come from these nodes are known as splitting predicates [30]. Algorithms may be differentiated in the way they choose splitting attributes and select their splitting predicates.

The performance of a DT [30] is affected by

- The choice of splitting attribute
- The order of splitting attribute
- The number of splits required over the whole tree
- The structure of the tree in terms of number levels and the degree of branching
- The stopping criteria
- The amount of training data involved
- Any pruning of the tree which may be required

The complexity of the algorithm depends on the product of the number of levels in the tree, and the maximum branching factor at each level. The time complexity to build a tree is $O(hq\ log\ q)$, where h is the number of attributes and q is the size of the training data. The time needed to classify a database of size n is based on the height of the tree. Assuming a height of $O(log\ q)$ this can be calculated as $O(n\ log\ q)$ [30].

The most common decision tree algorithms include C4.5, C5.0 and CART. C4.5 extended ID3. ID3 splits attributes by selecting the attribute with the highest information gain. Entropy is a measure of the amount of uncertainty, surprise or randomness in a dataset. Entropy [30] is defined as

Given probabilities $p_1, p_2, ..., p_s$ where $\sum_{i=1}^{s} p_i = 1$, entropy is defined as

$$H(p_1, p_2, ..., p_s) = \sum_{i=1}^{s} (p_i log(1/p_i)) \tag{1}$$

A split is calculated from the difference in entropies of the original dataset and the weighted sum of the entropies form each of the subdivided datasets. ID3 determines the gain as

$$Gain(D, S) = H(D) - \sum_{i=1}^{s} P(D_i)H(D_i) \tag{2}$$

Where *H(D)* finds the amount of order in database state *D*. When state D is split it produces s new states $S = \{D_1, D_2, ..., D_s\}$.

A hybrid algorithm of J48, Meta Paging, RandomTree, REPTree, AdaBoostM1, DecisionStump and NaiveBayes was applied to a dataset which had been filtered using information gain and the vote algorithm can be seen in [8].

Frank & Witten developed an approach for rule induction that combines both the divide and conquer technique for decision trees with the separate and conquer technique for rule learning in an attempt to overcome their limitations. This method of obtaining rules from partial decision trees avoids global optimisation and produces accurate, compact rule sets. It follows the separate and conquer technique in that it, [32].

1. Builds a rule X
2. Removes the instances covered by rule X
3. Continue to create rules recursively for the remaining instances until none are left

PART can be differentiated from the usual separate and conquer technique in its rule production. It creates a pruned decision tree from the current set of instances, and extract the leaf with the largest coverage as rule. The rest of the tree is then discarded. This approach removes the tendency to overprune associated with the separate and conquer approach [32].

Partial decision tree approaches tend to outperform separate and conquer approaches in terms of accuracy. They are comparable to divide and conquer approaches in terms of accuracy, but are faster than divide and conquer techniques such as C4.5. Partial decision tree approaches combine the simplistic approach of separate and conquer with the accuracy of divide and conquer techniques. The complexity of the partial decision tree approach is $O(a \text{ x } n \ log \ n)$. On a noise free dataset, partial decision trees are very fast, as there is no need to prune the tree. This means that as the level of noise in the data increase, there is a decrease in the performance speed [33].

5 Results

Metrics. For domains such as intrusion detection a misclassification can be crucial to the security of the network. In a binary case of normal or intrusion, each prediction has four possible outcomes represented by a confusion matrix.

- True Positive TP, class correctly determined as class 1
- True Negative TN, class correctly determined as class 2
- False Positive FP, class incorrectly classified as class 1
- False Negative FN, class incorrectly classified as class 2

The true positive rate is TP divided by the total number of positives which is $TP + FN$; the false positive rate FP divided by the total number of negatives, $FP + TN$. The overall success rate is the number of successful classifications divided by the total number of classifications as shown below. The error rate is one minus the overall success rate [32].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

Where there are more than two classes to be present, a two dimensional confusion matrix will be created with a row and a column for each class. Correct classifications are indicated by high numbers running down the diagonal diagonal. Any values not present on the diagonal are counts of instances that have been miscalssified. A confusion matrix is often represented using an Operating Characteristic (OC) curve that plots the percentage of false positive on the x axis against the percentage of true positives on the y axis [32].

Two other metrics commonly used in information retrieval, are precision and recall [32]. These are plotted against each other to produce recall-precision curves.

$$recall = \frac{numberofrelevantdocumentsretrieved}{totalnumberofrelevantdocuments} \tag{4}$$

$$precision = \frac{numberofrelevantdocumentsretrieved}{totalnumberofdocumentsretrieved} \tag{5}$$

In general a high precision and a high recall are desirable. However a high precision leads to a low recall and vise versa. A trade-off between these is the harmonic mean, or F-measure. This can be defined as

$$F = \frac{2XrecallXprecision}{recall + precision} = \frac{2xTP}{2x(TP + FP + FN)} \tag{6}$$

A Partial Approach to the Portscan Data. Table 2 presents the confusion matrix of a partial decision tree approach to the PortScan data from the CICIDS dataset. As with [18], the data was split as 67% training data and 33% testing data. It took 71.93 s to build the model and 0.17 s to test the model on the split data. This produced an accuracy rate of 99.98% and a ruleset of fifteen rules. This model outperformed the SVM and deep learning methods proposed in [18] as can be seen in Table 3.

Table 2. Confusion matrix: PART CICIDS PortScan data

Classified as ->	Normal	Attack
Normal	43387	8
Attack	11	53993

Table 3. Confusion matrix: Part comparison with [18] CICIDS PortScan data

Method	Accuracy	Precision	Recall	F1 Score
PART	99.98	1.0	1.0	1.0
Deep learning	97.80	0.99	0.99	0.99
SVM	69.79	0.80	0.70	0.65

A Partial Approach to the DDoS Data. The second version of the dataset used was the DDoS dataset. It consisted of 225,746 instances and took 63.72 s to build the model. It produced an accuracy rate of 99.99% over 14 rules. Ten of the normal instances were misclassified as being attacks and seven attacks were missed and classified as being normal. [28] analysed the same version of the dataset using an MLP approach and achieved an accuracy rate of 91% as can be seen in Table 7 (Tables 4 and 5).

Table 4. Confusion matrix: PART CICIDS DDoS data

Classified as ->	Normal	Attack
Normal	97708	10
Attack	7	128020

Table 5. Part comparison with [28] CICIDS DDoS data

Method	Accuracy
PART	99.98
MLP	91

A Partial Approach to the DoS/DDoS Data. The third version of the data used consisted of all DoS and DDoS data. This totaled 918,448 instances and produced an accuracy rate of 99.97%. The number of false alarms outweighed the number of missed attacks as can be seen in the confusion matrix in Table 6. The model took 890 s to build and produced 77 rules.

Table 6. Confusion matrix: PART CICIDS DDoS data

Classified as ->	Normal	Attack
Normal	537584	165
Attack	71	380628

A comparison with the results of [44] can be seen in Table 7. The Partial decision tree technique outperformed the Deep Learning techniques that had been employed.

Table 7. Part compared with deep learning techniques [44] for CICIDS DoS/DDoS data

Method	Accuracy	Precision	Recall	F1 score
PART	99.97%	1.000	1.000	1.00
DBN [44]	96.67%	0.952	0.973	0.97
SVM [44]	95.55%	0.943	0.962	0.95
RNN [44]	94.40%	0.939	0.956	0.94
SNN [44]	93.30%	0.920	0.943	0.93
FNN [44]	92.25%	0.911	0.911	0.92

A Partial Approach to the Full DataSet. The results of the third analysis can be seen in Table 8. This table presents the breakdown of the classification of each attack in the dataset. The model produced an accuracy rate of 99.98% and took 7812.19 s to build. The metrics associated with this experiment can be seen in Table 9.

Table 8. Confusion matrix: PART CICIDS full dataset

Classified as ->	A	B	C	D	E	F	G	H	I	J	K	L
(A) normal	2271622	989	6	4	381	17	1	8	9	2	56	2
(B) PortScan	17	158869	2	0	38	0	0	4	0	0	0	0
(C) FTP-Patator	3	0	7934	0	0	0	0	1	0	0	0	0
(D) SSH-Patator	7	0	1	5889	0	0	0	0	0	0	0	0
(E) DoS	65	12	0	0	252574	3	0	5	0	0	0	2
(F) DDoS	19	0	0	0	3	128005	0	0	0	0	0	0
(G) Heartbleed	1	0	0	0	0	0	10	0	0	0	0	0
(H) Web-Attack-Brute-Force	14	2	0	0	1	0	0	1468	22	0	0	0
(I) Web-Attack-XSS	6	3	1	0	4	0	0	582	56	0	0	0
(J) Infiltration	12	0	0	0	0	0	0	0	0	24	0	0
(K) Bot	544	0	0	0	0	0	0	0	0	0	1422	0
(L) Web-Attack-Sql-Injection	1	0	0	0	4	0	0	2	2	0	0	12

A Partial Approach to the PCA Reduced DataSet. The results of the fourth experiment are presented in Table 10. This table shows the breakdown of the classification of each attack in the dataset. The model produced an accuracy rate of 99.4% and took 1767.95 s to build. This dataset was reduced to 10 features using PCA in Weka. The variance used to produced the dataset was 0.95 and the 10 most important new features produced from this were used to create the new dataset. The detailed metrics associated with this partial decision tree experiment can be seen in Table 11.

Table 9. Detailed accuracy rates for the CICIDS 2017 full dataset

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC area	PRC area
(A) normal	0.999	0.001	1.000	0.999	1.000	0.998	1.000	1.000
(B) PortScan	1.000	0.000	0.994	1.000	0.997	0.996	1.000	0.997
(C) FTP-Patator	0.999	0.000	0.999	0.999	0.999	0.999	1.000	0.999
(D) SSH-Patator	0.999	0.000	0.999	0.999	0.999	0.999	1.000	0.999
(E) DoS	1.000	0.000	0.998	1.000	0.999	0.999	1.000	0.999
(F) DDoS	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000
(G) Heartbleed	0.909	0.000	0.909	0.909	0.909	0.909	0.955	0.826
(H) Web-Attack-Brute-Force	0.974	0.000	0.709	0.974	0.821	0.831	0.991	0.825
(I) Web-Attack-XSS	0.086	0.000	0.629	0.086	0.151	0.232	0.984	0.437
(J) Infiltration	0.667	0.000	0.923	0.667	0.774	0.784	0.912	0.638
(K) Bot	0.723	0.000	0.962	0.723	0.826	0.834	0.997	0.927
(L) Web-Attack-Sql-Injection	0.571	0.000	0.750	0.571	0.649	0.655	0.855	0.545
Weighted Avg	0.999	0.001	0.999	0.999	0.999	0.997	1.000	0.999

Table 12 present a comparison of the Full Dataset evaluation, the PCA dataset evaluation and the results produced by [43]. It can be seen that both Partial decision tree models out perform the Deep Neural Network.

6 Discussion

The partial decision tree approach was compared to deep learning approaches for the DDoS and Probing data of the dataset. It outperformed both of these in accuracy. [27] achieved an accuracy rate of 79.9% using SVM and accuracy rates between 93.1% and 96.3% using Deep Neural Networks compared to accuracy scores of 99.98%. The dataset used consisted of 93,500 instances, while the testing dataset contained 28,481, however the authors do not describe how this data was sampled. The partial decision tree method was also evaluated against the work of [18]. This focused on the evaluation of the portscan campaign that had been conducted. It again outperformed the MLP apporach which had an accuracy of 91% in comparison to our model which again scored 99.98%.

Table 10. Confusion matrix: PART CICIDS reduced feature dataset

Classified as ->	A	B	C	D	E	F	G	H	I	J	K	L
(A) normal	2269846	1145	8	80	1393	362	0	113	15	5	130	0
(B) PortScan	145	158733	0	1	37	12	0	2	0	0	0	0
(C) FTP-Patator	10	0	7920	5	2	0	0	1	0	0	0	0
(D) SSH-Patator	73	3	0	5817	1	0	0	3	0	0	0	0
(E) DoS	11266	10	1	1	240992	386	1	4	0	0	0	0
(F) DDoS	334	2	0	0	587	127103	0	0	0	0	1	0
(G) Heartbleed	2	0	0	0	1	1	7	0	0	0	0	0
(H) Web-Attack-Brute-Force	82	3	1	25	1	0	0	1303	92	0	0	0
(I) Web-Attack-XSS	44	2	0	0	2	0	0	518	86	0	0	0
(J) Infiltration	27	0	0	0	1	0	0	0	0	8	0	0
(K) Bot	1064	0	0	0	1	0	0	0	0	0	901	0
(L) Web-Attack-Sql-Injection	13	0	0	4	1	0	0	3	0	0	0	0

Table 11. Detailed accuracy rates for the CICIDS 2017 reduced feature dataset

Class	TP rate	FP rate	Precision	Recall	F-Measure	MCC	ROC area	PRC area
(A) normal	0.999	0.023	0.994	0.999	0.996	0.982	0.999	0.999
(B) PortScan	0.999	0.000	0.993	0.999	0.996	0.995	1.000	0.997
(C) FTP-Patator	0.998	0.000	0.999	0.998	0.998	0.998	0.999	0.997
(D) SSH-Patator	0.986	0.000	0.980	0.986	0.983	0.983	0.999	0.983
(E) DoS	0.954	0.001	0.992	0.954	0.972	0.970	0.999	0.994
(F) DDoS	0.993	0.000	0.994	0.993	0.993	0.993	0.999	0.995
(G) Heartbleed	0.636	0.000	0.875	0.636	0.737	0.746	0.864	0.564
(H) Web-Attack-Brute-Force	0.865	0.000	0.669	0.865	0.754	0.761	0.969	0.739
(I) Web-Attack-XSS	0.132	0.000	0.446	0.132	0.204	0.242	0.954	0.386
(J) Infiltration	0.222	0.000	0.615	0.222	0.327	0.370	0.806	0.146
(K) Bot	0.458	0.000	0.873	0.458	0.601	0.632	0.987	0.686
(L) Web-Attack-Sql-Injection	0.000	0.000	0.000	0.000	0.000	0.000	0.717	0.020
Weighted Avg	0.994	0.019	0.994	0.994	0.993	0.981	0.999	0.998

Table 12. Full dataset comparison with [43]

Method	Accuracy	Precision	Recall	F1 score
PART	99.98%	0.999	0.999	0.999
PART (PCA)	99.4%	0.994	0.994	0.993
DNN [43]	96.77%	0.978	0.968	0.973

In experiment 3 DoS and DDoS data were examined together, and a model was built for DoS/DDoS attack detection. The inclusion of the DoS attacks led to a very slight decline in accuracy over the model produced in the second experiment, and the model now missed 71 attacks which was a further 64 above the DDoS model. 165 patterns of normal usage were misclassified compared to 10 with the DDoS model. [44] evaluated Deep Belief, Recurrent Neural Network, Spike Neural Network and Fullforward Neural Network for the same purpose and found Deep Belief Neural Networks to achieve the highest accuracy at a rate of 96.67%.

The full dataset was analysed using the partial decision tree and all 12 classes. The class used were Normal, PortScan, FTP-Patator, SSH-Patator, DoS, DDoS, Heartbleed, Web-Attack-Brute-Force, Web-Attack-XSS, Infiltration, Bot, Web-Attack-Sql-Injection. The results of this can be seen in Table 8. The model was built in 7812.19 s and achieved an accuracy rate of 99.899%. It contained 235 rules. Web-Attack-Brute-Force, Web-Attack-XSS, Bot and Web-Attack-Sql-Injection performed worst all achieving detection rates of less than 90%. Web-Attack-XSS had the lowest detection rate at 0.086%, however 87.4% of the instances were detected as a brute force web attack rather than a XSS attack. Only 6 of the 666 XSS instances were missed and classed as normal. The class Bot had a detection rate of 72.3%. 544 of the 1,966 instances were missed and classed as normal behaviour. Of the 1,507 Web-Attack-Brute-Force instances, 22 were classed as Web-Attack-XSS, 2 as portscan and 1 as DoS. 14 were missed

and classed as normal behaviour. Web-Attack-Sql-Injection achieved a detection rate of 57.1%. However there were only 21 instances of this attack in the entire dataset, and only 1 of these was missed and classed as normal. The detection rate (TP), false alarm rates (FP), precision, recall, harmonic mean (F-Measure), Matthews Correlation Coefficient (MCC), ROC and PRC results are presented in Table 9. 1,475 instances of normal behaviour were classified as an attack. This model was evaluated against the model produced by [43]. The PART model produced an accuracy score of 99.98% in comparison to the Deep Neural Network model.

When the dataset was reduced to 10 features and 12 classes the accuracy of the model fell to 99.36% but the time to build the model dropped from 8812.19 s to 1767.95 s. This was a drop of 77% in the time taken with a fall in accuracy of just 0.5%. Except for Web-Attack XSS, there was a drop in the detection rate of each attack. There was an increase in all instances of normal behaviour being misclassified for each attack class. There was also an increase in the number of attacks missed for each class. This model was also found to outperform the depp learning model found in [43].

7 Conclusion

Data mining has become an important tool in the intrusion detection community toolbox. Intrusion detection systems have steadily moved to using the approach of a hybrid of data mining techniques to build intrusion detection systems. These components, or modules, can be used to detect different types of attacks and usually form one of several stages. A well designed IDS should be made up of interchangeable rules modules which can be used with varying architectures. These rules modules should have a shorter life span than the architecture itself and will change as the attacks advance.

The partial decision tree approach was evaluated on the CICIDS 2017 dataset. Five versions of this dataset were used in this work, the Portscan data, the DDoS data, the DoS/DDoS data, the twelve class dataset with all features present, and a reduced version of the dataset with 10 features produced from a PCA analysis of the data. The results were compared with deep learning and SVM approaches found in the literature. In each instance the partial decision tree approach out performed the deep learning and SVM approaches in terms of accuracy. The PART algorithm achieved accuracy rates of 99.98% on the Portscan data, 99.99% on the DDoS data and 99.89% on the entire dataset. The accuracy fell slightly when the dataset was reduced however the time taken was significantly reduced as was the size of the data.

7.1 Future Work

Future work in the area of intrusion detection involves the further evaluation of the features currently available in the dataset, and the investigation of new features that could be created from the raw .pcap files that are also available

to researchers. The evaluation of hybrid approaches between traditional data mining and deep learning techniques also warrants further investigation. Recent work has seen the expansion of deployment architectures to be more specific to domains such as IoT, vehicle, cloud and mobile computing environments. More work is needed in these areas regarding the creation of datasets, the suitability of current analysis and detection methods in these different environments, and the correlation of results from detection engines speaking to one another. Work is also needed in the area of eXplainable AI for network forensics and IDS.

References

1. Du, X., et al.: SoK: exploring the state of the art and the future potential of artificial intelligence in digital forensic investigation. In: Proceedings of the 15th International Conference on Availability, Reliability and Security, ARES 2020. ACM, August 2020. https://doi.org/10.1145/3407023.3407068, ISBN: 9781450388337
2. Nguyen Thi, N., Cao, V.L., Le-Khac, N.-A.: One-class collective anomaly detection based on LSTM-RNNs. In: Hameurlain, A., Küng, J., Wagner, R., Dang, T.K., Thoai, N. (eds.) Transactions on Large-Scale Data- and Knowledge-Centered Systems XXXVI. LNCS, vol. 10720, pp. 73–85. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-56266-6_4
3. Moustafa, N., Hu, J., Slay, J.: A holistic review of network anomaly detection systems: a comprehensive survey. J. Netw. Comput. Appl. **128**, 33–55 (2019)
4. Othman, S., Alsohybe, N., Ba-Alwi, F., Zahar, A.: Survey on intrusion detection system. Int. J. Cyber-Secur. Digital Forensics (IJCSDF) (2018). ISSN: 2305–001
5. Buczak, A., Guven, E.: A survey of data mining and machine learning methods for cybersecurity intrusion detection. IEEE Commun. Surv. Tutor. **18**(2), 1153–1176 (2016). https://doi.org/10.1109/COMST.2015.2494502
6. Ahmed, M., Naser, A., Hu, J.: A survey of network anomaly detection techniques. J. Netw. Comput. Appl. **60**, 19–31 (2016). https://doi.org/10.1016/j.jnca.2015.11.016. ISSN: 1084–8045
7. Modi, U., Jain, A.: A survey of IDS classification using KDD cup 99 dataset in WEKA. Int. J. Sci. Eng. Res. **6**(11), 947–954 (2015). ISSN 2229–5518
8. Aljawarneh, S., Aldwairi, M., Yassein, M.: Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. J. Comput. Sci. **25**, 152–160 (2018). https://doi.org/10.1016/j.jocs.2017.03.006
9. Ambusaidi, M., He, X., Nanda, P., Tan, Z.: Building an intrusion detection system using a filter-based feature selection algorithm. IEEE Trans. Comput. **65**(10), 2986–2998 (2016). https://doi.org/10.1109/TC.2016.2519914
10. Hasan, M., Nasser, S., Ahmad, M., Molla, K.: Feature selection for intrusion detection using random forest. J. Inf. Secur. **7**, 129–140 (2016)
11. Elhag, S., Fernández, A., Alshomrani, S., Herrera, F.: Evolutionary fuzzy systems: a case study for intrusion detection systems. In: Bansal, J.C., Singh, P.K., Pal, N.R. (eds.) Evolutionary and Swarm Intelligence Algorithms. SCI, vol. 779, pp. 169–190. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-91341-4_9
12. Denning, D.: An intrusion-detection model. In IEEE Trans. Softw. Eng., Piscataway, NJ, USA, vol. 13, pp. 222–232. IEEE Press, February 1987. https://doi.org/10.1109/TSE.1987.232894
13. Scarfone, K., Mell, P.: 800–94 rev-1. NIST Guide to Intrusion Detection and Prevention Systems (IDPS) Revision, vol. 1 (2012)

14. Stolfo, S., Lee, W., Chan, P., Fan, W., Eskin, E.: Data mining-based intrusion detectors: an overview of the Columbia ids project. ACM SIGMOD Rec. **30**(4), 5–14 (2001)
15. Gharib, A., Sharafaldin, I., Lashkari, A., Ghorbani, A.: An evaluation framework for intrusion detection dataset. In: 2016 International Conference on Information Science and Security (ICISS), pp. 1–6, December 2016. https://doi.org/10.1109/ICISSEC.2016.7885840
16. Sharafaldin, I., Lashkari, A., Ghorbani, A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: Proceedings of the 4th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP, pp. 108–116. INSTICC, SciTePress (2018). https://doi.org/10.5220/0006639801080116, ISBN: 978-989-758-282-0
17. Lin, P., Ye, K., Xu, C.-Z.: Dynamic network anomaly detection system by using deep learning techniques. In: Da Silva, D., Wang, Q., Zhang, L.-J. (eds.) CLOUD 2019. LNCS, vol. 11513, pp. 161–176. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23502-4_12
18. Aksu, D., Aydin, M.: Detecting port scan attempts with comparative analysis of deep learning and support vector machine algorithms. In: 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), pp. 77–80, December 2018. https://doi.org/10.1109/IBIGDELFT.2018.8625370
19. Aksu, D., Üstebay, S., Aydin, M.A., Atmaca, T.: Intrusion detection with comparative analysis of supervised learning techniques and fisher score feature selection algorithm. In: Czachórski, T., Gelenbe, E., Grochla, K., Lent, R. (eds.) ISCIS 2018. CCIS, vol. 935, pp. 141–149. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00840-6_16
20. Saber, M., El Farissi, I., Chadli, S., Emharraf, M.,Belkasmi, M.: Performance analysis of an intrusion detection systems based of artificial neural network. In: Europe and MENA Cooperation Advances in Information and Communication Technologies, pp. 511–521. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-46568-5_52, ISBN: 978-3-319-46568-5
21. Hodo, E., Bellekens, X., Hamilton, A., Tachtatzis, C., Atkinson, R.: Shallow and deep networks intrusion detection system: a taxonomy and survey. CoRR, abs/1701.02145, 2017. http://arxiv.org/abs/1701.02145
22. Papamartzivanos, D., Gómez Mármol, D., Kambourakis, G.: Introducing deep learning self-adaptive misuse network intrusion detection systems. IEEE Access **7**, 13546–13560 (2019). https://doi.org/10.1109/ACCESS.2019.2893871. ISSN: 2169–3536
23. Karatas, G., Demir, O., Koray Sahingoz, O.: Deep learning in intrusion detection systems. In: 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), pp. 113–116, December 2018. https://doi.org/10.1109/IBIGDELFT.2018.8625278
24. Yang, K., Liu, J., Zhang, C., Fang, Y.: Adversarial examples against the deep learning based network intrusion detection systems. In: MILCOM 2018–2018 IEEE Military Communications Conference (MILCOM), pp. 559–564, October 2018. https://doi.org/10.1109/MILCOM.2018.8599759
25. Gurung, S., Ghose, M., Subedi, A.: Deep learning approach on network intrusion detection system using NSL-KDD dataset. Int. J. Comput. Netw. Inf. Secur. **11**(3), 8 (2019). https://ucd.idm.oclc.org/login?url=search-proquest-com.ucd.idm.oclc.org/docview/2193195455?accountid=14507

26. McHugh, J.: Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. ACM Trans. Inf. Syst. Secur. **3**(4), 262–294 (2000)
27. Vinayakumar, R., Alazab, M., Soman, K., Poornachandran, P., Al-Nemrat, A., Venkatraman, S.: Deep learning approach for intelligent intrusion detection system. IEEE Access, 41525–41550 (2019). https://doi.org/10.1109/ACCESS.2019.2895334, ISSN: 2169–3536
28. Ustebay, S., Turgut, Z., Aydin, M.: Intrusion detection system with recursive feature elimination by using random forest and deep learning classifier. In: 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), pp. 71–76, December 2018. https://doi.org/10.1109/IBIGDELFT.2018.8625318
29. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann Publishers Inc., Burlington (2011). ISBN 0123814790, 9780123814791
30. Dunham, M.: Data Mining: Introductory and Advanced Topics. Prentice Hall PTR, Upper Saddle River (2002). ISBN 0130888923
31. Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q.: A deep learning approach to network intrusion detection. IEEE Trans. Emerg. Top. Comput. Intell. **2**(1), 41–50 (2018). https://doi.org/10.1109/TETCI.2017.2772792
32. Witten, I., Frank, E., Hall, M.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann Publishers Inc., Burlington (2011). ISBN 0123748569
33. Suh, S.: Practical Applications of Data Mining. Jones & Bartlett Learning, January 2011. ISBN 9780763785871
34. Benkhelifa, E., Welsh, T., Amouda, W.: A critical review of practices and challenges in intrusion detection systems for IoT: towards universal and resilient systems. IEEE Commun. Surv. Tutor. **20**(4), 1–15 (2018)
35. Pajouh, H., Javidan, R., Khayami, R., Ali, D., Choo, K.-K.R.: A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. IEEE Trans. Emerg. Top. Comput. **7**, 1–11 (2016)
36. Moustafa, N., Adi, E., Turnbull, B., Hu, J.: A new threat intelligence scheme for safeguarding industry 4.0 systems. IEEE Access **6**, 32910–32924 (2018)
37. Elrawy, M., Awad, A. and Hamed, H.: Intrusion detection systems for IoT-based smart environments: a survey. J. Cloud Comput. (2018). ISSN 2192–113X 10.1186/s13677-018-0123-6
38. Deng, L., Li, D., Yao, X., Cox, D., Wang, H.: Mobile network intrusion detection for IoT system based on transfer learning algorithm. Cluster Comput. **22**(4), 9889–9904 (2018). https://doi.org/10.1007/s10586-018-1847-2
39. Amouri, A., Alaparthy, V., and Morgera, S.: Cross layer-based intrusion detection based on network behavior for IoT. In 2018 IEEE 19th Wireless and Microwave Technology Conference (WAMICON), pp. 1–4, April 2018. https://doi.org/10.1109/WAMICON.2018.8363921
40. Liu, L., Xu, B., Zhang, X., Wu, X.: An intrusion detection method for internet of things based on suppressed fuzzy clustering. EURASIP J. Wirel. Commun. Netw. **2018**(1), 1–7 (2018). https://doi.org/10.1186/s13638-018-1128-z
41. Colom, J., Gil, D., Mora, H., Volckaert, B., Jimeno, A.: Scheduling framework for distributed intrusion detection systems over heterogeneous network architectures. J. Netw. Comput. Appl. **108**, 76–86 (2018). https://doi.org/10.1016/j.jnca.2018.02.004. ISSN 1084-8045

42. Roman, R., Lopez, J., Mambo, M., et al.: Mobile edge computing, fog: a survey and analysis of security threats and challenges. Future Gener. Comput. Syst. **78**, 680–698 (2018). https://doi.org/10.1016/j.future.2016.11.009. ISSN 0167-739X
43. Fernandez, G.: Deep Learning Approaches for Network Intrusion Detection, MSc Thesis Presented to the Graduate Faculty of The University of Texas at San Antonio, May 2019
44. Manimurugan, S., Al-Mutairi, S., Aborokbah, M., Chilamkurti, N., Ganesan, S., Patan, R.: Effective attack detection in internet of medical things smart environment using a deep belief neural network. IEEE Access **8**, 77396–77404 (2020). https://doi.org/10.1109/ACCESS.2020.2986013
45. Pelletier, Z., Abualkibash, M.: Evaluating the CIC IDS-2017 dataset using machine learning methods and creating multiple predictive models in the statistical computing language R. Int. Res. J. Adv. Eng. Sci. **5**(2), 187–191 (2020)