

Master's Thesis

Dealing with Imbalanced Classes in Bot-IoT Dataset

Jesse Atuhurra

Program of Information Science and Engineering
Graduate School of Science and Technology
Nara Institute of Science and Technology

Supervisor: Prof. Professor Shoji Kasahara
Large-Scale Systems Management Lab. (Division of Information Science)

Submitted on March 17, 2022

A Master's Thesis
submitted to Graduate School of Science and Technology,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
MASTER of ENGINEERING

Jesse Atuhurra

Thesis Committee:

Professor Shoji Kasahara
(Supervisor, Division of Information Science)
Professor Youki Kadobayashi
(Co-Supervisor, Division of Information Science)
Associate Professor Masahiro Sasabe
(Co-Supervisor, Division of Information Science)
Associate Professor YuanYu Zhang
(Co-Supervisor, Xidian University)
Assistant Professor Takanori Hara
(Co-Supervisor, Division of Information Science)

Dealing with Imbalanced Classes in Bot-IoT Dataset*

Jesse Atuhurra

Abstract

With the rapidly spreading usage of internet of things (IoT) devices, a network intrusion detection system (NIDS) has an important role to detect and protect various types of attacks in the IoT network. To evaluate the robustness of the NIDS in the IoT network, the existing work proposed a realistic botnet dataset in the IoT network (Bot-IoT dataset) and applied it to machine learning based anomaly detection. This dataset contains imbalanced normal and attack packets because the number of normal packets is much smaller than that of attack ones. The nature of imbalanced data may make it difficult to identify the minority class correctly. In this thesis, to address the class imbalance problem in the Bot-IoT dataset, we propose a binary classification method with synthetic minority over-sampling techniques (SMOTE). The proposed classifier aims at detecting the attack packets and overcoming the class imbalance problem with the help of SMOTE algorithm. Through numerical results, we demonstrate the fundamental characteristics of the proposed classifier and the impact of imbalanced data on the classifiers' performance.

Keywords:

Network intrusion detection system (NIDS), IoT network, SMOTE algorithm, class imbalance problem, machine learning

*Master's Thesis, Graduate School of Science and Technology, Nara Institute of Science and Technology, March 17, 2022.

Contents

1. Introduction	2
2. Related Work	5
3. Background	7
3.1 Bot-IoT Dataset	7
3.2 Machine Learning based Binary Classification	7
3.2.1 Logistic Regression	7
3.2.2 Support Vector Machine	9
3.2.3 Random Forest	10
3.2.4 Extreme Gradient Boosting	11
3.2.5 Multi-layer Perceptron Neural Network	12
3.3 Data Sampling Methods	12
4. Proposed Method	14
4.1 Overview	14
4.2 Preprocessing	15
4.3 Data Sampling	17
4.4 Binary Classifiers	18
5. Numerical Results	19
5.1 Evaluation Settings	19
5.2 Performance Comparison among Classifiers	21
5.3 Impact of Data Sampling	22
6. Conclusion	25
References	28
Publication List	34

List of Figures

1	Comparison of datasets used in intrusion detection (F=False, T=True).	6
2	An example of MLP with two hidden layers.	11
3	Intrusion Detection analysis based on Bot-IoT dataset.	14
4	Feature importance with the random forest algorithm.	15
5	Feature importance with the mutual information algorithm. . . .	16
6	Feature importance with the chi-squared algorithm.	16
7	FPR and FNR on Imbalanced and Balanced data.	23
8	SMOTE impact on Precision and Recall of Minority Class. . . .	24
9	Inference Time on Imbalanced and Balanced data.	25

List of Tables

1	BoT-IoT dataset features.	8
2	Class distribution in the BoT-IoT dataset.	9
3	Features selected by each feature selection algorithm.	17
4	Training and testing dataset sizes.	19
5	Confusion matrix.	19
6	Performance comparison among the proposed classifiers trained on imbalanced data.	21
7	Performance comparison among the proposed classifiers trained on the balanced data.	22

1. Introduction

Nowadays, the internet of things (IoT) plays a key role to facilitate and advance services and encompasses various application domains [22]. With rapidly increasing heterogeneous connected devices, that is, IoT devices, various services are constantly being created, and thus the network traffic in IoT networks has exponentially been increasing. To protect these devices against cyber-attacks included in the huge amount of network traffic, the need for IoT security arises [19]. The existing IoT networks are vulnerable to various types of attacks due to the accessibility to devices from anywhere via the internet and the proliferation of low-level security protection. These attacks lead to not only critical damage of infrastructures but also privacy violation. Since the distributed nature of IoT networks makes it difficult to monitor the network and to collect audit data, it is hard to take an effective security strategy against various attacks included in the huge amount of network traffic due to the heterogeneity of IoT devices and the limited resources available in IoT devices.

Under such background, IoT networks have increasingly become susceptible to intruder attacks, e.g., denial of service (DoS), distributed denial of service (DDoS), probe attacks, and IP spoofing. To protect these attacks by intruders, an intrusion detection system (IDS) [21, 26] was proposed, which is a monitoring system to detect abnormal and/or malicious activities by analyzing audit data. The existing studies have addressed the anomaly detection using the IDS, which monitors the normal activities of network traffic and alerts the administrator to the activity if any activities which deviate from the normal activities are found. In addition, several studies have proposed a network-based IDS (NIDS) [2] placed at the fog node in the IoT network to address the issue of huge latency [3, 9, 33, 37].

The NIDS's built on machine learning (ML) have attracted many researchers [9, 14, 17, 32, 37, 41], since the existing NIDS has the limitation of scalability and autonomic self-adoption due to constantly evolving attacks and threats in the IoT network. The combination of an NIDS and ML is expected to overcome these limitations. However, the existing ML methods, e.g., support vector machine (SVM), k -nearest neighbor (KNN), and decision tree (DT), may demonstrate low performance on anomaly detection because the NIDS requires high-dimensional representation. Deep neural network based NIDS's were proposed [1, 24], which

can exhibit high performance by exploring the high-dimensional representation from the audit data. In general, the audit data generated from network traffic tends to have the class imbalance problem because the activity which deviates from the normal activities, i.e., the attack, is a rare case. More precisely, such imbalanced traffic data has majority of normal traffic and minority of abnormal traffic. With imbalanced data, it becomes difficult for the classifier to correctly identify traffic which belongs to the minority class.

To apply ML to the NIDS, there are various datasets related to the cyber-attacks [4, 23, 27, 38–40]. In particular, Koroniotis et al. published the realistic botnet dataset, for network intrusion detection and network forensic analytics, which contains IoT network traffic including various types of attacks [23]. In [23], they evaluated the reliability of this dataset by using several ML methods and showed the baseline of binary classification. This dataset also contains imbalanced normal and attack packets because the number of normal packets is much smaller than that of attack ones.

We will describe the detail of the Bot-IoT dataset in Section 3.1.

In this thesis, to address the class imbalance problem in the Bot-IoT dataset, we propose a binary classification method at the network packet level considering the nature of imbalanced data. The proposed method aims at identifying normal or attack packets and overcoming the class imbalance problem with the help of the existing sampling method, i.e., synthetic minority over-sampling technique (SMOTE) [11]. Before selecting SMOTE technique for sampling, we report that we tried random over sampling and random under sampling too but SMOTE sampling showed more consistent results. Therefore we used SMOTE sampling for further analysis. Through numerical results, we demonstrate the fundamental characteristics of the proposed method from the viewpoint of the impact of SMOTE sampling on the performance of classifiers, during intrusion detection. Moreover, though binary classification is reported in this work, it is possible to extend the binary classification to multi-class classification especially after an attack has been detected, to identify the actual category of the attack.

The main contributions of this thesis are as follows:

1. Use the Bot-IoT dataset to develop a NIDS for IoT networks.
2. Investigate the class imbalance problem in Bot-IoT dataset.

The rest of this thesis is organized as follows. Section 2 gives the related work. In Section 3, we introduce the Bot-IoT dataset and ML techniques used in this thesis. In Section 4, we propose a binary classification method to deal with the class imbalance. Section 5 shows the fundamental characteristics of the proposed method. Finally, we give the conclusion and future work in Section 6.

2. Related Work

Many researchers have studied an NIDS to protect various types of attacks by intruders [2]. An NIDS runs on a strategic point and inspects the traffic among all the devices on the network. In the domain of IoT networks, several studies have proposed NIDS's placed at the fog node in the IoT network [3, 33, 37]. Aliyu et al. proposed a resource efficient IDS for man in the middle attacks at the fog layer [3]. There are studies to apply ML to the NIDS [33, 37]. Reddy et al. proposed an extreme greedy boosting ensemble method [12] based NIDS running at the fog node to monitor network traffic in IoT network by identifying and classifying the attacks based on abnormal activities [33]. Sahar et al. developed a deep learning based NIDS implemented on the fog node [37]. Recent surveys can be found in [9, 14, 17, 32, 41]. In this thesis, we propose the ML-based binary classification for an NIDS in IoT network.

There are various datasets related to intrusion detection [4, 23, 27, 38–40]. These datasets are summarized in the Figure 1. From this figure, we can see that only the Bot-IoT dataset contains traces of IoT network traffic. This makes it plausible for the analysis of intrusion detection in IoT networks.

The KDD dataset is a commonly used dataset for the evaluation of intrusion detection and contains seven weeks of network traffic including simulated attacks. In [40], Tavallaee et al. proposed NSL-KDD dataset, which is an extended version of the KDD dataset such that it does not contain redundant and duplicate records, the number of records is reasonable. The UNSW-NB15 dataset is generated by IXIA PerfectStorm, Tcpdump, Argus, and Bro-IDS tools, which create some types of attacks [27]. Shiravi et al. created ISCX dataset consisting of the seven days of both normal and abnormal activities [39]. Sharafaldin et al. proposed CICIDS2017 dataset, which contains common attacks resembling the real-world data, i.e., packet capture [38]. In contrast to these studies, Koroniotis et al. focused on the IoT network and created the Bot-IoT dataset, which consists of the IoT network traffic including various types of attacks [23]. We will describe the details of the Bot-IoT in Section 3.1. Since we also focus on the NIDS in IoT networks, the Bot-IoT dataset is used in this thesis. In addition we also tackle the class imbalance problem by using the over-sampling method.

We note that our work is similar to a recent study conducted by [31]. How-

Dataset	Realistic Testbed Configuration	Realistic Traffic	Labeled Data	IoT Traces	Diverse Attack Scenarios	Full Packet Capture	New generated Features
Darpa98	T	F	T	F	T	T	F
KDD99	T	F	T	F	T	T	T
DEFCON8	F	F	F	F	T	T	F
UNIBS	T	T	T	F	F	T	F
CAIDA	T	T	F	F	F	F	F
LBNL	F	T	F	F	T	F	F
UNSW-NB15	T	T	T	F	T	T	T
ISCX	T	T	T	F	T	T	T
CICIDS2017	T	T	T	F	T	T	T
TUIDS	T	T	T	F	T	T	T
Bot-IoT	T	T	T	T	T	T	T

Figure 1: Comparison of datasets used in intrusion detection (F=False, T=True).

ever, the major differences are: (1) We removed all non-network features before any analysis is conducted, that is, *pkSeqID*, *seq*, *dur*, *mean*, *stddev*, *sum*, *min*, and *max* (2) We employ three algorithms to select the list of features needed for intrusion detection analysis (3) Whereas their study investigates three machine learning models namely K-Nearest Neighbours (KNN), Naive Bayes and Multi-layer Perceptron, we investigated seven models, that is, Logistic Regression, Linear SVC, Linear Kernel SVM, RBF Kernel SVM, Random Forest, Extreme Gradient Boosting, and Multi-layer Perceptron (4) We evaluate and select the best classifiers based on false positive rate, false negative rate, and inference time, on top of accuracy, recall, precision and F1-scores.

3. Background

In this section, we introduce the Bot-IoT dataset and ML techniques used in this thesis.

3.1 Bot-IoT Dataset

Koroniotis et al. developed a realistic testbed to simulate the IoT network traffic including various types of attacks, i.e., DDos, DoS, operating system and service scan, key logging and data theft attacks, and published the Bot-IoT dataset generated by this simulation [23]. The Bot-IoT dataset contains more than 72 million records, each of which represents a network packet comprising of 43 features and is categorized into either a *normal* packet or an *attack* one. Note that all the features in the dataset are shown in Table 1.

To evaluate the reliability of this dataset, the authors proposed ML-based algorithms to detect attack packets and demonstrated good accuracy [23]. However, the dataset used in this evaluation contains imbalanced normal and attack packets because the number of attack packets is much higher than that of normal ones, as shown in Table 2. This distribution is counterintuitive because we can usually monitor more normal packets than attack ones. Such imbalanced data may cause performance degradation in the minority class.

In this thesis, we address the class imbalance problem in the Bot-IoT dataset to improve the performance by using the data sampling techniques.

3.2 Machine Learning based Binary Classification

3.2.1 Logistic Regression

Logistic regression [36] is an extension to the linear regression to estimate a dependent variable, i.e., binary variable, from one or more independent variables, i.e., a feature vector $\mathbf{x}$, where $\mathbf{x} = (x_1, \dots, x_D)$ is a D dimensional feature vector and x_k ($1 \leq k \leq D$) is a feature value.

More precisely, given the independent variables $\mathbf{x}$, the logistic regression estimates the probability p that the independent variables $\mathbf{x}$ belongs to the positive class. The logistic regression uses the log-odds, i.e., logit function, where the logit

Table 1: BoT-IoT dataset features.

Feature	Description
<i>pkSeqID</i>	Row Identifier
<i>sbytes</i>	Source-to-destination byte count
<i>Stime</i>	Record start time
<i>dbytes</i>	Destination-to-source byte count
<i>flgs</i>	Flow state flags seen in trans- actions
<i>rate</i>	Total packets per second in transaction
<i>flgs number</i>	Numerical representation of feature flags
<i>srute</i>	Source-to-destination packets per second
<i>Proto</i>	Textual representation of transaction protocols present in network flow
<i>drate</i>	Destination-to-source packets per second
<i>proto_number</i>	Numerical representation of feature proto
<i>TnBPSrcIP</i>	Total Number of bytes per source IP
<i>saddr</i>	Source IP address
<i>TnBPDstIP</i>	Total Number of bytes per Destination IP
<i>sport</i>	Source port number
<i>TnP_PSrcIP</i>	Total Number of packets per source IP
<i>daddr</i>	Destination IP address
<i>TnP_PDstIP</i>	Total Number of packets per Destination IP
<i>dport</i>	Destination port number
<i>TnP_PerProto</i>	Total Number of packets per protocol
<i>pkts</i>	Total count of packets in transaction
<i>TnP_Per Dport</i>	Total Number of packets per dport
<i>bytes</i>	Total number of bytes in transaction
<i>AR_P_Proto_P_SrcIP</i>	Average rate per protocol per Source IP. (calculated by pkts/dur)
<i>state</i>	Transaction state
<i>AR_P_Proto_P_DstIP</i>	Average rate per protocol per Destination IP
<i>state_number</i>	Numerical representation of feature state
<i>ltime</i>	Record last time
<i>N_IN_Conn_P_SrcIP</i>	Number of inbound connections per source IP
<i>seq</i>	Argus sequence number
<i>N_IN_Conn_P_DstIP</i>	Number of inbound connections per destination IP
<i>dur</i>	Record total duration
<i>AR_P_Proto_P_Sport</i>	Average rate per protocol per sport
<i>mean</i>	Average duration of aggregated records
<i>AR_P_Proto_P_Dport</i>	Average rate per protocol per dport
<i>stddev</i>	Standard deviation of aggregated records
<i>Pkts_P_State_P_Protocol_P_DstIP</i>	Number of packets grouped by state of flows and protocols per destination IP
<i>sum</i>	Total duration of aggregated records
<i>Pkts_P_State_P_Protocol_P_SrcIP</i>	Number of packets grouped by state of flows and protocols per source IP
<i>min</i>	Minimum duration of aggregated records
<i>attack</i>	Class label: 0 for Normal traffic, 1 for Attack Traffic
<i>max</i>	Maximum duration of aggregated records
<i>category</i>	Traffic category
<i>spkts</i>	Source-to-destination packet count
<i>subcategory</i>	Traffic subcategory
<i>dpkts</i>	Destination-to-source packet count

function $z(\cdot)$ is defined as the logarithm of the odds ratio,

$$z(p) = \ln \left(\frac{p}{1-p} \right). \quad (1)$$

Table 2: Class distribution in the BoT-IoT dataset.

Class label	Number of samples
Normal	477
Attack	3,668,041

Here, we can interpret the logit function $z(\cdot)$ as the linear combination of independent variables, i.e, the dot product of a learnable weight vector $\mathbf{w} = (w_1, \dots, w_D)$ and a feature vector $\mathbf{x}$. Therefore, we can map the logit function $z(\cdot)$ to the value at the range of $[0, 1]$, i.e., the probability p , by using the inverse function of (1).

$$p = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-\mathbf{w}^T \cdot \mathbf{x}}}. \quad (2)$$

If the probability p is higher than or equal to a certain threshold, i.e., 0.5, $\mathbf{x}$ belongs to the positive class. The logistic regression aims at learning the weight vector $\mathbf{w}$ to maximize the logarithm of the likelihood function.

3.2.2 Support Vector Machine

A support vector machine (SVM) [5] is one of the supervised learning models and aims at finding an appropriate separating hyperplane in high-dimensional feature spaces to discriminate between categories by using a kernel function, e.g., linear, polynomial, or radial basis function. In case of binary classification, the SVM classifier calculates the hyperplane to distinguish between the data belonging to a certain class and the rest of data. The appropriate hyperplane is obtained by maximizing the distance from the hyperplane to the closest point across both classes under the assumption where the training data are linearly separable. This maximum distance is called the *maximum margin separator*.

Consider a training dataset of n points of the form $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$ where the y_i are either 1 or -1 , each indicating the class to which the point $\mathbf{x}_i$ belongs. Each $\mathbf{x}_i$ is a real vector of dimensions D . Our aim is to find the hyperplane of maximum margin that divides the group of points $\mathbf{x}_i$ for which $y_i = 1$, from the group of points for which $y_i = -1$. Our aim is formulated so that the distance between the hyperplane and the nearest point $\mathbf{x}_i$ from either group is maximized. The hyperplane can be written as the set of points $\mathbf{x}$ which satisfy $\mathbf{w}^T \mathbf{x} - b = 0$,

where $\mathbf{w}$ is the normal vector to the hyperplane. The parameter $\frac{b}{\|\mathbf{w}\|}$ determines the offset of the hyperplane from the origin along the normal vector $\mathbf{w}$, that is, the margin. To compute the soft-margin SVM classifier is equivalent to minimizing the optimization problem:

$$\left[\frac{1}{n} \sum_{i=1}^n \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b)) \right] + \lambda \|\mathbf{w}\|^2. \quad (3)$$

The *primal problem* definition is obtained by reformulating the optimization problem above as:

$$\min \frac{1}{n} \sum_{i=1}^n \zeta_i + \lambda \|\mathbf{w}\|^2, \text{ subject to } y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1 - \zeta_i \text{ and } \zeta_i \geq 0, \forall i. \quad (4)$$

The variable $\zeta_i = \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b))$ is introduced for each $i \in \{1, \dots, n\}$. The *primal problem* can be further simplified to give the dual maximization problem, that is the *dual problem*. The *dual problem* is a quadratic programming problem and it is defined below:

$$\max f(c_1 \dots c_n) = \sum_{i=1}^n c_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i c_i (\mathbf{x}_i^T \mathbf{x}_j) y_j c_j, \quad (5)$$

subject to $\sum_{i=1}^n c_i y_i = 0$, and $0 \leq c_i \leq \frac{1}{2n\lambda}, \forall i$.

The variables c_i are defined such that $\mathbf{w} = \sum_{i=1}^n c_i y_i \mathbf{x}_i$. Moreover, $c_i = 0$ when $\mathbf{x}_i$ lies on the correct side of the margin, and $0 < c_i < (2n\lambda)^{-1}$ when $\mathbf{x}_i$ lies on the margin's boundary. It follows that $\mathbf{w}$ can be written as a linear combination of the support vectors. The offset, b can be recovered by finding an $\mathbf{x}_i$ on the margin's boundary and solving $b = \mathbf{w}^T \mathbf{x}_i - y_i$.

The sequential minimal optimization (SMO) algorithm was proposed, which is a fast algorithm for solving the dual maximization problem in order to train SVMs [30].

3.2.3 Random Forest

Random forest (RF) [6] is a supervised learning method and ensemble learning using multiple decision trees, where the ensemble learning algorithm combines multiple ML algorithms to obtain higher performance. The RF creates multiple

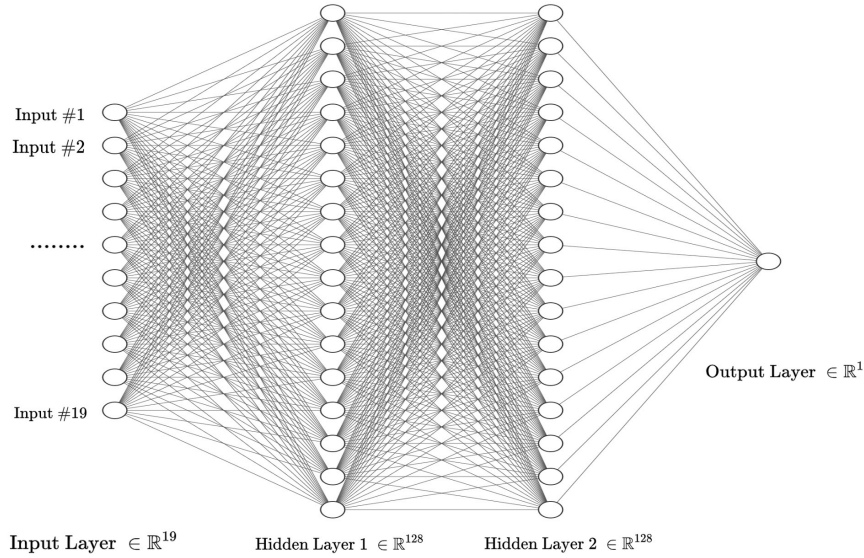


Figure 2: An example of MLP with two hidden layers.

decision trees, each of which is generated from a different training subset provided by the training dataset sampled with replacement. Each decision tree is independently trained and outputs a classification result. The RF outputs a classification result based on majority voting after aggregating all the classification results of all the decision trees.

3.2.4 Extreme Gradient Boosting

The extreme gradient boosting (XGBoost) algorithm [12] is an extended version of the gradient boosting decision tree (GBDT) [44] in a distributed manner. The GBDT is ensemble learning based on multiple decision trees similar to the RF. The GBDT, however, has a different aspect from the RF in terms of the ensemble algorithm.

The ensemble algorithm used in the RF combines full decision trees in a parallel manner while that used in the GBDT produces a classifier by combining weaker decision trees in a sequential manner.

3.2.5 Multi-layer Perceptron Neural Network

Multi-layer perceptron (MLP) is a multiple feedforward artificial neural network with one or more hidden layers between the input and output layers and succeeds in many applications including classification. Fig. 2 illustrates an example of MLP with two hidden layers. The MLP can be defined as a directed graph with multiple node layers. The left (resp. right) side layer indicates the input (resp. output) layer and the others mean the hidden layers. The MLP is a fully connected network where every node on a certain layer has connections with a certain weight to all the nodes on the next layer, except for the output layer. Each node corresponds to a processing unit with a nonlinear activation function, e.g., rectified linear unit function, except for the nodes on the input layer. Thanks to nonlinear mapping, the MLP can approximate to any continuous function, which is well-known as universal approximation theorem [13]. A backpropagation algorithm [34], which is one of the supervised learning methods, is adopted to train MLP.

3.3 Data Sampling Methods

The class imbalance problem arises from the real-world data.

In general, such imbalanced data causes performance degradation in the minority class. To tackle this issue, some sampling techniques were proposed [8, 11, 18, 20, 25]. Random minority over-sampling (ROS) and random majority under-sampling (RUS) are commonly used and increase the sensitivity of a classifier to the minority class. The ROS randomly duplicates samples with the minority class while the RUS randomly discards samples with the majority class from the dataset. Chawla et al. proposed an over-sampling method called synthetic minority over-sampling technique (SMOTE), where the minority class is over-sampled by creating synthetic data [11].

The SMOTE algorithm aims at generating synthetic samples with the minority class according to a certain sampling rate N based on the imbalanced proportion by the following procedures. This algorithm first finds the k -nearest neighbors $\mathcal{X}_k(\mathbf{x}_i) = \{\mathbf{x}_1 \dots, \mathbf{x}_k\}$ of each sample $\mathbf{x}_i \in \mathcal{X}$ by calculating the Euclidean distance between $\mathbf{x}_i$ and every sample $\mathbf{x}_j \in \mathcal{X}$ ($i \neq j$), and selecting

samples with the smallest Euclidean distance from the original sample. Here, we define $\mathcal{X}$ ($X = |\mathcal{X}|$) as a set of samples with the minority class and $\mathcal{X}_k(\mathbf{x}_i)$ as k -nearest neighbors of sample $\mathbf{x}_i$. Next, for each sample $\mathbf{x}_i \in \mathcal{X}$, the algorithm randomly selects one of the k -nearest neighbors $\mathcal{X}_k(\mathbf{x}_i)$, i.e., $\mathbf{x}_l$ ($1 \leq l \leq k$), and generates a new synthetic sample $\mathbf{x}'$ as follows:

$$\mathbf{x}' = \mathbf{x}_i + \gamma(\mathbf{x}_l - \mathbf{x}_i), \quad (6)$$

where γ denotes a vector of uniform random values between 0 and 1. We repeat the above selection and generation until N synthetic samples are generated. Applying these procedures to all samples in $\mathcal{X}$, we consequently obtain NX synthetic samples with minority class.

In addition to the SMOTE algorithm, several variants of the SMOTE algorithm have been proposed [8, 18, 20]. We should note that such over-sampling techniques may change the nature of the dataset but improve the performance of the classifiers [42].

4. Proposed Method

In this section, we propose binary classification to deal with the class imbalance problem in the Bot-IoT dataset.

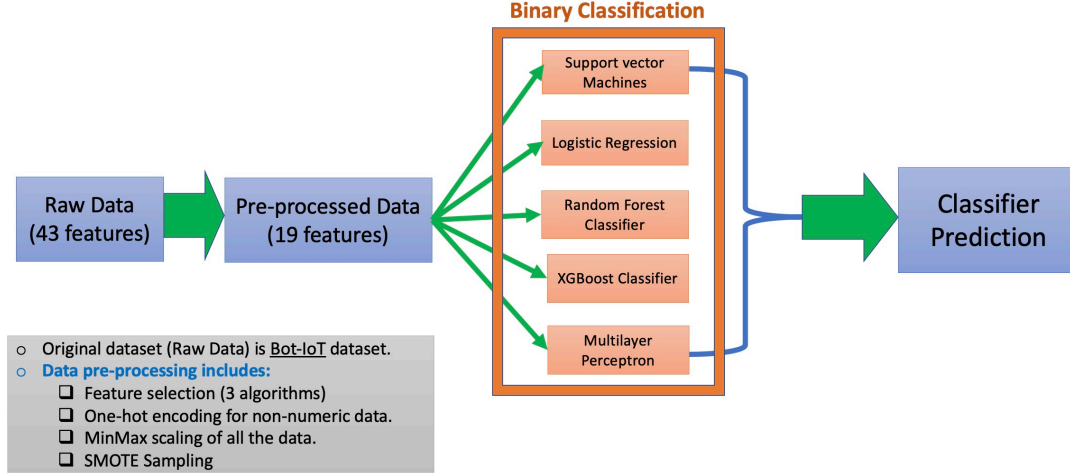


Figure 3: Intrusion Detection analysis based on Bot-IoT dataset.

4.1 Overview

As mentioned in Section 3.1, the Bot-IoT dataset contains imbalanced normal and attack packets because the number of attack packets is much higher than that of normal ones. This may cause performance degradation in the minority class. We propose the binary classification method to address the class imbalance problem in the Bot-IoT dataset. The proposed method consists of mainly three stages, i.e., preprocessing, data sampling, and binary classification. We first conduct the preprocessing to tackle the curse of dimensionality, which will be shown in Section 4.2. To investigate the positive (resp. negative) impact of the balanced (resp. imbalanced) dataset, we use the SMOTE algorithm, which generates the synthetic samples such that number of normal packets is equivalent to that of attack ones. (See the details of the data sampling in Section 4.3.) The proposed binary classifiers aim at detecting the attack packets in Bot-IoT dataset. As for performance comparison purposes, we adopt several binary classifiers, i.e.,

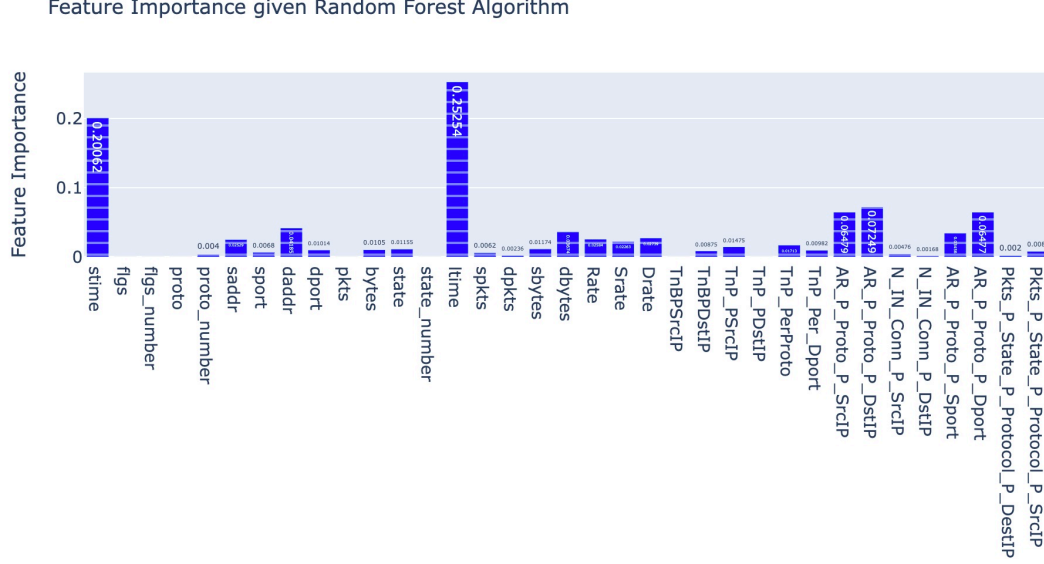


Figure 4: Feature importance with the random forest algorithm.

logistic regression, SVM, RF, XGBoost, and MLP. We will describe the details of the proposed binary classifiers in Section 4.4.

4.2 Preprocessing

We use the Bot-IoT dataset [23] containing 43 features. Recall that all the features in the Bot-IoT dataset are shown in Table 1. To tackle curse of dimensionality, we select important features related to the prediction accuracy from the original features by using feature selection algorithms based on three metrics, i.e., random forest [7], mutual information [35], and chi-squared algorithm [16]. Before executing the feature selection algorithms, we remove the features not related to the typical characteristics of the network traffic, i.e., *pkSeqID*, *seq*, *dur*, *mean*, *stddev*, *sum*, *min*, and *max*, from the original Bot-IoT dataset. Applying each algorithm to the Bot-IoT dataset with the remaining features, we calculate the feature importance scores with the three algorithms, as shown in Figs. 4 through 6. Figs. 4 through 6 illustrate the feature importance by applying each feature selection algorithm, respectively. We manually select the important features such that feature importance derived by each algorithm is higher than a certain thresh-

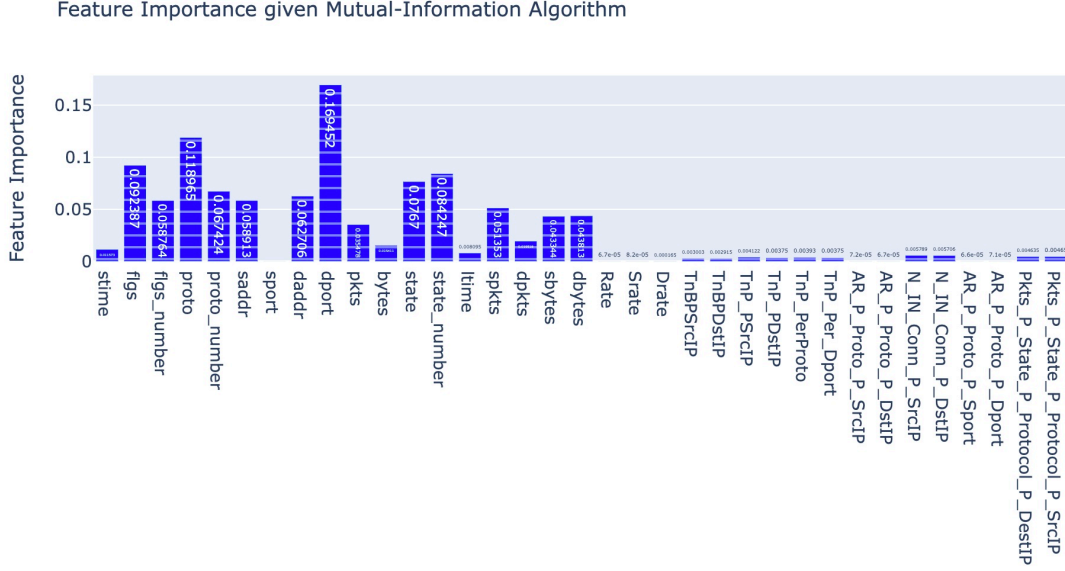


Figure 5: Feature importance with the mutual information algorithm.

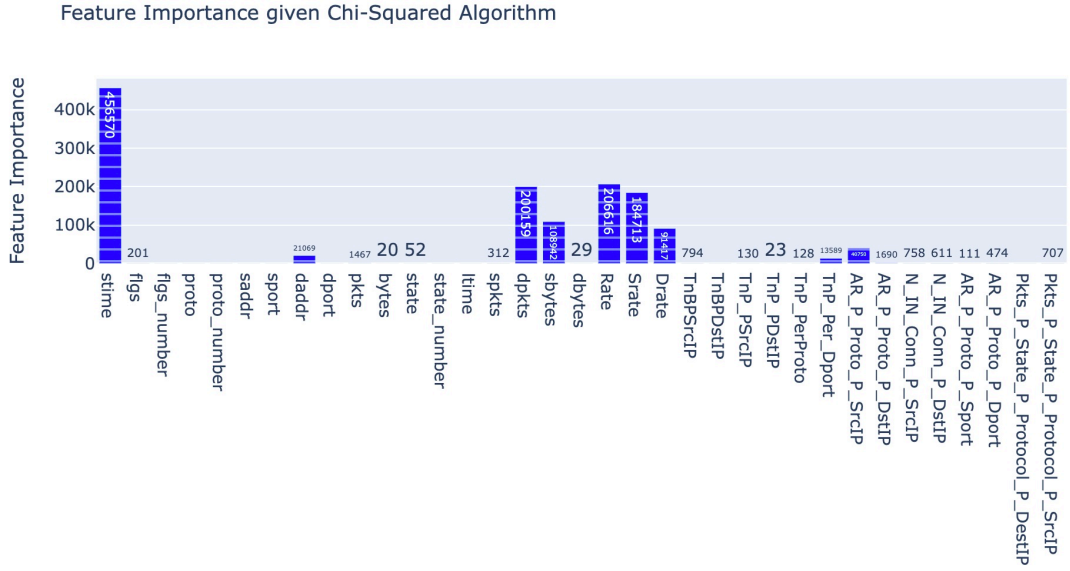


Figure 6: Feature importance with the chi-squared algorithm.

old. Table 3 shows the features selected by each feature selection algorithm. We use the 19 features, which is the union set of the features selected by the three

Table 3: Features selected by each feature selection algorithm.

Chi-squared	Mutual information	Random forest
<i>srate</i>	<i>dport</i>	<i>ltime</i>
<i>sport</i>	<i>proto</i>	<i>stime</i>
<i>AR_P_Proto_P_Sport</i>	<i>flgs</i>	<i>AR_P_Proto_P_DstIP</i>
<i>AR_P_Proto_P_SrcIP</i>	<i>state</i>	<i>AR_P_Proto_P_SrcIP</i>
<i>AR_P_Proto_P_DstIP</i>	<i>proto_number</i>	<i>AR_P_Proto_P_Dport</i>
<i>rate</i>	<i>daddr</i>	<i>daddr</i>
<i>ARP_Proto_P_Dport</i>	<i>saddr</i>	<i>AR_P_Proto_P_Sport</i>
	<i>flgs_number</i>	<i>rate</i>
		<i>TnP_Per_Proto</i>
		<i>bytes</i>

feature selection algorithms.

Since this dataset contains categorical features, we apply the one-hot encoding to the categorical features, which results in the numeric features. We also apply the min-max normalization to each feature to improve the classifier performance. More precisely, the min-max normalization translates the d th ($1 \leq d \leq D$) feature value $x_{i,d}$ of the i th sample (D dimensional feature vector) $\mathbf{x}_i = (x_{i,1}, \dots, x_{i,D})$ to the following value $x_{i,d}^{\text{normalized}}$:

$$x_{i,d}^{\text{normalized}} = \frac{x_{i,d} - \min_{\mathbf{x}_j \in \mathcal{X}} x_{j,d}}{\max_{\mathbf{x}_j \in \mathcal{X}} x_{j,d} - \min_{\mathbf{x}_j \in \mathcal{X}} x_{j,d}}, \quad (7)$$

where $\mathcal{X}$ denotes a set of all samples. As a result, we obtain the normalized feature value between zero and one.

4.3 Data Sampling

Since the Bot-IoT dataset contains imbalanced normal and attack packets, the classifier performance will be skewed. To address the class imbalance problem, we apply the SMOTE algorithm [11] to the dataset with the minority class, i.e., normal packets, which generates the synthetic samples from the dataset with the minority class such that the number of normal packets is equivalent to that of

attack ones. Remember that the detail of the SMOTE algorithm is described in Section 3.3. We use the oversampled dataset only for the training phase to mitigate the skewed classifier performance.

4.4 Binary Classifiers

The proposed binary classifier aims at detecting the attack packets from the packets in the Bot-IoT dataset. As for performance comparison purposes, we use several binary classifiers, i.e., logistic regression, linear SVM Classification (LinearSVC), linear kernel SVM, radial basis function (RBF) kernel SVM, random forest, XGBoost, and MLP. To train the SVM model, we adopt the two types of the solvers, i.e., *liblinear* solver [15] and *libsvm* one [10]. The *liblinear* solver is used for the Linear SVC without kernel transform while the *libsvm* solver is used for the Linear and RBF Kernel SVMs. As a result, the former can quickly solve the SMO algorithm and deal with a large amount of data, compared with the latter.

To realize the malicious packet detection, each of them is trained by using the preprocessed training dataset after applying the SMOTE algorithm. Given the preprocessed packet as the input, the proposed binary classifier involves categorizing the packet within a specific class, i.e., a normal packet or an attack one.

Table 4: Training and testing dataset sizes.

Scenario	Training dataset size		Testing dataset size	
	attack packet	normal packet	attack packet	normal packet
Imbalanced data scenario	2,457,583	324	1,210,458	153
Balanced data scenario	2,457,583	2,457,583	1,210,458	153

Table 5: Confusion matrix.

		Prediction class	
		Attack packet	Normal packet
Actual class	Attack packet	True Positive (TP)	False Negative (FN)
	Normal packet	False Positive (FP)	True Negative (TN)

5. Numerical Results

In order to verify the positive (resp. negative) impact of the balanced (resp. imbalanced) dataset on the classifier performance, we evaluate the proposed binary classifiers. We first explain the evaluation settings in Section 5.1. We show the fundamental characteristics of the proposed binary classifiers in Section 5.2. Finally, we further demonstrate the impact of data sampling on the classifier performance in Section 5.3.

5.1 Evaluation Settings

We use the server with Intel CPU Xeon Gold 6226R 16 core and 200 GB memory and with NVIDIA GeForce RTX 3090 GPU and 25.45 GB memory running on CUDA Version 11.1. We implement the proposed binary classifiers, i.e., logistic regression, LinearSVC, linear kernel SVM, radial basis function (RBF) kernel SVM, random forest, XGBoost, and MLP, by using the python libraries, i.e., scikit-learn [29] and Pytorch [28].

As for evaluation, we use the hold-out method [43] to evaluate each of the proposed classifiers where the 67% of dataset is used for training while the rest of dataset is used for testing. Table 4 presents the training and testing dataset sizes in case of the imbalanced and balanced data scenarios, respectively. To investigate impact of class imbalance, we prepare the two types of the trained models, i.e., the model trained on imbalanced data and that trained on balanced

data. In the former case, we use the dataset without the SMOTE algorithm for training. On the other hand, we use the dataset with the SMOTE algorithm for training in the latter case. Note that we apply the SMOTE algorithm to only the training dataset. As for the SMOTE algorithm parameter, we adopt the number $k = 5$ of nearest neighbors to construct the synthetic samples.

There are possible four cases for the binary classification result of each packet, i.e., true positive (TP), true negative (TN), false negative (FN), and false positive (FP). Table 5 presents a confusion matrix used in this thesis. TP (resp. TN) indicates the case where the binary classifier correctly predicts the attack (resp. normal) packet. On the other hand, FN (resp. FP) refers to the case where the packet is within the attack (resp. normal) class but the binary classifier performs incorrect prediction for the attack (resp. normal) packet.

As for evaluation metrics, we use *accuracy*, *recall*, *precision*, *false negative rate* (FNR), *false positive rate* (FPR), and *F1-score*. These metrics can be calculated as follows:

$$\text{accuracy} = \frac{TP + TN}{TP + FN + FP + TN}, \quad (8)$$

$$\text{recall} = \frac{TP}{TP + FN}, \quad (9)$$

$$\text{precision} = \frac{TP}{TP + FP}, \quad (10)$$

$$\text{FNR} = \frac{FN}{TP + FN}, \quad (11)$$

$$\text{FPR} = \frac{FP}{FP + TN}, \quad (12)$$

$$\text{F1-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}, \quad (13)$$

where TP , TN , FN , and FP represent the numbers of TP, TN, FN, and FP events, respectively. We also use the area under a receiver operating characteristic (ROC) curve, i.e., *AUC-score*, where an ROC curve is a curve plotting the TP rate against the FP rate at different classification thresholds and AUC-score indicates the entire two-dimensional area under the entire ROC curve. In addition, we use *inference time*, which indicates the actual time required for inference on all the testing data.

Table 6: Performance comparison among the proposed classifiers trained on imbalanced data.

Scheme	accuracy	recall	precision	FNR	FPR	F1-score	AUC-score	inference time [s]
Logistic regression	99.9960	99.9999	99.9961	0.00008	30.72	99.9961	84.6404	0.05
Linear SVC	99.9964	99.9995	99.9968	0.0004	24.84	99.9968	87.5814	0.05
Linear kernel SVM	99.9960	100.0	99.9960	0.000	31.373	99.9960	84.3137	9.72
RBF kernel SVM	99.9985	99.9999	99.9986	0.00008	10.457	99.9986	94.7712	27.06
Random forest	99.8018	99.8032	99.9985	0.196	11.764	99.9985	94.0192	4.37
XGBoost	99.9950	99.9976	99.9974	0.0023	20.26	99.9974	89.8680	0.43
MLP	99.8619	99.8697	99.9922	0.130	53.10	99.9922	73.3811	0.0012

5.2 Performance Comparison among Classifiers

In this section, we focus on the fundamental characteristics of the proposed binary classifiers. Table 6 presents the performance comparison among the proposed binary classifiers trained on the imbalanced dataset. We first observe that all the proposed classifiers achieve high accuracy yet the training dataset contains class imbalance. Focusing on recall, we confirm that all the proposed classifiers achieve over 99% recall. This indicates the proposed classifiers can correctly detect the attack packets. In terms of precision, the proposed classifiers also achieve the high performance. Next, focusing on the FPR, we observe that all the proposed classifiers have relatively high FPR, compared with FNR. This is because the performance of the classifier is skewed towards the “attack” class due to the imbalanced training dataset, which results in a large number of false positive events. From the viewpoint of AUC, the RBF kernel SVM and Random forest exhibit the high performance, i.e., 94.01% and 94.07%, while the AUC-score of the MLP is 73.38%.

Next, we focus on the inference time. We observe that logistic regression, Linear SVC, XGBoost, and MLP exhibit small inference time less than one second while Linear and RBF kernel SVMs and Random forest show much higher inference time than them. The random forest requires long inference time to average the classification results obtained from the 1000 decision trees used for the binary classification. Since Linear and RBF kernel SVM adjust the hyperplane such that the samples are distinguished into their class, they require the higher inference time.

Table 7: Performance comparison among the proposed classifiers trained on the balanced data.

Scheme	accuracy	recall	precision	FNR	FPR	F1-score	AUC-score	inference time [s]
Logistic regression	99.9722	99.9724	99.9998	0.027	1.307	99.9998	99.3326	0.05
Linear SVC	99.9786	99.9788	99.9998	0.021	1.307	99.9998	99.3358	0.05
Linear kernel SVM	99.6527	99.6535	99.9991	0.346	6.535	99.9991	96.5587	44.10
RBF kernel SVM	99.9982	99.9984	99.9998	0.00156	1.307	99.9998	99.3456	76.66
Random forest	99.9956	99.9957	99.9998	0.0042	1.307	99.9998	99.3442	4.35
XGBoost	99.9945	99.9947	99.9998	0.0052	1.307	99.9998	99.3437	0.45
MLP	99.9761	99.9781	99.9819	0.021	0.027	99.9819	99.9755	0.0012

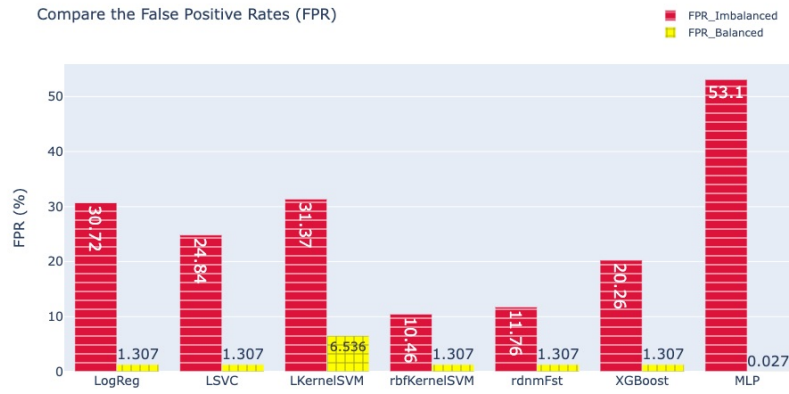
5.3 Impact of Data Sampling

In this section, we focus on the impact of class imbalance on the classifier performance. Table 7 presents the performance comparison among the proposed binary classifiers trained in the balanced dataset. Comparing Table 6, we first confirm that all the proposed classifiers trained on the balanced dataset can exhibit almost the same accuracy, recall, and precision compared with those trained on the imbalanced dataset, thanks to the oversampling method. We observe that the FPR is drastically improved because the proposed classifier is trained by using the large amount of data within the normal class.

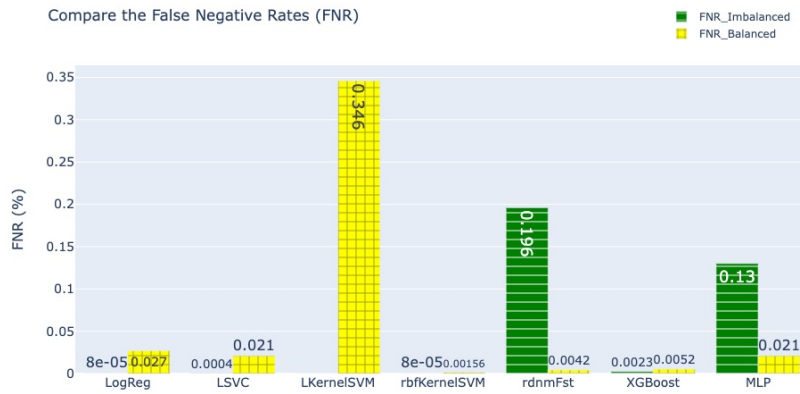
This result shows that the normal packet cannot be blocked from reaching the IoT devices by introducing the balanced dataset.

Next, we focus on how the oversampling method affects the inference time. We observe that the performance of Linear and RBF kernel SVM trained on the balanced data degrades in terms of inference time compared with those trained on the imbalanced data. This is because these classifiers require time to fit the large amount of samples by finding the hyperplane. On the other hand, the rest of the proposed classifiers trained in the balanced data exhibit almost the same inference time as those trained in the imbalanced data.

Figures 7a,7b,9,8a,8b that follow illustrate the observations observe. In these figures; LogReg, LSVC, LKernelSVM, rbfcKernelSVM, rdnmFst, XGBoost, and MLP refer to Logistic Regression, Linear SVC, Linear Kernel SVM, RBF Kernel SVM, Random Forest, Extreme Gradient Boosting, and Multi-layer Perceptron respectively.

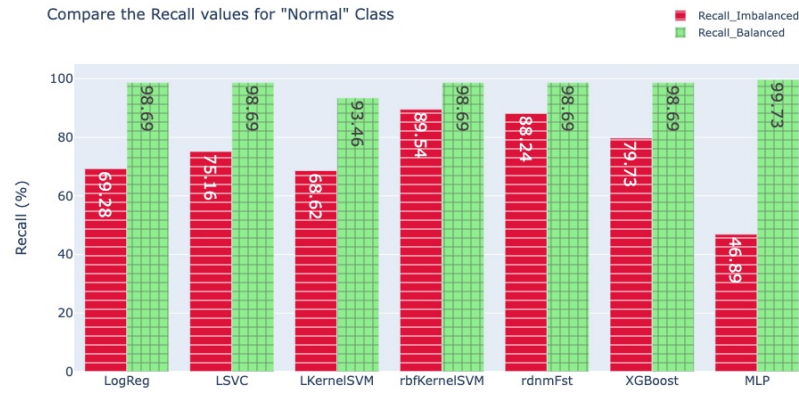


(a) FPR on Imbalanced and Balanced data.

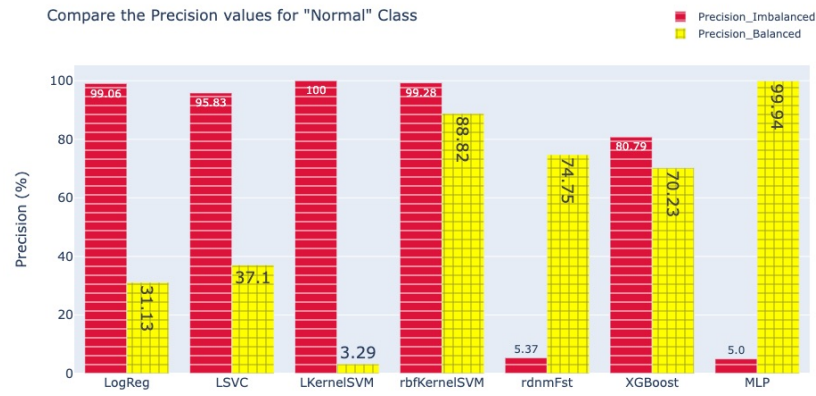


(b) FNR on Imbalanced and Balanced data.

Figure 7: FPR and FNR on Imbalanced and Balanced data.



(a) Recall on Imbalanced and Balanced data.



(b) Precision on Imbalanced and Balanced data.

Figure 8: SMOTE impact on Precision and Recall of Minority Class.

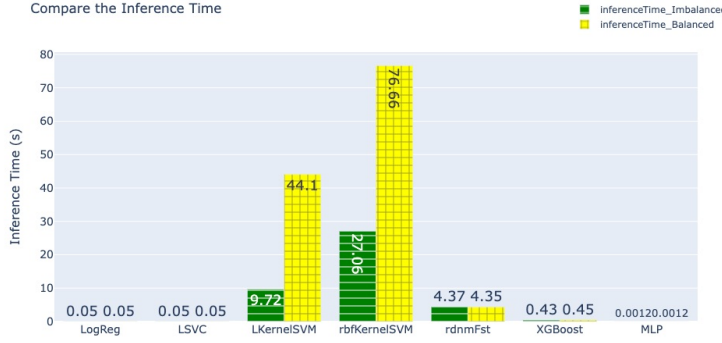


Figure 9: Inference Time on Imbalanced and Balanced data.

6. Conclusion

The existing Bot-IoT dataset contains imbalanced normal and attack packets since the number of normal packets is much smaller than that of attack ones. Such class imbalance leads to inaccurate results, especially for the minority class. In this thesis, we have addressed the class imbalance problem to apply the SMOTE algorithm to the original Bot-IoT dataset, where the SMOTE algorithm generates synthetic samples such that number of normal packets is equivalent to that of attack ones. We further have proposed the binary classification method to identify the normal and attack packet.

Through the numerical results, we first have shown the performance comparison among the different classifiers. We observe that all the proposed classifiers achieve high accuracy, recall, and precision. Next, we have demonstrated the impact of data sampling on the classifier performance. The proposed classifiers trained in the balanced dataset also achieves almost the same accuracy, recall, and precision as that in imbalanced dataset, and drastically improves the FPR.

In future work, we plan to develop the resource efficient NIDS based on both the reinforcement learning and federated learning to learn patterns of attacks in the IoT network over time. In addition, we aim to evaluate the performance of the NIDS under the resource constraints. The more immediate efforts include extending this work to do multi-class classification especially when an attack has been detected, in order to identify the actual type of attack.

Acknowledgment

I extend my sincere appreciation to my principal supervisor Professor Shoji Kasahara, for his continued guidance over the master’s course. Our time together at the Large-scale Systems Management Laboratory has certainly shaped me into the researcher I am today. It is due to his mentorship that I have been able to undertake such a challenging but very satisfying research project. I will always be indebted to him and I am very grateful for the opportunity he gave me to join his Laboratory at Nara Institute of Science and Technology (NAIST). He’s made my life in Japan convenient for research such that I have not been stressed throughout graduate school. Moreover, I am very thankful for the permission and the opportunity he gave me to undertake an internship with HONDA Research Institute Japan (HRI-JP). This has broadened my knowledge and understanding of specific Artificial Intelligence disciplines especially Robotics and Natural Language Processing. Sensei, I will always be grateful.

I’d also like to thank all my co-supervisors: Professor Youki Kadobayashi, Associate Professor Masahiro Sasabe, Associate Professor YuanYu Zhang, and Assistant Professor Takanori Hara, for the relentless support and guidance throughout this research project on Network Intrusion Detection. I am really delighted with the insights on the research direction that I got because I couldn’t make it to the end without such valuable help. Assistant Professor Takanori Hara has been very resourceful to me in terms of setting up the coding environments required to conduct the research experiments and the revision of this thesis. I appreciate your tremendous help Dr. Hara.

I cannot forget the valuable time I spent with Professor Masatoshi Yoshikawa at the Graduate school of Informatics of Kyoto University. His daily encouragement and words of advice helped me a lot to master several methods and techniques in Information Retrieval, Databases, Human-Computer Interface design and Artificial Intelligence.

I want to extend my appreciation to Kentaro Fujita who I met at the Large-scale Systems Management Laboratory. He played such a fundamental role in showing me how to implement several machine learning algorithms in Python. Thank you Fujita-san.

My friends at the laboratory have always been there for me, even as we have

been physically distanced by the COVID-19 pandemic. I started my research in the middle of the pandemic and only the constant updates from my friends—at the lab and the school at large—helped me to keep up with the latest information regarding laboratory research meetings, and seminars. Thank you

A big reason for my success has been the incredibly selfless love I have always gotten from my parents. To you, mom and dad, I say thank you very much. I extend special thanks to my siblings with whom we have continued to challenge each other to do better every day. To everyone back in Wakiso, Uganda and beyond, I am forever grateful for the incredible moments we have always shared together.

References

- [1] Zeeshan Ahmad, Adnan Shahid Khan, Cheah Wai Shiang, Johari Abdullah, and Farhan Ahmad. Network Intrusion Detection System: A Systematic Study of Machine Learning and Deep Learning Approaches. *Transactions on Emerging Telecommunications Technologies*, 32(1):e4150, 2021.
- [2] Mohiuddin Ahmed, Abdun Naser Mahmood, and Jiankun Hu. A Survey of Network Anomaly Detection Techniques. *Journal of Network and Computer Applications*, 60:19–31, January 2016.
- [3] Farouq Aliyu, Tarek Sheltami, and Elhadi M. Shakshuki. A Detection and Prevention Technique for Man in the Middle Attack in Fog Computing. *Procedia Computer Science*, 141:24–31, January 2018.
- [4] The UCI KDD Archive. KDD Cup 1999 Data. <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>, 2022. Accessed 14 Jan. 2022.
- [5] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.
- [6] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.
- [7] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
- [8] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Safe-Level-SMOTE: Safe-Level-Synthetic Minority Over-Sampling TEchnique for Handling the Class Imbalanced Problem. In Thanaruk Theeramunkong, Boonserm Kijsirikul, Nick Cercone, and Tu-Bao Ho, editors, *Proc. of Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 475–482, Berlin, Heidelberg, 2009. Springer.
- [9] Nadia Chaabouni, Mohamed Mosbah, Akka Zemmari, Cyrille Sauvignac, and Parvez Faruki. Network Intrusion Detection for IoT Security Based on Learning Techniques. *IEEE Communications Surveys & Tutorials*, 21(3):2671–2701, 2019.

- [10] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3), may 2011.
- [11] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, Jun 2002.
- [12] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proc. of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, August 2016. Association for Computing Machinery.
- [13] G. Cybenko. Approximation by Superpositions of a Sigmoidal Function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, December 1989.
- [14] A. V. Deorankar and Shiwani S. Thakare. Survey on Anomaly Detection of (IoT)- Internet of Things Cyberattacks Using Machine Learning. In *Proc. of 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, pages 115–117, March 2020.
- [15] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, jun 2008.
- [16] Karl Pearson F.R.S. X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175, 1900.
- [17] Shital Gulghane, Vishal Shingate, Shivani Bondgulwar, Gaurav Awari, and Parth Sagar. A Survey on Intrusion Detection System Using Machine Learning Algorithms. In Jennifer S. Raj, Abul Bashar, and S. R. Jino Ramson, editors, *Proc. of Innovative Data Communication Technologies and Application*, Lecture Notes on Data Engineering and Communications Technologies, pages 670–675, Cham, 2020. Springer International Publishing.

- [18] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In De-Shuang Huang, Xiao-Ping Zhang, and Guang-Bin Huang, editors, *Proc. of Advances in Intelligent Computing*, Lecture Notes in Computer Science, pages 878–887, Berlin, Heidelberg, 2005. Springer.
- [19] Vikas Hassija, Vinay Chamola, Vikas Saxena, Divyansh Jain, Pranav Goyal, and Biplab Sikdar. A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures. *IEEE Access*, 7:82721–82743, 2019.
- [20] Haibo He, Yang Bai, Eduardo A. Garcia, and Shutao Li. ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. In *Proc. of 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328, June 2008.
- [21] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. Survey of Intrusion Detection Systems: Techniques, Datasets and Challenges. *Cybersecurity*, 2(1):20, July 2019.
- [22] Jyoti Mante Khurpade, Devakanta Rao, and Parth. D. Sanghavi. A Survey on IOT and 5G Network. In *Proc. of 2018 International Conference on Smart City and Emerging Technology (ICSCET)*, pages 1–3, January 2018.
- [23] Nickolaos Koroniotis, Nour Moustafa, Elena Sitnikova, and Benjamin Turnbull. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Generation Computer Systems*, 100:779–796, 2019.
- [24] Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C. Suh, Ikkyun Kim, and Kuinam J. Kim. A Survey of Deep Learning-based Network Anomaly Detection. *Cluster Computing*, 22(1):949–961, January 2019.
- [25] Joffrey L. Leevy, Taghi M. Khoshgoftaar, Richard A. Bauder, and Naeem Seliya. A Survey on Addressing High-Class Imbalance in Big Data. *Journal of Big Data*, 5(1):42, November 2018.

- [26] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. Intrusion Detection System: A Comprehensive Review. *Journal of Network and Computer Applications*, 36(1):16–24, January 2013.
- [27] Nour Moustafa and Jill Slay. UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set). In *Proc. of 2015 Military Communications and Information Systems Conference (MilCIS)*, pages 1–6, November 2015.
- [28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
- [29] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.
- [30] John Platt. Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. Technical Report MSR-TR-98-14, Microsoft, April 1998.
- [31] Satish Pokhrel, Robert Abbas, and Bhulok Aryal. Iot security: Botnet detection in iot using machine learning, 2021.
- [32] Md Arafatur Rahman, A. Taufiq Asyhari, L. S. Leong, G. B. Satrya, M. Hai Tao, and M. F. Zolkipli. Scalable Machine Learning-based Intrusion Detection System for IoT-enabled Smart Cities. *Sustainable Cities and Society*, 61:102324, October 2020.
- [33] Dukka Karun Kumar Reddy, H. S. Behera, Janmenjoy Nayak, Bighnaraj Naik, Uttam Ghosh, and Pradip Kumar Sharma. Exact Greedy Algorithm based Split Finding Approach for Intrusion Detection in Fog-enabled IoT

- Environment. *Journal of Information Security and Applications*, 60:102866, August 2021.
- [34] Raúl Rojas. *Neural Networks: A Systematic Introduction*. Springer Science & Business Media, 2013.
 - [35] Brian C. Ross. Mutual Information between Discrete and Continuous Data Sets. *PLOS ONE*, 9(2):e87357, February 2014.
 - [36] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edition, 2009.
 - [37] Nausheen Sahar, Ratnesh Mishra, and Sidra Kalam. Deep Learning Approach-Based Network Intrusion Detection System for Fog-Assisted IoT. In Shailesh Tiwari, Erma Suryani, Andrew Keong Ng, K. K. Mishra, and Nitin Singh, editors, *Proc. of International Conference on Big Data, Machine Learning and Their Applications*, Lecture Notes in Networks and Systems, pages 39–50, Singapore, 2021. Springer.
 - [38] Iman Sharafaldin., Arash Habibi Lashkari., and Ali A. Ghorbani. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proc. of the 4th International Conference on Information Systems Security and Privacy - ICISSP*., pages 108–116. INSTICC, SciTePress, 2018.
 - [39] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, and Ali A. Ghorbani. Toward Developing a Systematic Approach to Generate Benchmark Datasets for Intrusion Detection. *Computers & Security*, 31(3):357–374, May 2012.
 - [40] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A Detailed Analysis of the KDD CUP 99 Data Set. In *Proc. of 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pages 1–6, July 2009.
 - [41] Ankit Thakkar and Ritika Lohiya. A Review on Machine Learning and Deep Learning Perspectives of IDS for IoT: Recent Updates, Security Issues, and Challenges. *Archives of Computational Methods in Engineering*, 28(4):3211–3243, June 2021.

- [42] Jason Van Hulse, Taghi M. Khoshgoftaar, and Amri Napolitano. Experimental Perspectives on Learning from Imbalanced Data. In *Proc. of the 24th International Conference on Machine Learning - ICML '07*, pages 935–942, Corvalis, Oregon, 2007. ACM Press.
- [43] Sanjay Yadav and Sanyam Shukla. Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification. In *Proc. of 2016 IEEE 6th International Conference on Advanced Computing (IACC)*, pages 78–83, February 2016.
- [44] Jerry Ye, Jyh-Herng Chow, Jiang Chen, and Zhaohui Zheng. Stochastic Gradient Boosted Distributed Decision Trees. In *Proc. of the 18th ACM Conference on Information and Knowledge Management, CIKM '09*, page 2061–2064, New York, NY, USA, 2009. Association for Computing Machinery.

Publication List

1. Jesse Atuhurra, Takanori Hara, Yuanyu Zhang and Shoji Kasahara, “On Attack Pattern Classification in IoT Networks for Network Intrusion Detection Systems,” IEICE Tech. Rep., Nov. 2021.