Received: March 3, 2024. Revised: March 25, 2024.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A New Approach of Botnet Activity Detection Models Using Combination of Univariate and ANOVA Feature Selection Techniques

Dandy Pramana Hostiadi[1]* Tohari Ahmad[2] Muhammad Aidiel Rachman Putra[2]
Gede Angga Pradipta[1] Putu Desiana Wulaning Ayu[1] Made Liandana[3]

[1]*Department of Magister Information Systems, Instittut Teknologi dan Bisnis STIKOM Bali, Bali, Indonesia*
[2]*Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia*
[3]*Department of Informatics and Computer, Institut Teknologi dan Bisnis STIKOM Bali, Bali, Indonesia*
* Corresponding author's Email: dandy@stikom-bali.ac.id

Abstract: The number of cases in the cyber era has increased significantly, which are caused by malicious software known as malware. This malicious software penetrates the network, infects several computers, and forms a collection of zombie computer networks commonly known as Botnets. These botnet threats can gravely impact valuable system resources and stored data and cause severe financial losses if not handled appropriately. Several previous studies introduced a botnet detection model using algorithms from machine learning by optimizing the feature selection process and having high detection results. However, feature selection is carried out without determining the role of features in the mandatory and non-mandatory categories. In fact, not all features can be selected because they have an important role and influence detection performance. This paper proposes a detection model by optimizing feature selection techniques. The initial process is to categorize features into mandatory and non-mandatory features. The feature selection process is carried out on non-mandatory features using two approaches: Univariate and ANOVA. Then, the best features from the feature selection results are aggregated with the Mandatory features and processed in a classification model for detecting malware attacks. The aim is to obtain the best features used in the classification model to improve detection performance by measuring accuracy, precision, and recall. The classification model used is a Decision tree and was tested on three different datasets, namely CTU-13, NCC, and NCC-2. The experiment result obtained an accuracy of 99.27% on the CTU-13 dataset, 98.96% on the NCC dataset, and 98.87% on the NCC-2 dataset. The resulting average precision value is 98.68% in the CTU-13 dataset, 98.26% in the NCC dataset, and 97.90% in the NCC-2 dataset. Finally, the resulting average recall value was 99.27% on the CTU-13 dataset, 98.96% on the NCC dataset, and 98.87% on the NCC-2 dataset. The detection results showed better results than previous research. This model can make analyzing attacks easier and determine treatment when a malware attack occurs.

Keywords: Feature selection, Botnet detection, Botnet flows, Network security, Network infrastructure.

1. Introduction

Handling attacks and threats in the cyber era requires serious treatment [1, 2]. So, security techniques in the form of IDS are needed [3]. Some IDS have challenges detecting malicious activity involving illegal software and pose a serious threat. Some well-known and dangerous threats include Trojan horses, phishing, viruses and worms, spyware, malware, botnets, zero-hours, hacker attacks, and others [4]. Botnets are among the most well-known threats that give rise to harmful attacks such as Distributed Denial of Service (DDoS), spamming, phishing, identity theft, and personal information theft [5, 6]. Malicious activities carried out by botnets tend to severely impact valuable system resources and stored data, leading to substantial financial losses [7].

One characteristic that distinguishes botnets from other types of attacks is their communication structure, which consists of a client and a botmaster [1, 8, 9]. Botmaster is a bot that gives instructions in

the form of commands to the bot client in carrying out new attacks. Meanwhile, the controls and instructions provided by the botmaster are executed by the bot client [10–12]. Along with the latest development, the characteristics of the botnet activities have evolved from centralized to decentralized [8]. Thus, some botnet activity detection models, such as Intrusion Detection Systems (IDS) or other types of antivirus, are struggling to provide precise and accurate detection results [13].

Previous research has proposed a machine learning classification approach, anomaly-based signature-based, and DNS-based machine learning classification approaches for botnet activity detection [14, 15]. Two important things that need to be considered in machine learning are feature selection and algorithm selection [4, 12, 16, 17]. The selection of features in the botnet detection model is required to improve detection accuracy. Additionally, feature selection can prevent overfitting in the data because it eliminates unnecessary data and features [18]. In contrast, the selection of algorithms in the detection model is needed to get time efficiency in the computational process in the botnet activity detection model. The development of feature selection techniques in botnet detection models has been used in previous studies. Alshamkhany et al. [19] used Principal Component Analysis (PCA) and chi-square methods on the UNSW dataset. However, attack activity was explicitly detected in DDoS attacks. The attack activities carried out by botnets could vary, such as DNS queries, spamming, and click fraud. Hossain et al. [20] introduce the wrapper method for their feature selection on a botnet model detection based on machine learning methods. In [21] introduced a botnet detection model by developing a feature selection mechanism in the filter method, which measures the feature subset with the highest predictive value from each feature vector. The experiment of [22] performed manual feature selection to determine features based on feature characteristics as numerical values. Several detection approaches proposed in the previous studies have not been optimal in feature selection technique. They still involve numerical and categorical data types On the feature selection based on the characteristics of the network traffic flow. In [23] introduce the botnet detection model by extracting attack characteristics and stages. To extract the botnet behavior, the feature engineering process uses one-hot encoding. It increases the number of existing features by transforming the values of several categorical attributes into numerical ones. However, each feature is still considered to have the same role. In fact, several essential or mandatory features cannot be selected. So, feature selection errors can affect the performance and accuracy of detection in the botnet detection model.

This study proposes a new approach to detect botnet attacks by optimizing feature selection techniques in the Decision Tree classification model. Feature selection begins by categorizing features into mandatory and non-mandatory. Feature selection is only based on features in the non-mandatory category previously carried out in the feature engineering process. The feature selection technique uses two techniques, namely Univariate and ANOVA, and the feature selection results are aggregated with mandatory features to be processed in the classification model. The proposed model aims to obtain the best features used in the classification model to improve detection performance. The research contributes to developing a new approach to feature selection for non-mandatory category features through closeness analysis and data distribution using the Univariate method and obtaining the best features selected using the ANOVA method. In this paper, the proposed model is tested on three different botnet attack datasets to see the model's performance in detecting malware attacks, namely CTU-13, NCC, and NCC-2 datasets. This model can be used to help administrators handle or adopt policies against attacks that occur. In addition, the proposed model can be developed to improve detection mechanisms in intrusion detection models or antivirus applications.

This paper is divided into several parts. Section 2 describes previous research that is related to the proposed research. The methodology is explained in Section 3. The results and analysis of the study are presented in Section 4. Finally, Section 5 offers conclusions and plans for future research direction.

2. Related work

In the previous research, the botnet activity detection models used classification [1, 8, 18, 24], clustering [25–28], and similarity. Research by [10, 16] used the CTU-13 dataset by combining feature selection methods. On the other hand, [22] and [30] opted to perform manual feature selection. Several other works reported that the other commonly used detection models are classification-based algorithms such as Naïve Bayes [31], Support Vector Machine [19], and Decision Tree [32].

Alshamkhany et al. [19] introduced the botnet detection model and optimized feature selection by combining two methods, Principal Component Analysis (PCA) and Chi-square method on the UNSW dataset to reduce data features. Meanwhile, the proposed feature selection method combines Chi-square and ANOVA F-Value to determine features' correlation strength with labels to obtain the best features. The features resulted from the selection process are: *Spkts, Rate, Dbytes, Sbytes, Dpkts, sttl, Sload, Dload, dloss, Sintpkt, Dintpkt, Sjit, stcpd, Djit, swin, dwin, dtcpb, dmean, res_bdy_an, ct_dst_ltm, ct_dst_src_ltm, ct_src_ltm, ct_dst_sport_ltm, ct_src_dport_ltm, cat*. The classification models that use algorithms such as the Decision Tree, Naïve Bayes, k-NN, and SVM-brf produced accuracy levels of 100%, 96.39%, 82.23%, and 99.00%, respectively. However, this research only processed the relevant features to increase computational efficiency. Computational efficiency is carried out in the evaluation process using the 5-fold cross-validation technique, and the best combination of hyperparameters is obtained in each classification model.

Hossain et al. [20] implemented the wrapper selection feature method to eliminate unnecessary and redundant features in the classification process. This method evaluates all possible combinations of features and selects the combination to obtain the best features. The combination is done with 2^n, where n is the number of features. However, performing the combination will take a long time to compute, especially in large datasets. The proposed method produces several selected features, namely: *'Duration', 'AvgDur', 'PBS', 'TBS','PBR', 'AvgPBR', 'TBR', 'PktSent', 'PktRcvd',* and *'SRPR'* with the ANN classification method which produces an accuracy value of 91%, and precision value of 88%. Pektas and Acarman [30] performed a feature selection method using three selection methods, namely: Recursive Feature Elimination (RFE), Linear models penalized (Lasso), and Tree-based feature (Ranking using random forest). However, these selection features mechanisms have a high discriminative ability to differentiate anomaly from normal traffic on the ISOT dataset. The feature selection begins with extracting a feature that exists with a flow extraction Tranalyzer. As a result of the feature selection process, network flow is represented as a feature matrix and class labels that indicate flow, botnet, or normal categories. Of the three classification models used, the Tree-based feature produces the highest accuracy, and there are nine selected features, namely: *Duration, numBytesRcvd, minPktSz, maxPktSz, avePktSize, stdPktSize, pktAsm, bytAsm and tcpMinWinSz* and the resulting accuracy value of Lasso was 93.6%, RFE was 94.3% and Tree-based was 99.5%.

In the work of Khan et al. [29], they implemented a decision tree that was used as a feature selection on the CTU-13 dataset. The model used is a wrapper that selects a subset of features. After the subset is generated, it will be processed in the evaluation model. The three evaluation methods used are Naïve Bayes, ANN, and Decision Tree, which produced accuracy levels of 75.50%, 93.80%, and 94.40% respectively. However, the training process in this study was carried out using network flows from various sources. The study performed by Hostiadi et al. [32] used the CTU-13 dataset that implemented a manual feature selection mechanism based on cosine similarity. This method used the cosine equation to measure the similarity of value for each features from each central node on different segments of the threshold value. The selected features based on this method are *Duration, Source IP, Source Port, Destination Port, Destination IP, Total Packets, Protocol, and Total Bytes*, which resulted in an accuracy level of 97.35% based on the number of IP bots detected in the bot chain. Mathur et al. [21] performed feature selection on the CTU-13 and ISOT datasets using the filter method. They measured the feature subset with the highest predictive value because it did not depend on any algorithm, and it was taken from the library, namely CfsSubsetEval to select the most relevant features from the experiment. The selected features according to the highest prediction level were: *td (flow duration), da (destination address),* and *pr (protocol)*. The detection process was carried out with five classification models, namely: the Multiclass Classifier, Logistics Regression, Random SubSpace, Random Committee and Randomizable Filtered Classifier, which produced an accuracy of 98.40%, 98.40%, 97.50, 95.3% and 97.70% respectively. However, this research could only process small datasets because large datasets would require a certain amount of processing power unavailable in personal computing systems. Segmentation techniques can be used to solve these issues, as implemented in [25].

Previous studies have introduced several approaches for feature selection in the botnet activity detection model, but they needed to be considered optimal. Besides, they have yet to deal with the feature selection problems involving categorical feature value characters. In fact, the categorical feature values can affect detection

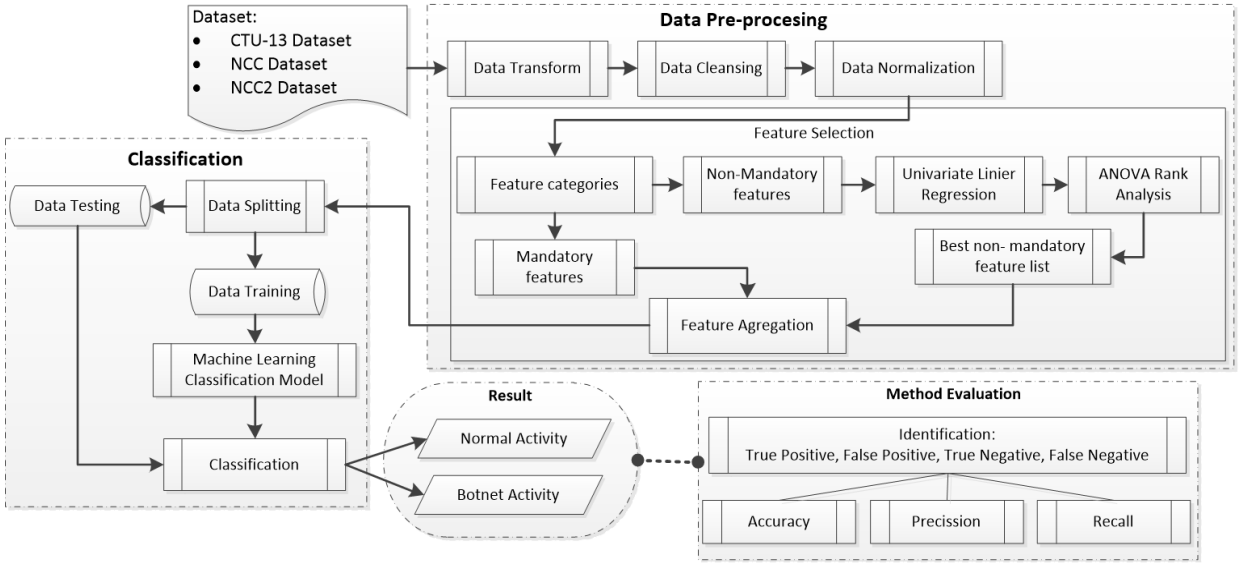


Figure. 1 Model Overview

performance, especially in measuring accuracy. Thus, the data transformation approach and the weighting of each feature were used as optimization methods in the feature selection mechanism for the botnet detection model proposed in this paper.

3. Methodology

This research focuses on developing a feature selection mechanism in the botnet activity detection model. Feature selection is part of the detection model that is important in optimizing botnet activity detection performance. The proposed model comprises three main components: pre-processing, classification, and evaluation. In this paper, the proposed detection model is shown in Fig. 1.

3.1 Preprocessing

The pre-processing stage is the initial stage in developing a botnet activity detection model. At the data transformation stage, categorical feature values are converted into numeric form. The data transformation approach used in this study is one-hot-encode [23, 33]. Botnet dataset (DT) is a collection of network traffic having several features (k) which can be formulated with. The features in the botnet dataset (k_n) have different types, namely: categorical and numeric. Thus, it can be formulated with $DT = \{kc_1, kc_2, \dots, kc_r, kn_1, kn_2, \dots, kn_s\}$. Where $\{kc_1, kc_2, \dots, kc_r\}$ is categorical data and $\{kn_1, kn_2, \dots, kn_s\}$ is numeric data. The number of categorical data with numerical is initialized with r and s so $r + s = n$. An example of a data transformation using one-hot-encode on a State feature is to form a new feature such as State_PA and State_SPA_PA. The IP address in the form of "147.32.84.193" is transformed into 32-Bit Binary and converted into an integer with the value of "2468369601". After the data transformation process, it continues with the data cleansing process. The data cleansing process is the process of removing unnecessary features. Three features are removed in the botnet detection process: sTos, dTos, and StartTime. The sTos and dTos features have relatively high null values, affecting the accuracy. Therefore, these features must be removed [12]. The StartTime feature is one of the features used in previous research to determine the link between activities or activity chains of botnet attacks [1, 25].

In this paper, the proposed model does not focus on analyzing time relationships and activity correlations, so the StartTime feature is not used in subsequent processes in the data normalization process. Normalization data is used to overcome the scale differences in each feature's values [34]. If left unchecked, this problem will affect the detection process with the classification algorithm [35]. Thus, it is necessary to normalize the data by rescaling it using the min-max scaler method by changing it into specific ranges of values [36]. Each value in a feature is reduced by the minimum value of the feature, then divided by the range of values or the maximum value minus the minimum value of the feature [34].

$$x_{new} = \frac{x_{old} - x_{min}}{x_{max} - x_{min}} \quad (1)$$

where, the x_{new} is the new value for the feature resulting from the difference between the value of x_{old} and x_{min}, which then divided by the sum of x_{max} and x_{min}.

In this paper, the feature selection method uses the univariate regression-ANOVA method. Feature selection is a technique that aims to eliminate features in a training set to eliminate features considered redundant in the classification process [17]. Feature selection also optimizes a subset of features from the original feature according to specific criteria [11]. The first phase in feature selection is feature categorization. In this stage, features are divided into mandatory and non-mandatory features. Mandatory features are not selected and contain essential information such as SrcAddr, DstAddr, Sport, Dport, and target features as labels. If these features are removed, the botnet detection process will not occur and cannot be detected accurately. Then, the non-mandatory features will be processed in the feature selection [21]. The examples of features included in the selection process are Dur, dTos, Dir, State, TotPkts, sTos, Proto, SrcBytes and TotBytes. After the feature categorization stage, the following process measures the relationship between non-mandatory features using univariate regression. The goal is to get the influence or closeness of each feature based on the distribution of data values. The univariate regression measurement process is shown in algorithm 1.

Algorithm 1. Univariate Linier Regression

INPUT: X, Y
OUTPUT: $b1; b0$
$b1$: slope
$b0$: intercept
X : matrix of input data ($m \times n$)
Y : vector of target feature ($n \times 1$)
n : number of columns in matrix X
m : number of rows in matrix X
i : index of row in loop
j : index of column in loop
x_{ij} : value of data x with j column and i row in matrix
$sumX$: sum of all X data
$sumX2$: sum of all $X \times X$ data
$sumY$: sum of all Y data
$sumXY$: sum of all $X \times Y$ data

Step 1: calculate sum
$sumX \leftarrow 0$
$sumX2 \leftarrow 0$
$sumY \leftarrow 0$
$sumY2 \leftarrow 0$
for $j \leftarrow 1$ to n **do**
$sumX \leftarrow sumX + X_m$
$sumX2 \leftarrow sumX2 + X_m * X_m$
$sumY \leftarrow sumY + Y_m$
$sumXY \leftarrow sumXY + X_m * Y_m$
end for
Step 2: calculate slope (b1) and intercept (b0)
$b1 \leftarrow (n * sumXY - sumX * sumY) / (i * sumX2 - sumX^2)$
$b0 \leftarrow (sumY - b1 * sumX) / i$
Step 3: return $b1; b0$

Then, using the univariate regression method, every feature value that has a value above 0 will be selected based on all non-mandatory features using ANOVA to get the best ones. Most machine-learning algorithms require data to be stored in a two-dimensional array or matrix [37]. The matrix size contains [sample, feature] [38]. The first parameter (X) is the input data/training data matrix, where m is the row, n is the column, and the second parameter (Y) is the target data. Then, both parameters are used to calculate ANOVA [39].

$$Feature\ matrix\ X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ x_{31} & x_{32} & \dots & x_{3n} \\ \dots & \dots & \dots & \dots \\ x_{m1} & x_{m2} & \dots & x_{mn} \end{bmatrix} \quad (2)$$

$$Label\ vector\ Y = [y_1, y_2, y_3, \dots, y_m] \quad (3)$$

One of the main processes in feature selection is univariate feature selection with ANOVA F-test for scoring each feature. This process analyzes each feature to determine how the feature is related to the label, measures the similarity between relevant features, reduces feature dimensions, and increases accuracy in the botnet detection process [39]. Each feature is scored and ranked. The calculation obtained using ANOVA is F. The higher the F value, the more separate the labels are. The score of each feature by calculating the distance between classes can be calculated with Equation 4:

$$\sigma_{cl}^2 = \frac{\sum(\bar{x}_o - \bar{x})^2 a_i}{(k-l)} \quad (4)$$

where a_o is the number of classes o occurrences in the set, $\bar{x}_o$ is the mean of class o, and $\bar{x}$ is the average of the features. Then, calculate the distance between classes denoted as Equation 5:

$$\sigma_{err}^2 = \frac{\left(\sum\sum\left(x_{op} - \bar{x}\right)^2\right) - \left(\sum(\bar{x}_o - \bar{x})^2 a_o\right)}{(k-l)} \quad (5)$$

The sum of the squared values per class on the feature is reduced by the average of the features, then reduced by the sum of the results of the square of the class average minus the feature average is denoted in Equation 6:

$$F = \frac{\sigma_{cl}^2}{\sigma_{err}^2} \quad (6)$$

The ANOVA calculation will produce a f_value that will be used as a parameter for calculating the select percentile process. The ANOVA calculation generates a set of f_values for each feature. The features in K are denoted as k_j where $j = 1,2,3,\ldots n$, where n is the number of features in K. Besides, the results of the ANOVA calculation are collected in a vector if denoted as $Fval$, which contains a value that represents each feature number of n, then it is denoted as Equation 7:

$$Fval = [\, kval_1, kval_2, kval_3, \ldots, kval_n]; \quad (7)$$
$$j = \{1,2,3,\ldots n\}$$

The ANOVA calculation process is shown in algorithm 2.

Algorithm 2. ANOVA

INPUT: X, Y
OUTPUT: $Fval$

$Fval$	: collection of f_values
X	: matrix of input data ($m \times n$)
Y	: vector of target feature ($n \times 1$)
n	: number of columns in matrix X
m	: number of rows in matrix X
i	: index of row in loop
j	: index of column in loop
x_{ij}	: value of data x with j column and i row in matrix
$f\mu$	: feature mean value
$y\mu$	: target feature mean value
$bVar$	: between group variance
$wVar$	: within group variance
$sumY$	: variable for summation of all elements in Y
sum	: variable for summation of all elements in feature
bdf	: between group degree of freedom
wdf	: within group degree of freedom
bms	: between group mean square
wms	: within group mean square
$fval$	: feature f-value

Step 1: loop all columns in matrix X
$Fval \leftarrow []$
for $j \leftarrow 1$ to n **do**
Step 2: calculate total and group means
$sumX \leftarrow 0$
$sum \leftarrow 0$
for $i \leftarrow 1$ to m **do**
$sumY \leftarrow sumY + Y_{ji}$
$sum \leftarrow sum + x_{ji}$
end for
$y\mu \leftarrow sumY/m$
$f\mu \leftarrow sum/m$
Step 3: calculate the between and within group variance
$bVar \leftarrow (f\mu - y\mu)^2$
$wVar \leftarrow 0$
for $i \leftarrow 1$ to m **do**
$wVar \leftarrow wVar + \left((y_{ji} - f\mu)^2\right)$
end for
Step 4: calculate degrees of freedom and mean squares values
$bdf \leftarrow 1$
$wdf \leftarrow m - 1$
$bms \leftarrow bVar/bdf$
$wms \leftarrow wVar/wdf$
$fval \leftarrow bms/wms$
$Fval_i \leftarrow fval$
end for
Step 5: return $Fval$

The results of the ANOVA selection produce the best features, which will be processed in the feature aggregation stage, combining the mandatory features with the features selected using ANOVA. The results of this combination are used in the Classification process. The preprocessing process from the data transformation, data cleansing, and data normalization stages, as well as the feature selection stage, are shown in Algorithm 3.

Algorithm 3. Pre-processing & Feature Selection

INPUT: F
OUTPUT: selected F

F	: set of dataset features $\{f_1, f_2, \ldots, f_j\}$
f_j	: feature in dataset with index = j
n	: number of features in dataset
m	: number of rows
j	: index of feature loop
x_{ji}	: value of data f_j in i row
$select$	: total feature allowed for next step ($i \times 50\%$)
$FSelect$	: set of selected features $\{f_1, f_2, \ldots, f_{select}\}$
$Fmin_j$	: minimum value in f_j
$Fmax_j$	: maximum value in f_j

Step 1: data Transform
do manual categorical to numeric data transform
do one-hot encode
Step 2: data Cleansing

for $j \leftarrow 1$ to n **do**
if $f_j = sTos$ **or** $f_j = dTos$ **or** $f_j = StartTime$ **do**
do skip to next feature
else
go to step 3
end for
Step 3: data Normalization
for $j \leftarrow 1$ to n **do**
set $Fmin_j$
set $Fmax_j$
for $i \leftarrow 1$ to m **do**
$x_{ji} \leftarrow \frac{(x_{ji} - Fmin_j)}{(Fmax_j - Fmin_j)}$
end for
end for
Step 4: check mandatory feature
for $j \leftarrow 1$ to n **do**
if $f_j = SrcPort$ **or** $f_j = SrcAddr$ **or** $f_j = DstPort$ **or** $f_j = DstAddr$ **do**
go to step 8
Else
go to step 5
end for
Step 5: Univariate Linier Regression
do Algorithm 2. Univariate Linier Regression
Step 6: ANOVA
do Algorithm 3. ANOVA
Step 7: Best Non-Mandatory Feature Analysis
do Sort feature based on ANOVA (high to low)
for $j \leftarrow 1$ *to select* **do**
$FSelect_j \leftarrow f_j$
end for
Step 8: Feature Aggregation
$FSelect_{select+1} \leftarrow SrcAddr$
$FSelect_{select+2} \leftarrow SrcPort$
$FSelect_{select+3} \leftarrow DstAddr$
$FSelect_{select+4} \leftarrow DstPort$

3.2 Classification

In this research, the classification method used is the Decision Tree algorithm. The classification process begins with the data splitting process, which divides the dataset into two parts: the training and test sets. The training set contains the training data used to build the classification model, while the test set is the test data. Data distribution is done by 70% of each class in the dataset, allocated as training data, and the remaining 30% as test data. The division's results produce four new data frames, namely bot_test, bot_train, normal_test, and normal_train. Then, the four datasets are grouped into test data containing bot_test, normal_test, and training data containing bot_train and normal_train. The decision tree is a classification and predictive model using a hierarchical structure [40]. The decision tree is useful for exploring data and finding the relationship between input and target variables. Both methods are used to calculate the value of impurity or heterogeneity of data, namely the Gini Index in Equation 8 and Entropy in Equation 9.

$$Gini = 1 - \sum_{j=1}^{C} (P_j)^2 \quad (8)$$

$$Entropy = \sum_{j=1}^{C} - P_j * log_2(P_j) \quad (9)$$

where j is the iteration number, c is the number of data and P_j is the probability of feature classification.

3.3 Evaluation

In this paper, the evaluation of activity detection from the decision tree classification method is carried out by measuring the accuracy (Acc.), precision (Pre.), and recall (Rec.) values, by Equation 10, 11 and 12.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (11)$$

$$Precision = \frac{TP}{TP + FP} \quad (12)$$

$$Recall = \frac{TP}{TP + FN} \quad (13)$$

Evaluation of the classification model is based on four values: true positive, false negative, false positive, and true negative. True positive (TP) represents the number of botnet attack data that was successfully predicted as an attack. The botnet attack data predicted as normal activity is called False negative (FN). True negative (TN) is the number of normal activity successfully predicted as normal. Meanwhile, a False positive (FP) represents normal data which is predicted as botnet attack data.

4. Result and discussion

This study aims to detect botnet activity based on network traffic flow analysis. The detection system being executed used a laptop with 2.5 GHz dual-core Intel Core i7 processor, 16 GB RAM, and 1 Terabyte SSD storage space, and using Python Programming. In this study, three public datasets, namely CTU- 13 [41], NCC [42] and NCC-2 [43]were used to test the model.

Table 1. CTU-13 Dataset Description [41]

Scenario	Duration (hours)	Number of Bots	Botnet Flows	Normal Flows	C&C Flows	Background Flows	Total Flows
1	6.15	1	39,933 (1.41%)	30,387(1.07%)	1,026(0.03%)	2,753,290(97.47%)	2,824,636
2	4.21	1	18,839(1.04%)	9,120(0.5%)	2,102(0.11%)	1,778,061(98.33%)	1,808,122
3	66.85	1	26,759(0.56%)	116,887(2.48%)	63(0.001%)	4,566,929(96.94%)	4,710,638
4	4.21	1	1,719(0.15%)	25,268(2.25%)	49(0.004%)	1,094,040(97.58%)	1,121,076
5	11.63	1	695(0.53%)	4,679(3.6%)	206(1.15%)	124,252(95.7%)	129,832
6	2.18	1	4,431(0.79%)	7,494(1.34%)	199(0.03%)	546,795(97.83%)	558,919
7	0.38	1	37(0.03%)	1,677(1.47%)	26(0.02%)	112,337(98.47%)	114,077
8	19.5	1	5,052(0.17%)	72,822(2.46%)	1,074(2.4%)	2,875,282(97.32%)	2,954,230
9	5.18	10	179,880(6.5%)	43,340(1.57%)	5,099(0.18%)	2,525,565(91.7%)	2,753,884
10	4.75	10	106,315(8.11%)	15,847(1.2%)	37(0.002%)	1,187,592(90.67%)	1,309,791
11	0.26	3	8,161(7.6%)	2,718(2.53%)	3(0.002%)	96,369(89.85%)	107,251
12	1.21	3	2,143(0.65%)	7,628(2.34%)	25(0.007%)	315,675(96.99%)	325,471
13	16.36	1	38,791(2.01%)	31,939(1.65%)	1,202(0.06%)	1,853,217(96.26%)	1,925,149

Table 2. NCC Dataset Description [42]

Scenario	Duration (hours)	Number of Bots	*Bot* flows	Normal host	Normal Flows	Total flows
1	8 hours	1	23,000 (1.09%)	342,740	2,089,224 (98.91%)	2,112,224
2	8 hours	1	24,000 (1.64%)	252,263	1,441,182 (98.36%)	1,465,182
3	8 hours	1	2,000 (0.07%)	240,780	2,903,611 (99.93%)	2,905,611
4	8 hours	1	11,000 (1,52%)	66,013	713,388 (98.48%)	724,388
5	8 hours	1	19,000 (20.45%)	10,346	73,917 (79.55%)	92,917
6	8 hours	1	6,000 (1.17%)	46,627	506,021 (98.83%)	512,021
7	8 hours	1	9,000 (10,78%)	9,598	74,473 (89.22%)	83,473
8	8 hours	1	14,000 (0.49%)	252,162	2,857,217 (99.51%)	2,871,217
9	8 hours	10	220,000 (13.98%)	180,554	1,353,304 (86.02%)	1,573,304
10	8 hours	10	60,000 (6.10%)	89,915	924,369 (93.90%)	984,369
11	8 hours	3	120,000 (38.75%)	3,729	18,964 (61.25%)	30,964
12	8 hours	3	9,000 (3.28%)	33,613	265,186 (96.72%)	274,186
13	8 hours	1	19,000 (1.01%)	209,865	1,857,489 (98.99%)	1,876,489

Table 3. NCC-2 Dataset Description [43]

Sensors	Duration (hours)	Number of Bots	Bot Activity	Normal Activity	Total Activity
1	8 hours	5	146,000	4,749,158	4,895,158
2	8 hours	4	364,000	5,634,133	5,998,133
3	8 hours	5	294,000	3,591,792	3,885,792

Table 4. Original Feature on CTU-13, NCC and NCC-2 Dataset

Dataset	Number	Original Feature
CTU-13	14	SrcAddr, sTos, dTos, Dur, TotBytes, Sport, DstAddr, StartTime, Dport, State, TotPkts, Dir, SrcBytes, Proto
NCC	14	sTos, TotBytes, Dur, Sport, StartTime, Dport, DstAddr, State, TotPkts, Dir, SrcBytes, Proto, dTos, SrcAddr,
NCC-2	17	SrcAddr, sTos, dTos, Dur, BotnetName, TotBytes, Sport, DstAddr, ActivityLabel, StartTime, Dport, State, TotPkts, Dir, SensorId, SrcBytes, Proto

Table 5. Data Transform Results on CTU-13, NCC and NCC-2 Dataset

Dataset	Number	Feature after Data Transforming	
		Mandatory Feature	Non-Mandatory Feature
CTU-13	20	Srcport, SrcAddr, DstAddr, Dport	SrcAddr, Dir_?>, sTos, Dir_->, DstAddr, Dir_who, Dir_<?, Dport, Dir_<->, TotBytes, Proto, State, Sport, Dir_<-, Dur, StartTime, dTos, TotPkts, SrcBytes, Dir_<?>
NCC	20	DstAddr, Dport, Srcport, SrcAddr	State, Dir_->, sTos, Dir_who, Dir_<?>, Dira_<?, Dport, Dir_<->, TotBytes, SrcAddr, Dir_?>, Proto, Sport, Dir_<-, Dur, StartTime, dTos, TotPkts, SrcBytes, DstAddr
NCC-2	23	Dport, SrcAddr, DstAddr, Srcport	SrcAddr, Dir_?>, sTos, Dir_->, DstAddr, Dir_who, Dir_<?, ActivityLabel, Dport, Dir_<->, TotBytes, Proto, BotnetName, State, Sport, Dir_<-, Dur, StartTime, dTos, TotPkts, SensorId, SrcBytes, Dir_<?>

4.1 Dataset description

The CTU-13 dataset contains 13 scenarios captured in the actual network environment, including botnets, normal traffic, and background traffic [41]. The duration of recorded NetFlow data varies from 0.26 to 66.85 hours, and then the amount of NetFlow data also varies. Most scenarios have only one Bot (1-8 and 13), whereas scenarios 9-12 have multiple bots. The description of the CTU-13 dataset is shown in the Table 1.

NCC Dataset is a group botnet activity data presented as a bidirectional flow file [42]. In the NCC dataset, periodic means that the pattern of bot activity appears across multiple segments, while intensity means that bot activity exists in each segment. These periodic and intensity characteristics are suitable for evaluating bot group detection with a timed segmentation approach. The NCC dataset has 12,896,345 total flows, divided into Bot flows, Normal host, and Normal flows. The total number of Bot flows from 13 scenarios that ran for 8 hours was 536,000 (90.33%), and the total number of normal hosts from 13 scenarios that ran for 8 hours was 1,720,205. The description of the NCC dataset is shown in the Table 2.

The NCC-2 dataset is data on simultaneous botnet attack activities [43]. Simultaneous characteristics mean that attack activity occurs at the same time and is captured on several detection sensors. The NCC-2 dataset has three types of sub-datasets with the names Sensor Id-1, Sensor Id-2, and Sensor Id-3, with the types of botnet malware being Rbot, Neris, Sogo, NSIS.ay, and Virut. The three sub-datasets were recorded for 8 hours, and the composition of botnet attack activity was around 146,000 to 294,000 or 2,983% to 7,566%. Meanwhile, normal activity ranges from 4,749,758 – 3,591,792 or 97,017% to 92,434%. This dataset adopts the behavior of botnet attacks on the CTU-13 and NCC datasets. The description of the NCC-2 dataset is shown in the Table 3.

Each dataset has original features shown in Table 4.

4.2 Data transformation

The initial stage of pre-processing is data transformation. At this stage, the types of mandatory and non-mandatory features are transformed into numerical forms. Examples of data transformation on mandatory types such as *SrcAddr* and *DstAddr* to numeric. In addition, using the one hot encode technique produces several new features from the initial features which are the result of feature aggregation. In this paper the data transform technique using one hot encode in this paper was adopted in research [44]. A description of the results of the data transform stage on the CTU-13, NCC and NCC-2 dataset are shown in Table 5.

The data transformation results in the CTU-13 and NCC datasets have increased the number of features from 13 to 19. This addition is caused by the one hot encode technique, where the features generated are the values of the non-mandatory feature Dir into features Dir_<?>, Dir_ ->, Dir_<?, Dir_ <-, Dir_?> Dir_<->, Dir_ who. Meanwhile, in the NCC-2 dataset, there was an increase from 17 features to 23 features. The addition of features occurs due to the extraction of the Dir feature into features Dir_ ->, Dir_?>, Dir_ <-, Dir_<->, Dir_<?>, Dir_ who, Dir_<?. Extraction of features that do not exist in the NCC and CTU-13 datasets that only exist in the NCC-2 dataset is the Dir_<? Feature.

4.3 Data cleansing

The In the CTU-13 and NCC datasets, three featuresare removed in the botnet detection process: sTos, dTos, and StartTime. Meanwhile, in the NCC2 dataset, six features were deleted, namely StartTime, sTos, dTos, ActivityLabel, BotnetName, and SensorId. The features sTos and dTos were removed because they have relatively high null values, affecting the accuracy. Therefore, these features must be removed [12]. The StartTime feature is removed because the botnet detection process is

Table 6. Univariate Results on CTU-13 and NCC Dataset

Scenario	CTU-13 Dataset		NCC Dataset	
	Number	Non-Mandatory Feature	Number	Non-Mandatory Feature
1	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
2	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
3	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
4	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
5	12	Dir_<->, SrcBytes, Dir_ ->, Dir_ ?>, TotBytes, Dir_ <-, State, Dir_<?>, Dir_ who, Proto, Dir_<?, Dur	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
6	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
7	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
8	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
9	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
10	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
11	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
12	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.
13	13	State, Dir_<->, Dir_ ->, SrcBytes, Dir_ ?>, Dur, Dir_ <-, TotPkts, Proto, Dir_<?>, Dir_ who, Dir_<?, TotBytes.	13	SrcBytes, Dir_ ->, Dur, Dir_ <-, State, Dir_<?, TotBytes, Dir_ ?>, Dir_<->, Proto, Dir_<?>, Dir_ who, TotPkts.

Table 7. Univariate Results on NCC-2 Dataset

Sensor-ID	Number	Non-Mandatory Feature
1	13	Dir_<->, SrcBytes, Dir_<?, State, Dir_ ?>, TotPkts, Dir_<?>, TotBytes, Dir_ ->, Dir_ <-, Dur, Dir_ who, Proto
2	13	TotPkts, Dir_<->, Dir_<?>, SrcBytes, Dir_<?, State, Dir_ ?>, TotBytes, Dir_ ->, Dur, Dir_ who, Proto, Dir_ <-
3	13	TotBytes, Dir_ ->, TotPkts, Dir_<->, SrcBytes, Dir_<?, State, Dir_ ?>, Dur, Dir_ who, Proto, Dir_ <-Dir_<?>

Table 8. ANOVA Results on CTU-13 and NCC Dataset

Scenario ID/Sensor	CTU-13 Dataset			NCC Dataset		
	Number	Feature non-mandatory	Feature Reduction (%)	Number	Feature	Feature Reduction (%)
1	7	Proto, Dir_->, Dur, Dir_<, State, Dir_<->, Dir_<?>.	46.15	7	TotPkts, Dir_->, Dir_<->, Dir_<-, State, Proto, Dur	46.15
2	7	Dir_<->, Proto, Dir_<-, State, Dir_<?>, Dur, Dir_->	46.15	7	Dir_->, TotPkts, Dir_<->, Proto, Dir_<?>, State, Dur	46.15
3	7	Proto, Dir_->,Dir_<-, Dur, Dir_?>, State, Dir_<->.	46.15	7	Dir_<?>, Dur, Dir_<-, Dir->, State, Dir_<->, Proto	46.15
4	7	Dur, Dir_->,Dir_<-, Proto, Dir_<?>, State, Dir_<->.	46.15	7	State, Dir_<?>, Dur , Dir_<->, Dir_<-, Dir->, Proto	46.15
5	7	Dir_<?>, State, Dir_->, Dir_<-,Dur, Dir_<->, Proto.	41.67	7	SrcBytes,Dir->, Dir_<->,Proto,State,Dir_<- ,TotBytes	46.15
6	7	Dir_<?>, Proto, Dir_->, Dir_<-,State,Dur,Dir_<->.	46.15	7	Dur, Dir_->, State, Dir_<->, Proto, Dir_<-, Dir_<?>	46.15
7	7	Dur,Dir_->,Dir_<?>,State, Proto,Dir_<->, TotBytes.	46.15	7	State,Dir_->,Proto, TotBytes, Dir_<->, TotPkts, Dur	46.15
8	7	Dur,Dir_->,Dir_<->,State, Dir_<-,Proto, Dir_<?>.	46.15	7	Dir_->, State, Dir_<->, Proto, Dir_<-, Dir_<?>, Dur	46.15
9	7	Dir_<->,State,Dir_->, Dir_<?>,Proto,Dur,Dir_<-	46.15	7	Dir_?>, Dir_->, State, Dir_<->, Dir_<-, Proto, Dir_<?>,	46.15
10	7	Dir_->, Proto, Dir_<->, Dir_<?>,State,Dir_<-,Dur	46.15	7	Dur, Dir_->, State, SrcBytes, Dir_<->, TotPkts, TotBytes	46.15
11	7	Dir_<-,Dur,Dir_->,Dir_<->, Dir_<?>, State, Proto	46.15	7	Dur, Dir_->, State, Dir_<->, Proto, Dir_<-, Dir_?>	46.15
12	7	State,Dir_->, Dir_<->, Dir_<?>,Dir_?>,Dur,Dir_<-	46.15	7	Dir_<?>, Dur, Dir_<->, Proto, Dir_<-, Dir_->, State	46.15
13	7	Dir_->, State, Dir_<-, Proto,Dir_<?>,Dur,Dir_<->	46.15	7	State, Dir_<->, Dir_->, Dur, Dir_<?>, Proto, Dir_<-	46.15

Table 9. ANOVA Results on NCC-2

Scenario ID/Sensor	Number	Feature	Feature Reduction (%)
1	7	TotBytes, SrcBytes, Dir_->, Proto, Dir_<->, State, TotPkts	46.15
2	7	Dir_->,Dir_<->,State,Proto, SrcBytes,TotPkts,TotBytes	46.15
3	7	Dir_who, Dir_<->, State, Dir_<-, Proto, Dir_->, Dir_?>	46.15

Table 10. Classification Result

Dataset	Evaluation	Scenario / Sensor-ID												
		1	2	3	4	5	6	7	8	9	10	11	12	13
CTU-13	Acc. (%)	98.50	98.80	99.40	99.80	99.30	99.20	99.90	99.80	99.90	99.40	99.30	99.30	97.90
	Prec. (%)	97.10	97.70	98.90	99.50	98.60	98.40	99.90	99.60	99.90	99.40	99.30	98.70	95.90
	Rec. (%)	98.50	98.80	99.40	99.80	99.30	99.20	99.90	99.80	99.90	99.40	99.30	99.30	97.90
NCC	Acc. (%)	98.90	98.40	99.90	98.50	98.90	98.80	99.70	99.50	99.90	98.90	99.40	96.70	99.90
	Prec. (%)	97.80	96.80	99.90	97.00	98.90	97.70	99.70	99.90	99.90	98.90	99.40	93.50	98.00
	Rec. (%)	98.90	98.40	99.90	98.50	98.90	98.80	99.70	99.50	99.90	98.90	99.40	96.70	99.00
NCC-2	Acc. (%)	97.00	99.90	99.70										
	Prec. (%)	94.10	99.90	99.70										
	Rec. (%)	97.00	99.90	99.70										

unrelated to the time series. Besides, the model removed three features, namely ActivityLabel, BotnetName, and SensorId features, in the NCC-2 dataset because they are only labels and cannot be compared with the similarity of features in the CTU-13 and NCC datasets. Besides the cleaning features, data records are also cleaned. In the CTU-13 dataset, the data cleansing process handles the null value in three features: Dport, Sport, and State. In the Dport feature, there are 7,900; in Sport, there are 463 records that have null values or empty values. Besides, the state has 92 total records that have an empty value. In this condition, each empty value is filled with "0". In the NCC and NCC-2 dataset, the model did not find the null values. Thus, the model does not process the data cleansing. The data normalization process is equalizing each feature's value scale with a high range of values between features. In this study, data normalization was carried out on a scale ranging from 0 to 1 on all three datasets.

4.4 Feature selection

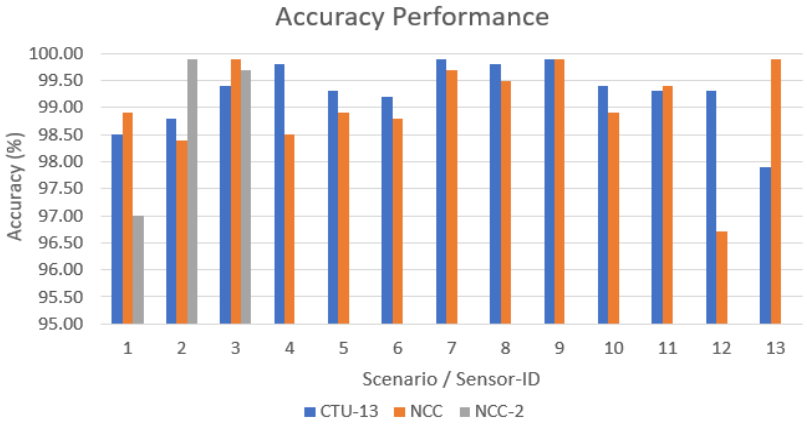
The The preprocessing stage begins with dividing features into mandatory and non-mandatory feature categories. Specifically for the non-mandatory category feature, a process of analyzing the closeness of the data distribution was carried out using univariate analysis. The results of univariate measurements on the CTU-13 and NCC datasets are shown in Table 6, and the NCC-2 datasets are shown in Table 7.

The univariate results show that all non-mandatory features have close data distribution relationships in each data scenario for the CTU-13 and NCC datasets and each sensor ID in the NCC-2 dataset. There are 13 features used in scenarios 1 to 4 and scenarios 6 to 13 in the CTU-13 dataset. In scenario 5, the model found only 12 features with close data distribution. One feature that is not used in scenario 5 in the CTU-13 dataset is the TotPkts feature. In the NCC dataset, 13 features are used in all scenario datasets. Meanwhile, in the NCC-2 dataset, there are 13 features produced by univariate for the sub-dataset in sensors 1, 2, and 3. Then, all features are selected using ANOVA. Selection with ANOVA features reduces the number of features used in the following process: the classification stage. The feature selection results on the CTU-13 and NCC dataset are shown in Table 8, and the NCC-2 dataset in Table 9.

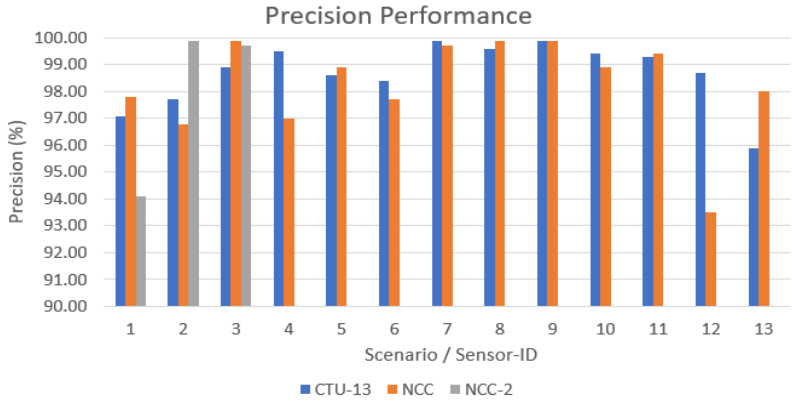
The results of ANOVA feature selection reduced the number of non-mandatory features by 41.67% to 46.15%. Feature selection using ANOVA produces the seven best features from the non-mandatory category. Then, the seven features are combined with the mandatory features to be used in the classification stage.

4.5 Classification result

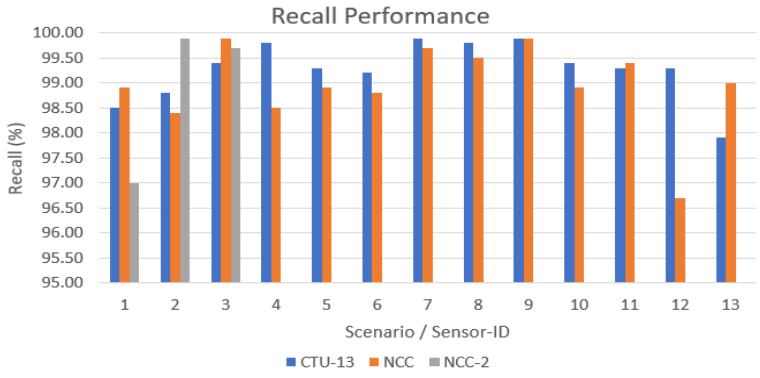
The feature aggregation process results in 11 features resulting from two categories, namely mandatory and non-mandatory features. Next, all data with 11 features is divided into traffic data with a composition of 70% as training data and 30% as testing data. In the classification process, a decision tree model is used. The classification results in the three datasets are shown in Table 10. The highest detection accuracy evaluation of the Decision Tree method is on the CTU-13 dataset, with an average detection accuracy value in each scenario of 99.27%. The highest detection accuracy value is in scenario seven and scenario 9 in the CTU-13 Dataset, with a value of 99.90%. In the NCC dataset, the highest accuracy value is in scenarios 3, 9, and 12, with a value of 99.90%. The comparison results of detection accuracy, precision, recall and time computation analysis are shown in Fig. 2.



(a)



(b)



(c)

Figure. 2 Model Evaluations: (a) Accuracy Analysis, (b) Precision Analysis, and (c) Recall Analysis

Meanwhile, the highest accuracy value in the NCC-2 dataset is sensor-ID two at 99.90%. Of the three datasets, the lowest accuracy value is in the NCC dataset in scenario 12, which is 96.70%. The lowest value in the CTU-13 dataset is 97.90% in scenario 13. Meanwhile, the lowest accuracy value in the NCC-2 dataset is 97% in Sensor 1 data. Besides, the evaluation is carried out by looking at the precision value. From Table 10, The highest precision value is 99.90% in scenarios 7 and 9 in the CTU-13 dataset, scenarios 3, 8, and 9 in the NCC dataset is 99.90% and sensor-ID 2 data in the NCC-2 dataset 99.90%. The highest detection average produced by the decision tree method was 98.68% on the CTU-13 dataset. The lowest precision value is on the NCC dataset at 97.90%.

The last evaluation of the classification results of the decision tree method is the recall measurement. The model obtained the highest recall value in the CTU-13 dataset, with an average of 99.27%. The highest recall value from the dataset is 99.90%, where on the CTU-13 dataset in scenarios 7 and 9, on the NCC dataset in scenarios 3 and 9, and on the

Table 11. Comparison Proposed Model and Previous Studies

Authors	Dataset	Feature selection method	Classification method	Accuracy (%)	Precision (%)	Recall (%)	Measuring computation time (s)
Khan et al. [29]	CTU-13	Wrapper Method	Naive Bayes	75.50	-	-	-
			ANN	93.80	-	-	
			Decision Tree	94.40	-	-	
Joshi, Ranjan and Bharti [45]	CTU-13	Fuzzy	ANN	99.94	99.92	99.96	-
Letteri, Penna and Caianiello [46]	CTU-13	Multilayer Perceptron (MLP)	Decision Tree	97.54	97.75	97.26	-
Mathur et al. [21]	CTU-13 and ISOT	CfsSubsetEval	Logistic Regression	98.40	-	-	-
			Random Subspace	97.50	-	-	
			Randomizable Filtered	97.70	-	-	
			Multiclass Classifier	98.40	-	-	
			Random Committee	95.30	-	-	
Putra, MAR et.al [23]	CTU-13	Manual	Random Forest	99.998	-	-	-
	NCC			99.999	-	-	-
Naseri, Abidin and Eslahi [47]	CTU-13	Manual	C4.5	98.20	98.20	98.20	-
			Random Forest	98.20	98.20	98.20	-
			Naïve Bayes	97.00	97.00	97.00	-
			Support Vector Machine	98.40	98.40	98.40	-
			Feedforward Neural Network (FNN)	98.50	98.50	98.50	-
Putra, MAR et al. [1]	CTU-13	Manual	Decision Tree	99.93	8.93	8.93	-
	NCC			99.99	61.54	80	
	NCC-2			100	70	70	
Proposed method	CTU-13	Univariate Analysis and ANOVA	Decision Tree	99.27	98.68	99.27	28.8523
	NCC			99.03	98.26	98.96	17.7299
	NCC-2			98.87	97.90	98.87	44.3419

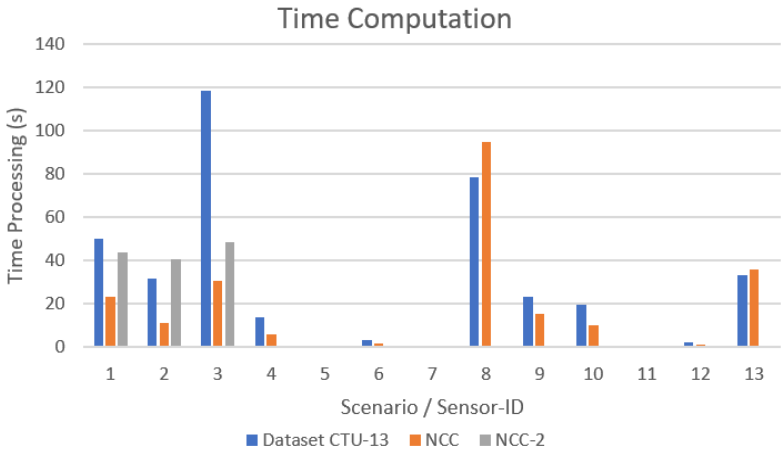


Figure. 3 Time Computation Analysis

NCC-2 dataset in sensor-ID 2 data. The lowest recall value is on the dataset NCC in scenario 12, with a value of 96.70%. Meanwhile, in the CTU-13 dataset, the lowest recall value was 97.90% in scenario 13, and in the NCC-2 dataset, the lowest recall value was 97% in sensor-ID 1 data.

In this paper, the model measures computing time to detect botnet attacks on three datasets with different processing times. The comparison results of time computation analysis are shown in Fig. 3.

In the CTU-13 dataset, the processing time from the start to the detection process using the decision tree method requires an average of 28.8523 seconds, with the lowest time in scenario seven and the highest processing time in scenario 3. Meanwhile, in the NCC dataset, the average process took 17.7299 seconds and was the lowest average time compared to the other two datasets. In the NCC dataset, the

highest processing time is in scenario 8, with a time of 95.1052 seconds, and the lowest processing time is 0.0738 in scenario 11. Meanwhile, in the NCC-2 dataset, the average processing time is 44.3419 seconds, the longest processing time of the other two datasets. The lowest processing time is for sensor-ID 2 data, and the highest is for sensor-ID 3. The resulting processing time has differences between scenarios in each dataset due to the characteristics of the data seen from the number of records, the number of bot activities, and the number of attacking bots in each scenario or each sensor data, data cleansing process, and feature selection techniques in the non-mandatory feature category.

4.6 Discussion

In this paper, the proposed model is compared with previous research. Previous research used feature selection techniques with the wrapper method [29], fuzzy [45], CfsSubsetEval [21], and carried out the manual selection [1, 23, 47].

In this paper, the features used in classification are divided into two types: mandatory and non-mandatory. Non-mandatory features are extracted and selected using univariate and ANOVA to obtain new features. Then, the feature selection results are merged with the mandatory features to be processed at the classification stage. The proposed model results are tested on three different datasets, namely CTU-13, NCC, and NCC-2. Test results with the CTU-13 dataset show that the proposed model has a detection accuracy of 99.27%, higher than previous research such as [29], which only maximum reached 94.40% by Decision tree algorithm, research in [21] which only reached maximum at 98.40% by Multiclass Classification and of 98.40% by Logistic Regression Algorithm, research in [46] which only reached a value of 97.54% by Decision tree algorithm, and research in [47] of 98.50% using Feedforward Neural Network (FNN). However, the detection accuracy results of the proposed model have a lower value than research in [23] with detection accuracy of 99.99% in the CTU-13 dataset and 99.99% in the NCC dataset, lower than [25] in the NCC dataset of 99.73%, lower than research [1] of 99.93% in CTU-13, 99.99% of 99.93% in the CTU-13 dataset, 99.99% in the NCC and 100% in the NCC-2 dataset.

The comparison results with previous research show that the detection accuracy value of the proposed model has a lower accuracy value. Still, the precision and recall measurement results produce better performance. A comparison of precision values shows that the proposed model has a value of 98.68% in the CTU-13 dataset, 98.26% in the NCC dataset and 97.90% in NCC-2, the highest compared to several previous studies[1, 46, 47]. However, it has a lower precision and recall value than [45], which reached at 99.20% and 99.60% using the ANN algorithm. Compared with research in [1, 46, 47] the results of measuring the recall value, the proposed model has a higher value in the three datasets, namely 99.27% in the CTU-13 dataset, 98.96 in the NCC dataset, and 98.87% in the NCC-2 dataset. Besides, the proposed model has computational time analysis, which has yet to be carried out in previous research. Computing time is calculated from the data transformation process to evaluate accuracy, precision, and recall measurements. The highest use of memory resources during the computing process uses 95% of the memory specification of 16 GB. The comparison results are shown in Table 11.

In previous research, the detection model was calculated based on the number of attackers, namely Botnet IP addresses, so the accuracy value in [1, 45] had a higher accuracy value. Meanwhile, the proposed model calculates based on the number of traffic flows successfully detected as Botnet activity. The analysis model based on the number of traffic flows in the proposed model influenced the increase in precision and recall measurements compared to the research in [1]. Besides, separating mandatory and non-mandatory features influences the detection process, increasing the precision and recall values.

5. Conclusions

This research proposes a botnet detection model using the Decision Tree classification method by analyzing network traffic. The features contained in network traffic are divided into two parts: mandatory features, which are basic and must be present in the detection model. Meanwhile, the second part is non-mandatory features, which are extracted and selected for use in the detection model. Two approaches were used in the feature selection process: Univariate and ANOVA in the non-mandatory feature category. The results of the feature selection analysis obtained the seven best features that can be used in the detection model using the Decision Tree classification method. Botnet activity detection testing was carried out on three public datasets, namely CTU-13, NCC, and NCC-2. The analysis obtained an average accuracy value of 99.27%, and the highest accuracy was 99.90% on the CTU-13 dataset. The average accuracy value in the NCC dataset is 98.96%, with the highest accuracy being 99.90%. Meanwhile, the

average detection rate in NCC-2 is 98.87%, with the highest accuracy being 99.90%. The resulting average precision value is 98.68% in the CTU-13 dataset, 98.26% in the NCC dataset, and 97.90% in the NCC-2 dataset. Finally, the resulting average recall value was 99.27% on the CTU-13 dataset, 98.96% on the NCC dataset, and 98.87% on the NCC-2 dataset. Processing time measurements on the three datasets show that the model can detect botnet attacks with the fastest average processing time of 17.7299 seconds on the NCC dataset, with the average number of data records in each scenario being 1,192,796. Compared with previous research, the proposed model can have the highest accuracy on the CTU13 dataset with a value of 99.27% higher than previous research in [1, 21, 29, 46, 47]. The precision value has a higher value obtained at 98.68% tested in CTU-13 Dataset and shows higher value research in [1, 9, 46]. On the recall evaluation, the proposed model obtained 99.27% tested with CTU-13 Dataset and shows higher than [1, 9, 46]. Still, the proposed model has lower accuracy than research in [23, 45], and has lower precision and recall than [45]. Even though it has a lower value compared to previous research, the proposed model has a processing time analysis that has never been carried out in previous research.

The feature selection process influences the results of the accuracy, precision, and recall analysis. So, comparison and development of feature selection techniques can be carried out in further research. Besides, testing of classification methods for developing feature selection techniques can be carried out in future research to perfect and optimize the malware attack detection process. Thus, the detection model can make it easier for administrators to take steps to handle or anticipate malware attacks that occur.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization DPH, TA and MARP; methodology DPH, TA and MARP; software DPH, GAP and ML; analysis DPH, MARP, DWA, and GAP; investigation DPH, TA, and MARP; writing—original draft preparation DPH and DWA; validation DPH and ML; writing—review and editing TA, DPH, and MARP; visualization DPA and MARP; funding acquisition DPH.

Acknowledgments

This work has been supported by the Institut Teknologi dan Bisnis STIKOM Bali and Institut Teknologi Sepuluh Nopember (ITS).

References

[1] M. A. R. Putra, T. Ahmad, D. P. Hostiadi, R. M. Ijtihadie, and P. Maniriho, “Botnet Attack Analysis through Graph Visualization”, *Int. J. Intell. Eng. Syst.*, Vol. 17, No. 1, 2024, doi: 10.22266/ijies2024.0229.75.

[2] R. Younisse, M. Alkasassbeh, M. Almseidin, and H. Abdi, “an Early Detection Model for Kerberoasting Attacks and Dataset Labeling”, *Jordanian J. Comput. Inf. Technol.*, Vol. 9, No. 1, pp. 1–10, 2023, doi: 10.5455/jjcit.71-1661423262.

[3] Z. Ashi, L. Aburashed, M. Al-Qudah, and A. Qusef, “Network Intrusion Detection Systems Using Supervised Machine Learning Classification and Dimensionality Reduction Techniques: A Systematic Review”, *Jordanian J. Comput. Inf. Technol.*, Vol. 07, No. 04, pp. 0–3, 2021.

[4] Y. Xing, H. Shu, H. Zhao, D. Li, and L. Guo, “Survey on Botnet Detection Techniques: Classification, Methods, and Evaluation”, *Math. Probl. Eng.*, Vol. 2021, 2021, doi: 10.1155/2021/6640499.

[5] K. E. Mwangi, S. Masupe, and J. Mandu, “Modelling malware propagation on the internet of things using an agent-based approach on complex networks”, *Jordanian J. Comput. Inf. Technol.*, Vol. 6, No. 1, pp. 26–40, 2020, doi: 10.5455/jjcit.71-1568145650.

[6] M. Singh, M. Singh, and S. Kaur, “Issues and challenges in DNS based botnet detection: A survey”, *Comput. Secur.*, Vol. 86, pp. 28–52, 2019, doi: 10.1016/j.cose.2019.05.019.

[7] S. Fallah and A. J. Bidgoly, “Benchmarking machine learning algorithms for android malware detection”, *Jordanian J. Comput. Inf. Technol.*, Vol. 5, No. 3, pp. 216–230, 2019, doi: 10.5455/jjcit.71-1558862640.

[8] S. Hosseini, A. E. Nezhad, and H. Seilani, “Botnet detection using negative selection algorithm, convolution neural network and classification methods”, *Evol. Syst.*, Vol. 13, No. 1, pp. 101–115, 2022, doi: 10.1007/s12530-020-09362-1.

[9] T. S. Naseri and F. S. Gharehchopogh, “A Feature Selection Based on the Farmland Fertility Algorithm for Improved Intrusion Detection Systems”, *J. Netw. Syst. Manag.*, Vol.

30, No. 3, pp. 1–27, 2022, doi: 10.1007/s10922-022-09653-9.

[10] O. A.M. and J. R.G., "Towards Building an Improved Botnet Detection Model in Highly Imbalance Towards Building an Improved Botnet Detection Model in Highly Imbalance Botnet Dataset-A Methodological Framework", *Mejast.Com*, Vol. 3, No. March, pp. 33–38, 2020.

[11] M. Asadi, M. A. Jabraeil Jamali, S. Parsa, and V. Majidnezhad, "Detecting botnet by using particle swarm optimization algorithm based on voting system", *Futur. Gener. Comput. Syst.*, Vol. 107, pp. 95–111, 2020, doi: 10.1016/j.future.2020.01.055.

[12] J. Velasco-Mata, V. Gonzalez-Castro, E. F. Fernandez, and E. Alegre, "Efficient Detection of Botnet Traffic by Features Selection and Decision Trees", *IEEE Access*, Vol. 9, pp. 120567–120579, 2021, doi: 10.1109/ACCESS.2021.3108222.

[13] T. S. Naseri and F. S. Gharehchopogh, "A Feature Selection Based on the Farmland Fertility Algorithm for Improved Intrusion Detection Systems", *J. Netw. Syst. Manag.*, Vol. 30, No. 3, pp. 40, 2022, doi: 10.1007/s10922-022-09653-9.

[14] S. Li, Y. Cao, S. Liu, Y. Lai, Y. Zhu, and N. Ahmad, "HDA-IDS: A Hybrid DoS Attacks Intrusion Detection System for IoT by using semi-supervised CL-GAN", *Expert Syst. Appl.*, Vol. 238, No. PF, pp. 122198, 2024, doi: 10.1016/j.eswa.2023.122198.

[15] A. Affinito, S. Zinno, G. Stanco, A. Botta, and G. Ventre, "The evolution of Mirai botnet scans over a six-year period", *J. Inf. Secur. Appl.*, Vol. 79, No. October, pp. 103629, 2023, doi: 10.1016/j.jisa.2023.103629.

[16] R. Abrantes, P. Mestre, and A. Cunha, "Exploring Dataset Manipulation via Machine Learning for Botnet Traffic," *Procedia Comput. Sci.*, Vol. 196, No. 2021, pp. 133–141, 2021, doi: 10.1016/j.procs.2021.11.082.

[17] Z. Ismail, A. Jantan, M. N. Yusoff, and M. U. Kiru, "The effects of feature selection on the classification of encrypted botnet", *J. Comput. Virol. Hacking Tech.*, Vol. 17, No. 1, pp. 61–74, 2021, doi: 10.1007/s11416-020-00367-7.

[18] R. Nair and A. Bhagat, "Feature selection method to improve the accuracy of classification algorithm", *Int. J. Innov. Technol. Explor. Eng.*, Vol. 8, No. 6, pp. 124–127, 2019.

[19] M. Alshamkhany, W. Alshamkhany, M. Mansour, M. Khan, S. Dhou, and F. Aloul, "Botnet Attack Detection using Machine Learning", In: *Proc. 2020 14th Int. Conf. Innov. Inf. Technol. IIT 2020*, no. February, pp. 203–208, 2020, doi: 10.1109/IIT50501.2020.9299061.

[20] M. I. Hossain, S. Eshrak, M. J. Auvik, S. F. Nasim, R. Rab, and A. Rahman, "Efficient Feature Selection for Detecting Botnets based on Network Traffic and Behavior Analysis", In: *ACM Int. Conf. Proceeding Ser.*, pp. 56–61, 2020, doi: 10.1145/3428363.3428378.

[21] L. Mathur, M. Raheja, and P. Ahlawat, "Botnet Detection via mining of network traffic flow", *Procedia Comput. Sci.*, Vol. 132, pp. 1668–1677, 2018, doi: 10.1016/j.procs.2018.05.137.

[22] D. P. Hostiadi, T. Ahmad, and W. Wibisono, "A New Approach to Detecting Bot Attack Activity Scenario", In: *Proc. of the 12th International Conference on Soft Computing and Pattern Recognition (SoCPaR 2020)*, 2021, pp. 823–835.

[23] M. A. R. Putra, T. Ahmad, and D. P. Hostiadi, "Analysis of Botnet Attack Communication Pattern Behavior on Computer Networks", *Int. J. Intell. Eng. Syst.*, Vol. 15, No. 4, pp. 533–544, 2022, doi: 10.22266/ijies2022.0831.48.

[24] V. Quezada, F. Astudillo-Salinas, L. Tello-Oquendo, and P. Bernal, "Real-time bot infection detection system using DNS fingerprinting and machine-learning", *Comput. Networks*, Vol. 228, No. October 2022, p. 109725, 2023, doi: 10.1016/j.comnet.2023.109725.

[25] D. P. Hostiadi and T. Ahmad, "Hybrid model for bot group activity detection using similarity and correlation approaches based on network traffic flows analysis", *J. King Saud Univ. - Comput. Inf. Sci.*, Vol. 34, No. 7, pp. 4219–4232, 2022, doi: 10.1016/j.jksuci.2022.05.004.

[26] M. Aamir and S. M. Ali Zaidi, "Clustering based semi-supervised machine learning for DDoS attack classification", *J. King Saud Univ. - Comput. Inf. Sci.*, Vol. 33, No. 4, pp. 436–446, 2021, doi: 10.1016/j.jksuci.2019.02.003.

[27] S. Chowdhury, M. Khanzadeh, R. Akula, F. Zhang, S. Zhang, H. Medal, M. Marufuzzaman, and L. Bian, "Botnet detection using graph-based feature clustering", *J. Big Data*, Vol. 4, No. 1, 2017, doi: 10.1186/s40537-017-0074-7.

[28] C. Y. Wang, C. L. Ou, Y. E. Zhang, F. M. Cho, P. H. Chen, J. B. Chang, and C. K. Shieh, "BotCluster: A session-based P2P botnet clustering system on NetFlow", *Comput. Networks*, Vol. 145, pp. 175–189, 2018, doi: 10.1016/j.comnet.2018.08.014.

[29] R. U. Khan, X. Zhang, R. Kumar, A. Sharif, N.

A. Golilarz, and M. Alazab, "An adaptive multi-layer botnet detection technique using machine learning classifiers", *Appl. Sci.*, Vol. 9, No. 11, 2019, doi: 10.3390/app9112375.

[30] A. P. and T. A. T., "'Effective Feature Selection for Botnet Detection Based on Network Flow Analysis", *Int. Conf. Autom. Informatics*, 2017.

[31] S. Haq and Y. Singh, "Botnet detection using machine learning", In: *Proc. of PDGC 2018 - 2018 5th Int. Conf. Parallel, Distrib. Grid Comput.*, pp. 240–245, 2018, doi: 10.1109/PDGC.2018.8745912.

[32] D. P. Hostiadi, T. Ahmad, and W. Wibisono, "A New Approach of Botnet Activity Detection Model based on Time Periodic Analysis", In: *CENIM 2020 - Proc. Int. Conf. Comput. Eng. Network, Intell. Multimed. 2020*, no. Cenim 2020, pp. 315–320, 2020, doi: 10.1109/CENIM51130.2020.9297846.

[33] R. F. M. Dollah, M. A. Faizal, F. Arif, M. Z. Mas'ud, and L. K. Xin, "Machine learning for HTTP botnet detection using classifier algorithms", *J. Telecommun. Electron. Comput. Eng.*, Vol. 10, No. 1–7, pp. 27–30, 2018.

[34] P. Jamal, M. Ali, R. H. Faraj, P. J. M. Ali, and R. H. Faraj, "Data Normalization and Standardization: A Technical Report", *Mach. Learn. Tech. Reports*, Vol. 1, No. 1, pp. 1–6, 2014.

[35] S. C. Nayak, B. B. Misra, and H. S. Behera, "Impact of Data Normalization on Stock Index Forecasting", *Int. J. Comput. Inf. Syst. Ind. Manag. Appl.*, Vol. 6, No. 2014, pp. 257–269, 2014.

[36] S. G. K. Patro and K. K. sahu, "Normalization: A Preprocessing Stage", *Iarjset*, no. March, pp. 20–22, 2015, doi: 10.17148/iarjset.2015.2305.

[37] M. Al-Sarem, F. Saeed, E. H. Alkhammash, and N. S. Alghamdi, "An aggregated mutual information based feature selection with machine learning methods for enhancing iot botnet attack detection", *Sensors*, Vol. 22, No. 1, 2022, doi: 10.3390/s22010185.

[38] H. Nkiama, S. Zainudeen, and M. Saidu, "A Subset Feature Elimination Mechanism for Intrusion Detection System", *Int. J. Adv. Comput. Sci. Appl.*, Vol. 7, No. 4, pp. 148–157, 2016, doi: 10.14569/ijacsa.2016.070419.

[39] M. Alazab, "Automated malware detection in mobile app stores based on robust feature generation", *Electron.*, Vol. 9, No. 3, 2020, doi: 10.3390/electronics9030435.

[40] R. F. M. Dollah, M. A. Faizal, F. Arif, M. Z. Mas'ud, and L. K. Xin, "Machine learning for HTTP botnet detection using classifier algorithms", *J. Telecommun. Electron. Comput. Eng.*, Vol. 10, No. 1–7, pp. 27–30, 2018.

[41] S. García, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods", *Comput. Secur.*, Vol. 45, pp. 100–123, 2014, doi: 10.1016/j.cose.2014.05.011.

[42] D. P. Hostiadi and T. Ahmad, "Dataset for Botnet Group Activity with Adaptive Generator", *Data Br.*, Vol. 38, pp. 107334, 2021, doi: 10.1016/j.dib.2021.107334.

[43] M. A. R. Putra, D. P. Hostiadi, and T. Ahmad, "Botnet dataset with simultaneous attack activity", *Data Br.*, Vol. 45, 2022, doi: 10.1016/j.dib.2022.108628.

[44] M. A. R. Putra, T. Ahmad, and D. P. Hostiadi, "Analysis of Botnet Attack Communication Pattern Behavior on Computer Networks", *Int. J. Intell. Eng. Syst.*, Vol. 15, No. 4, 2022, doi: 10.22266/ijies2022.0831.48.

[45] C. Joshi, R. K. Ranjan, and V. Bharti, "A Fuzzy Logic based feature engineering approach for Botnet detection using ANN", *J. King Saud Univ. - Comput. Inf. Sci.*, Vol. 34, No. 9, pp. 6872–6882, 2021, doi: 10.1016/j.jksuci.2021.06.018.

[46] I. Letteri, G. Della Penna, and P. Caianiello, "Feature selection strategies for http botnet traffic detection", In: *Proc. of 4th IEEE Eur. Symp. Secur. Priv. Work. EUROS PW 2019*, pp. 202–210, 2019, doi: 10.1109/EuroSPW.2019.00029.

[47] M. Eslahi, W. Z. Abidin, and M. V. Naseri, "Correlation-based HTTP Botnet detection using network communication histogram analysis", In: *Proc. of 2017 IEEE Conf. Appl. Inf. Netw. Secur. AINS 2017*, Vol. 2018-Janua, pp. 7–12, 2017, doi: 10.1109/AINS.2017.8270416.