

Intrusion detection system for imbalance ratio class using weighted XGBoost classifier

Januar Al Amien¹, Hadhrami Ab Ghani³, Nurul Izrin Md Saleh³, Edi Ismanto², Rahmad Gunawan¹

¹Department of Informatics Engineering, Faculty Computer Science, University Muhammadiyah Riau, Riau, Indonesia

²Department of Informatics Education, Faculty Computer Science, University Muhammadiyah Riau, Riau, Indonesia

³Department of Data Science, Universiti Malaysia Kelantan, Kelantan, Malaysia

Article Info

Article history:

Received Oct 25, 2022

Revised Jan 20, 2023

Accepted Feb 16, 2023

Keywords:

Imbalanced ratio class

Intrusion detection

Weighted XGBoost

ABSTRACT

The rapid development of the internet of things (IoT) has taken an important role in daily activities. As it develops, IoT is very vulnerable to attacks and creates IoT for users. Intrusion detection system (IDS) can work efficiently and look for activity in the network. Many data sets have already been collected, however, when dealing with problems involving big data and high data imbalances. This article proposes, using the dataset used by BotIoT to evaluate the system framework to be created, the XGBoost model to improve the detection performance of all types of attacks, to control unbalanced data using the imbalance ratio of each class weight (CW). The experimental results show that the proposed approach greatly increases the detection rate for infrequent disturbances.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Januar Al Amien

Department of Informatics Engineering, Faculty Computer Science

University Muhammadiyah Riau, Riau, Indonesia

Email: januaralamien@umri.ac.id

1. INTRODUCTION

The internet of things (IoT) is undoubtedly vital for industry, hospitals, education because it provides services that are cost-efficient and time-saving [1]. Although IoT is preferred over conventional devices and systems, however, as it develops, IoT is highly vulnerable to attacks and raises concerns for users. Despite this, they are able to provide helpful insights on the behaviour of the traffic besides contributing to the discovery of important information. The way to recognize network traffic changes is through the intrusion detection system (IDS), it can help find, determine, and identify anomalous network traffic [2].

IDS is able to function effectively and can monitor the network for potentially malicious behaviour. The majority of attacks originate from remote locations, specifically from computers connected to the internet that are attempting to communicate with the host network. IDS operates as a sensor, keeping a watchful eye on the network that it is utilised on. IDS can detect a deviant network and report it to the network administrator. Another advantage of IDS is that it is capable of communicating with computers and smart phones [3]. IDS can identify unknown attacks from network traffic and has become one of the most effective network security. At this time, existing methods for detecting network anomalies typically use machine learning (ML) models, such as K-nearest neighbor (KNN) and support vector machine (SVM). Although this method achieves satisfactory performance, its achievable accuracy is relatively low [4]. Applying a low accuracy IDS is risky as most of the computer networks nowadays are dealing with millions of network traffic flow obtained from the internet. Datasets can be used to evaluate IDS in order to detect activity abnormal attacks such as denial-of-service (DoS) attacks, brute force, ping scan and port scan [5].

In the past few years, ML techniques have become increasingly popular, such as in finding normal actions in the beginning while looking for malicious activities in computer systems. Therefore, it can protect the user by keeping the system safe and stopping attacks much faster than before. Many mechanisms which applies ML have been proposed in literature such as AdaBoost [6], Naïve Bayes (NB) and random forest (RF) [7]. On top of the popular existing methods like logistic regression (LR) and support vector machine (SVM) [8], new algorithms have also been proposed to boost the performance such as XGBoost [9], [10], where the idea is taken from an algorithm that is often used by various antivirus software companies - the decision tree [11]. Hyperparameter optimization has been proposed in [12], to combat attacks based on artificial intelligence (AI) approach. Conventional ML model names, such as traditional learning techniques are no longer suitable for big data. Therefore, formulating and developing improved ML algorithms such as XGBoost for designing an effective IDS in computer and IoT systems and networks will be the focus of this project. In the next session, the problem background of this research project will be further described.

The majority of network traffic flow data and cyber security data suffer from an innate flaw: the data are grossly unbalanced. This means that the amount of normal or benign traffic data is significantly larger than malicious traffic data in most situations. Relatively few assault incidents [13]. Researchers need to give serious thought to the problem of class imbalance before putting security analytics into action [14]. However, their usefulness and precision as intrusion detection systems (IDSs) are still open to debate, in particular when addressing issues that concern unbalanced data [15]. The focus of the contribution in this research is:

- a) To design an IDS framework that can handle highly unbalanced network traffic data based on ratio class weights.
- b) To design enhanced ensemble learning techniques with XGBoost to maximize attack classification speed and detection performance.

The organization of the remaining parts of the paper is mainly made in three different parts which are the related work, the method and the results, as presented in section 2, section 3 and section 4 respectively. The conclusion is written in section 5 to summarize the findings in this article. Acknowledgement is then presented afterwards.

2. RELATED WORK

The problem found is the need to overcome class problems in datasets that are still experiencing data imbalance. According to [16], the suggested approach, coined as Siam-IDS, is demonstrated effective to find attacks including user to root (U2R) and root to local (R2L) via improved balancing schemes. This is significant since these methods can be used to detect R2L and U2R assaults. Comparisons were made between the performance of Siam-IDS and that of other IDSs already on the market that were built utilising DL approaches. In comparison to its contemporaries Siam-IDS has performed better in identifying U2R and R2L classes to detect these attacks for resolving uneven data distribution issues [17].

It is reported in Liu *et al.* [18] the difficult set sampling technique (DSSTE) approach can be used to tackle the issue of class imbalance in intrusion detection, which reduces the used samples in the set's majority while simultaneously raising those from the set's minority. During the training phase, the classifier is better able to comprehend the differences that exist between the sets as a result of this. One-side selection is the first strategy that reduces the number of noise samples taken from the majority category. The second technique, synthetic minority over-sampling technique (SMOTE), increases the number of samples taken from the minority category.

Establishing a balanced dataset in this way helps the model to better understand the characteristics of the minority samples while also significantly reducing the training time needed for the model. Establishing a balanced dataset also helps ensure that the model is accurate. In the second step, a deep hierarchical network model is constructed using a combination of a convolutional neural network (CNN) and a bidirectional long short-term memory (BiLSTM) to extract spatial and temporal features, respectively. Both of these networks belong to the family of convolutional neural networks (CNNs). The procedure will be finished after this step [19].

An imbalance in the classes refers to the situation in which a dataset has a different number of entries in positive and negative classes. This can happen when the dataset comprises both positive and negative categories. The dominance (Dom) or prevalence connection between a positive class and a negative class is what defines the imbalance, which is defined as total positive ratings minus total negative ratings (TPR – TNR). After that point, this amount is used as a correction factor to performance indicators that have been adversely impacted as a result of concerns of imbalance. On the other hand, dominance is not a measure of imbalance in datasets since it already takes into account imbalances in the findings of classifiers. Instead, it is a measure of the extent to which a dataset is dominated by a single variable [20]. This research seeks to maximize the existing data by designing and building a model to overcome the problem of unbalanced data size use ratio class weight.

3. METHOD

3.1. Proposed imbalance ratio class weight (CW)

The dataset utilized in this study is the Bot_Iot dataset. There are also problems in the unbalanced dataset and detected outliers which can cause the prediction performance to be less than optimal. Although the problem of imbalance has been discussed in previous studies [21]. To overcome the problem of imbalance-XGBoost. In order to achieve the best accuracy and can be applied later to ML. Designed to improve unbalanced data by maximizing existing data and increasing accuracy by using the XGBoost model, here are the stages in the study.

Research motivation will contribute to using ensemble techniques, gradient enhancement, with the XGBoost framework shown in Figure 1. Combining training and test data and processing including data cleaning. Then, selecting features and extracting features and calculating in each class dividing it into ratio form, finally, the process of classifying the model using XGBoost, and to evaluate the model using the confusion matrix.

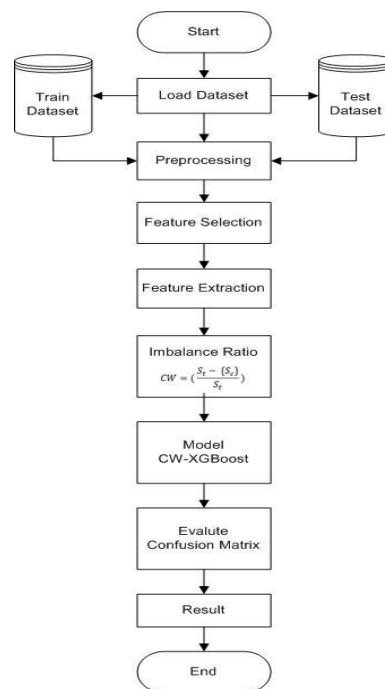


Figure 1. Diagram classification model

3.2. Dataset BotIoT

The author uses a dataset from Koroniotis. The dataset has a column count of 19 from XGBoost in the Bot_IoT dataset. The dataset columns used in this study are categories and attacks containing letters and numbers. The attack column contains the number 1 for attack and 0 for normal. The number of training data consists of 2934447 attacks and 370 normal, the testing data contains 733598 attacks and 107 normal. Then in this data analysis, data exploration is carried out that describes and explains visually, and explains information from a lot of data. This process is intended to understand the structure of the data to be processed.

3.3. SMOTE

As its name implies, SMOTE is an oversampling approach. This approach generates synthetic data by oversampling data from minority classes. By generating synthetic data, SMOTE avoids the problem of overfitting caused by random oversampling procedures. The SMOTE technique selects instances that are close in the feature space. It starts by selecting a random sample from the minority class and identifies its two nearest neighbors. Then, one of the nearest neighbors is chosen at random, and the difference between the feature values of the sample and its selected neighbor is multiplied by a random value between 0 and 1 and added to the feature values of the sample. This creates a line between the two samples, along which synthetic samples are generated. The number of synthetic samples generated depends on the desired level of oversampling, which is determined by randomly selecting instances from the k nearest neighbors [22].

3.4. Adaptive synthetic (ADASYN)

ADASYN is predicated on the concept of producing data samples for minority classes in an adaptive manner, taking into account the distributions of those classes; the ADASYN approach generates additional synthetic data for minority class samples that are more difficult to learn, compared to those that are easier to learn. This approach not only aims to minimize learning bias due to initial unbalanced data distribution, but also has the potential to adjust the decision boundary adaptively by focusing on variables that are the most challenging to predict [23]. Overall, the ADASYN technique provides a promising solution to address the problem of imbalanced datasets in machine learning, and has been shown to achieve state-of-the-art results in various classification tasks.

3.5. Borderline SMOTE

In order to make the sample distribution more even, the borderline-synthetic minority oversampling technique, also known as SMOTE, is used in this study. However, borderline-SMOTE is more focused on the distribution and creation of minority samples near the border, making its sample selection more representative than regular SMOTE. Additionally, to address the issue of imbalanced sample distribution, this study uses undersampling for the majority class. The balanced dataset resulting from these techniques is compared to the original dataset [24].

3.6. TOMEK

Algorithmic techniques can provide the most effective solution to class imbalance problems by improving classifiers. Cost-sensitive techniques develop a cost function to prevent misclassification, which takes into account the class imbalance; this method also counts among the effective algorithmically-based approaches [25]. According [22], the SMOTE + Tomek-Links (STL) algorithm, The STL algorithm, or the synthetic minority over-sampling technique with local interpolation factors, first generates synthetic data by sampling from the majority class. Then, in the oversampled data set, the noise samples, which are synthetic samples that are not well-supported by the original data, are identified and removed to provide better class refinement.

3.7. SMOTEENN

The hybrid-sampling technique, the combined over-sampling and under-sampling technique, typically referred to as hybrid sampling, is used to generate more samples that are similar to the minority class while discarding some of the majority class data. This is accomplished by combining the over-sampling method with the under-sampling method (or classes). This is done in order to generate more samples that are similar to the data belonging to the minority class. It does this by employing two of the most common strategies, namely: SMOTETomek – SMOTE + Tomek and SMOTEENN – SMOTE + ENN [26].

3.8. XGBoost

According to Chen and Guestrin [27], because the ensemble tree model utilises functions themselves as parameters, it is not possible to optimise the model in Euclidean space using the conventional optimization techniques. In contrast to that, the model is trained incrementally or progressively. Let the formall $\hat{y}_i(t)$ let the prediction of the $-i$ instance in $-t$ iteration, we need to add f_t to minimize the following goal.

$$\mathcal{L}^{(t)} = \sum_{i=1}^n I(\hat{y}_i, y_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \quad (1)$$

Second-order approximations can be employed to rapidly optimize objectives in various scenarios.

$$\mathcal{L}^{(t)} = \sum_{i=1}^n [l(\hat{y}_i, y_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \quad (2)$$

Where $g_i = \partial \hat{y}^{(t-1)} l(\hat{y}_i, y_i^{(t-1)})$ and $h_i = \partial^2 \hat{y}^{(t-1)} l(\hat{y}_i, y_i^{(t-1)})$ the first and second derivatives of the loss function are referred to as first and second-order gradient statistics. By ignoring the constant term, we can obtain a simplified objective for step t :

$$\mathcal{L}^{(t)} = \sum_{i=1}^n (g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)) + \Omega(f_t) \quad (3)$$

Assuming a fixed structure $q(x)$, we can compute the optimal weight w_j^* leaf j by:

$$w_j^* = - \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} \quad (4)$$

And calculate the optimal value corresponding to:

$$\mathcal{L}^{(t)}(q) = -\frac{1}{2} \sum_{j=1}^T \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma T \quad (5)$$

Enumerating all possible tree structures q is usually infeasible. The approach typically used is an algorithm that starts with a single leaf and gradually adds branches to the tree. Assume that I_L dan I_R is the set of instances of the left and right nodes after splitting. Letting $I = I_L \cup I_R$, the reduction in loss after the split can be calculated as:

$$\mathcal{L}_{split} = \frac{1}{2} \left[\frac{(\sum_{i \in L_j} g_i)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{(\sum_{i \in R_j} g_i)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{(\sum_{i \in I} g_i)^2}{\sum_{i \in I} h_i + \lambda} \right] - \gamma \quad (6)$$

3.9. Confusion matrix observation

According to Liu *et al.* [26], various measurement metrics are pertinent. Since it is widely accepted that accuracy alone does not provide an adequate performance evaluation, we provide values for most of these metrics. This is especially the case when the datasets contain more positive examples than negative examples.

Precision is a representation of the reliability of the retrieval. Therefore, it needs to be as high as it possibly can be. It determines the proportion of true positive samples to those that the system identifies as positive, whether they are true or not. It indicates the level of trustworthiness in the system's predictions, and therefore should be maximized to the greatest extent possible.

$$Precision \% = \frac{TP}{TP+FP} \quad (7)$$

The ratio of true positive samples to the total number of actual positive samples is denoted by the recall, which is often referred to as the detection rate (DR). It is a fundamental metric that is frequently used to quantify the quality of the anomaly detection that is being taken into consideration, and it represents the completeness of the retrieval. As a result, it ought to be as high as is humanly feasible, and its value may be determined as:

$$Recall \% = \frac{TP}{TP+FN} \quad (8)$$

The term "F1" refers to the harmonic mean of the terms "precision" and "recall". It comprises a summary of the results of the retrieval performance. When the value of F1 is larger, it implies that the method performs better in terms of recall and precision. Therefore, it need to be as high as it possibly can be.

$$F1 = 2 \times \frac{Presisi \times Recall}{Presisi + Recall} \quad (9)$$

The false negative rate (FNR) is the ratio of the number of false negative samples to the total number of actual positive samples. It is a symbol of the incapacity to recognise a true good result. If this number is high, the genuine assaults won't be caught, which will leave the system vulnerable to being exploited by hostile users and put it in an unsafe state. As a result, it need to be reduced to the barest essentials.

$$FNR \% = \frac{FN}{TP+FN} \quad (10)$$

The false positive rate (FPR), also known as the false alarm rate (FAR), is the ratio of false positive samples to the total number of actual negative samples. If this number is continuously high, the security analysis operator may intentionally disregard the system warnings, leading to the system entering a hazardous state [28]. Therefore, it needs to be reduced to the barest essential.

$$FPR \% = \frac{FP}{FP+TN} \quad (11)$$

When looking at metrics from a more holistic perspective, accuracy is by far the most important. It is the proportion of total samples to which each individual sample has been appropriately assigned. It is a representation of the classification's level of confidence. Therefore, it needs to be as high as it possibly can be.

$$Accuracy \% = \frac{TP+TN}{TP+FN+FP+TN} \quad (12)$$

4. RESULTS AND DISCUSSION

4.1. Dataset description

The Bot_IoT dataset published on the internet by Nickolaos Koroniotis is used as the primary input data for detecting botnet attacks: <https://research.unsw.edu.au/projects/bot-iot-dataset>. At this stage, the analysis process is carried out first on the data, in order to obtain data quality that is suitable for use in the mining process. This analysis is carried out to see how many features and columns are in the data, as tabulated in Table 1. There are two raw data, namely train data and test data, where train data is 2934817 and test data is 733705 with 19 features.

Table 1. Identification of dataset attack preparation on label

No	Category	Label (attack)
0	Anomly	3668045
1	Normal	477

4.2. Preprocessing

At this stage, preprocessing is done for classification on the dataset. The dataset has not been well structured, because there are non numeric. In order for data to be used, it must be done several stages of preprocessing, but on the sidelines there are other stages that can be done several stages of preprocessing, or features that contain categories (category and subcategory) using LabelEncoder.

4.3. Data selection and extraction

Feature selection is the process of identifying some of the most important features that help get better model accuracy. At this stage, the features are cleaned that are not used in the classification and only select numeric features. This dataset used 19 features and only 13 features will be used, as shown in Table 2.

The non-numeric fatures that will be deleted are: proto, saddr, sport, daddr, dport and remove the pkSeqID and attack features because they are not included in the classification. After cleaning the features, the features that will be classified are: seq, stddev, N_IN_Conn_P_SrcIP, min, state_number, mean, N_IN_Conn_P_DstIP, drate, srate, max, attack, category, subcategory. The selection of these features allows ML methods to research fast, increasing model accuracy. For feature extraction, the class on the label is extracted into 2 parts, namely 0 for anomaly and 1 for normal, this feature extraction will be calculated using the class weight formula.

Table 2. Features selection

Features	Used
PkSeqID	×
Proto	×
Saddr	×
Sport	×
Daddr	×
Dport	×
Seq	✓
Stddev	✓
N_IN_Conn_P_SrcIP	✓
Min	✓
State_number	✓
Mean	✓
N_IN_Conn_P_DstIP	✓
Drate	✓
Srate	✓
Max	✓
Attack	✓
Category	✓
Subcategory	✓

4.4. Imbalance data

While the XGBoost algorithm performs well for a variety of challenging problems, it offers a large number of parameters, many of which require fine tuning to get the most out of the algorithm on a given data set hence parameter tuning uses scale_pos_weight to control the balance of positive and negative ratios. To maximize the data imbalance due to problems in the large minority class, this study uses the ratio obtained by using the:

$$CW = \left(\frac{S_t - \{S_c\}}{S_t} \right) \tag{13}$$

This formula is a modified formula from parameters `scale_pos_weight`, where S_t is the total of all classes and S_c is the total share of each class in the dataset in each class, CW significantly controls the balance of positive and negative ratios created by the model during sample training. As shown in Table 3, the total number of S_t is 3668522 and the value of CW is 7689.82180293501 so that the ratio value is 0.9979038365306423.

In Figure 2 shows the results of applying various sampling techniques to address the imbalanced data problem in a binary classification task. The “algorithm” column lists the names of the techniques used, while the “after sampling” column lists the class distribution after applying the respective technique. Figure 2(a) the “our purpose” class weight technique was designed specifically for this task and has a relatively small number of samples in the minority class (0) compared to the other techniques. Figure 2(b) SMOTE, Figure 2(c) ADASYN increased the number of minority class samples slightly less than the other techniques. Figure 2(d) borderlineSMOTE, and Figure 2(e) SMOTETomek increased the number of minority class samples to match the majority class (1). On the other hand, Figure 2(f) SMOTEENN is a combination of over- and under-sampling techniques and resulted in a lower number of samples for the majority class (1). t can be observed that the resulting samples are evenly distributed in the minority sample group and it can also be seen that the number of each imbalance technique can be proven in Table 4.

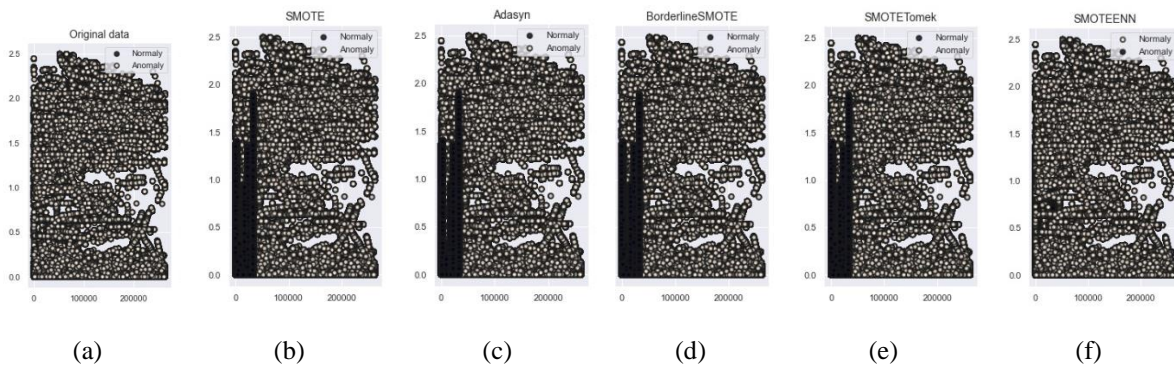


Figure 2. Comparison of binary classification techniques on imbalanced data: (a) original data, (b) SMOTE, (c) ADASYN, (d) borderlineSMOTE, (e) SMOTETomek, and (f) SMOTEENN

Table 3. Imbalance ratio class-weight

No	Category	Label (attack)
0	Anomaly	3668045
1	Normal	477
Total:		3668522
Ratio:		0.9979038365306423

Table 4. Compare sampler combining imbalance ratio class-weight and over-sampling

No	Algorithm	Before sampling	After sampling
0	Our purpose	{1: 3668045, 0: 477}	{1: 3668045, 0: 477}
1	SMOTE	{1: 3668045, 0: 477}	{1: 3668045, 0: 3668045}
2	Adasyn	{1: 3668045, 0: 477}	{1: 3668045, 0: 3667977}
3	BorderlineSMOTE	{1: 3668045, 0: 477}	{1: 3668045, 0: 3668045}
4	SMOTETomek	{1: 3668045, 0: 477}	{1: 3668045, 0: 3668045}
5	SMOTEENN	{1: 3668045, 0: 477}	{0: 3668042, 1: 3667680}

4.5. Predictive classification with XGBoost models

To create an XGBoost Model, you must first install some libraries to be able to call the XGBoost model. After installing the library, first set the XGBClassifier hyperparameters, especially the ratio to create the XGBoost model. If each hyperparameter is set independently, the search space dimension that the swarm needs to explore will be reduced. Thus the swarm can run for fewer iterations and gradually build on previous results. For tuning use (booster, objective, num_class, scale_pos_weight, eta, max_depth, eval_metric, nthread, and tree_method).

By performing statistical analysis on the original dataset, a subset of 13 features was identified as producing the best results out of the initial 19 features. From the Bot_IoT dataset using the XGBoost method with a share of 80% for training data and 20% for data testing. Based on the test results with the ensemble technique using the XGBoost algorithm.

The results shown in Figure 3 suggest that employing XGBoost with different imbalanced data procedures, such as Figure 3(a) original data, Figure 3(b) SMOTE, Figure 3(c) ADASYN, Figure 3(d) borderlineSMOTE, Figure 3(e) SMOTETomek, and Figure 3(f) SMOTEENN, can produce positive results, including a balanced class weight ratio, which is important in addressing the challenge of imbalanced data in binary classification tasks. The comparison of these procedures is presented in the form of a matrix table that describes the performance of the classification model on a series of test data, providing valuable information on the classification results obtained by the system with respect to the actual classification results.

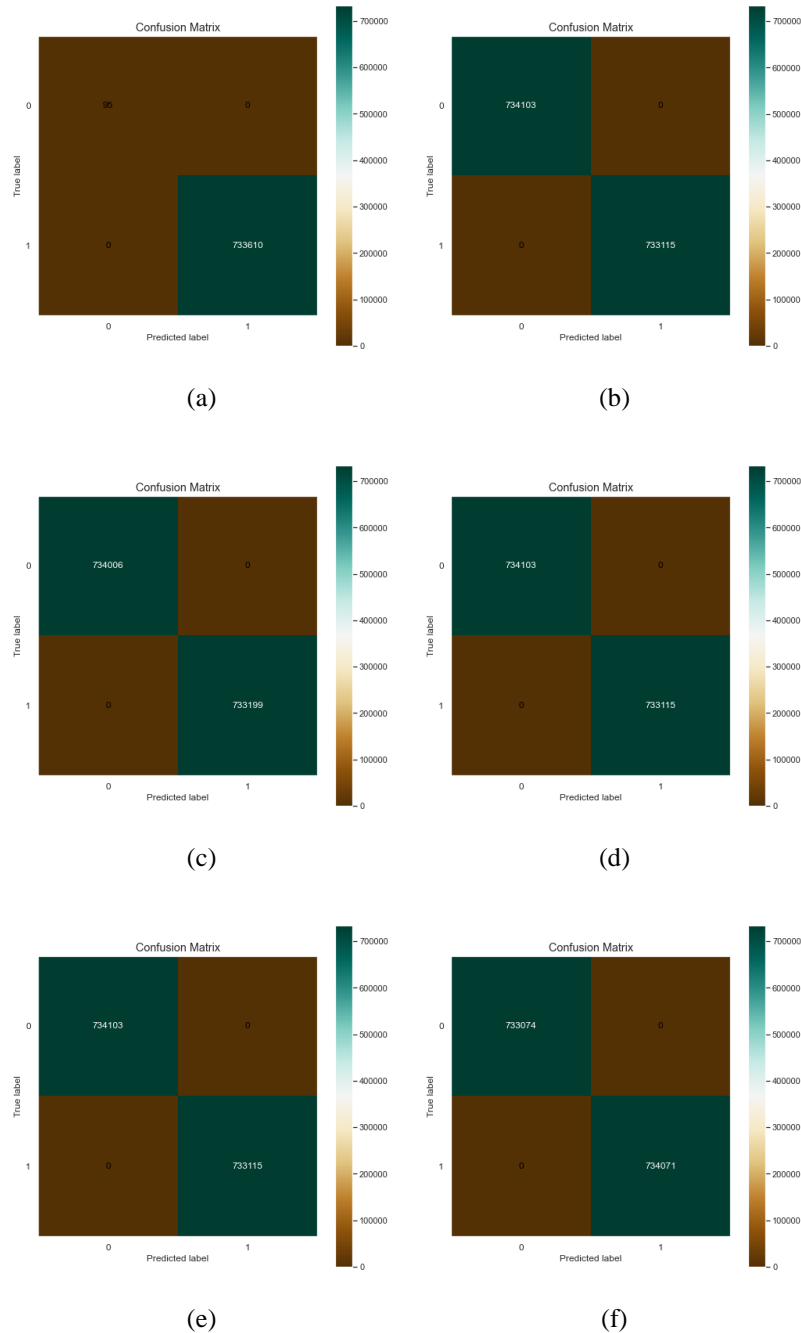


Figure 3. Compare binary classification results according to classes: (a) original data, (b) SMOTE, (c) ADASYN, (d) borderlineSmote, (e) SMOTETomek, and (f) SMOTEENN

4.6. Confusion matrix evaluation

Following the testing phase comes the assessment utilising a confusion matrix that comes next. The confusion matrix is essentially a matrix table that describes how a classification model performs on a set of test data whose real value is known, as shown in Table 5. This table offers information on the comparison of the system's (model's) classification results with the real classification results. This is performed by comparing the system's (model's) classification findings to the actual categorization results. In the confusion matrix, there are four words that together reflect the outcomes of the categorization process. These terms are:

Table 5. Compare binary classification confusion matrix evaluate

No	Algorithm	Accuracy	Precision	Recall	F1_Score	Time sec.
0	OUR PURPOSE	1.0	1.0	1.0	1.0	36.690853
1	SMOTE_XGBOOST	1.0	1.0	1.0	1.0	44.413793
2	ADASYN_XGBOOST	1.0	1.0	1.0	1.0	44.020221
3	BorderlineSMOTE_XGBOOST	1.0	1.0	1.0	1.0	43.463336
4	TOMEK_XGBOOST	1.0	1.0	1.0	1.0	45.369582
5	SMOTEENN_XGBOOST	1.0	1.0	1.0	1.0	44.770189

The accuracy obtained in the proposed model is 100%.

- True positive (TP) which is the class that is recorded correctly and the prediction model is also correct.
- True negative (TN) is the wrong class and the precision model is also wrong.
- False positive (FP) is a class in the data that is wrong but the prediction model is correct.
- False negative (FN) is a class in the data is true but the prediction model is wrong.

In the confusion matrix, it can be concluded that the TP value is 1099933 and overall accuracy: 100%, however, there is a difference in the time it is proven that our purpose can reduce the classification time.

5. CONCLUSION

In the dataset set Bot_IoT, which combines the two, is normal related to IoT and other network traffic, along with different types of attacks commonly used by botnets. With feature labels indicating the flow of attacks, attack categories and subcategories for possible multiclassified. Through statistical analysis a subset of the original dataset consisting of 19 features produced the best 13 features. From the Bot_IoT dataset using the XGBoost Method with a share of 80% for training data and 20% for data testing. Based on the test results with the ensemble technique using the XGBoost algorithm, it can be concluded that. Overcome imbalance classes using ratio technique and get a balanced number of classes. Determining the right parameters using XGBClassifier to improve accuracy for the model. Classification on XGBoost model gets 99.94% accuracy with minimum prediction error at confusion matrix. While in after the test results and evaluated using confusion matrix, the result that can be 100%. From the conclusions that have been described, it is recommended for other researchers who want to conduct research using datasets BoT_IoT it is recommended to use other algorithms and use other optimizations to find better accuracy performance and classification time. In Feture, the framework system created can be used, comparing it with the IDS dataset that has been collected by other researchers. For performance model XGBoost can use optimization.

ACKNOWLEDGEMENTS

The article is supported by University Muhammadiyah Riau, Indonesia and Universiti Malaysia Kelantan, Malaysia. The authors highly acknowledge the full support given by both universities towards the accomplishment of this paper. Besides, the authors also appreciate all helps and supports given by fellow researchers, formally or informally, in preparing this paper.




REFERENCES

- N. Koroniotis, "Designing an effective network forensic framework for the investigation of botnets in the internet of things," Ph.D. dissertation, School of Engineering and Information Technology, The University of New South Wales, AU, 2020, doi: 10.26190/unsworks/21942.
- L. Vigoya, D. Fernandez, V. Carneiro, and F. J. Nóvoa, "IoT dataset validation using machine learning techniques for traffic anomaly detection," *Electronics*, vol. 10, no. 22, 2021, doi: 10.3390/electronics10222857.
- S. S. Dhaliwal, A. Al Nahid, and R. Abbas, "Effective intrusion detection system using XGBoost," *Information*, vol. 9, no. 7, 2018, doi: 10.3390/info9070149.
- T. Su, H. Sun, J. Zhu, S. Wang, and Y. Li, "BAT: deep learning methods on network intrusion detection using NSL-KDD dataset," *IEEE Access*, vol. 8, pp. 29575-29585, 2020, doi: 10.1109/ACCESS.2020.2972627.




- [5] N. Oliveira, I. Praça, E. Maia, and O. Sousa, "Intelligent cyber attack detection and classification for network-based intrusion detection systems," *Applied Sciences*, vol. 11, no. 4, 2021, doi: 10.3390/app11041674.
- [6] H. Tyagi and R. Kumar, "Attack and anomaly detection in IoT networks using supervised machine learning approaches," *Revue d'Intelligence Artificielle*, vol. 35, no. 1, pp. 11-21, 2021, doi: 10.18280/ria.350102.
- [7] J. L. G. Torres, C. A. Catania, and E. Veas, "Active learning approach to label network traffic datasets," *Journal of Information Security and Applications*, vol. 49, 2019, doi: 10.1016/j.jisa.2019.102388.
- [8] M. Mimura, "Adjusting lexical features of actual proxy logs for intrusion detection," *Journal of Information Security and Applications*, vol. 50, 2020, doi: 10.1016/j.jisa.2019.102408.
- [9] P. Kumar, G. P. Gupta, and R. Tripathi, "An ensemble learning and fog-cloud architecture-driven cyber-attack detection framework for IoMT networks," *Computer Communications*, vol. 166, pp. 110-124, 2021, doi: 10.1016/j.comcom.2020.12.003.
- [10] I. Dutt, S. Borah, and I. K. Maitra, "Immune system based intrusion detection system (IS-IDS): A proposed model," *IEEE Access*, vol. 8, pp. 34929-34941, 2020, doi: 10.1109/ACCESS.2020.2973608.
- [11] D. Papamartzivanos, F. G. Mármol, and G. Kambourakis, "Dendron: Genetic trees driven rule induction for network intrusion detection systems," *Future Generation Computer Systems*, vol. 79, pp. 558-574, 2018, doi: 10.1016/j.future.2017.09.056.
- [12] B. A. Tama, L. Nkenyereye, S. M. R. Islam, and K. -S. Kwak, "An enhanced anomaly detection in web traffic using a stack of classifier ensemble," *IEEE Access*, vol. 8, pp. 24120-24134, 2020, doi: 10.1109/ACCESS.2020.2969428.
- [13] S. Bagui and K. Li, "Resampling imbalanced data for network intrusion detection datasets," *Journal of Big Data*, vol. 8, no. 6, 2021, doi: 10.1186/s40537-020-00390-x.
- [14] J. L. Leevy, J. Hancock, R. Zuech, and T. M. Khoshgoftaar, "Detecting cybersecurity attacks across different network features and learners," *Journal of Big Data*, vol. 8, no. 38, 2021, doi: 10.1186/s40537-021-00426-w.
- [15] M. H. L. Louk and B. A. Tama, "Revisiting gradient boosting-based approaches for learning imbalanced data: A case of anomaly detection on power grids," *Big Data and Cognitive Computing*, vol. 6, no. 2, 2022, doi: 10.3390/bdccc6020041.
- [16] P. Bedi, N. Gupta, and V. Jindal, "Siam-IDS: handling class imbalance problem in intrusion detection systems using siamese neural network," *Procedia Computer Science*, vol. 171, pp. 780-789, 2020, doi: 10.1016/j.procs.2020.04.085.
- [17] A. K. Balyan *et al.*, "A hybrid intrusion detection model using EGA-PSO and improved random forest method," *Sensors*, vol. 22, no. 16, 2022, doi: 10.3390/s22165986.
- [18] L. Liu, P. Wang, J. Lin, and L. Liu, "Intrusion detection of imbalanced network traffic based on machine learning and deep learning," *IEEE Access*, vol. 9, pp. 7550-7563, 2021, doi: 10.1109/ACCESS.2020.3048198.
- [19] K. Jiang, W. Wang, A. Wang, and H. Wu, "Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network," *IEEE Access*, vol. 8, pp. 32464-32476, 2020, doi: 10.1109/ACCESS.2020.2973730.
- [20] A. Luque, A. Carrasco, A. Martín, and A. D. L. Heras, "The impact of class imbalance in classification performance metrics based on the binary confusion matrix," *Pattern Recognition*, vol. 91, pp. 216-231, 2019, doi: 10.1016/j.patcog.2019.02.023.
- [21] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, vol. 100, pp. 779-796, 2019, doi: 10.1016/j.future.2019.05.041.
- [22] S. Al and M. Dener, "STL-HDL: A new hybrid network intrusion detection system for imbalanced dataset on big data environment," *Computers & Security*, vol. 110, 2021, doi: 10.1016/j.cose.2021.102435.
- [23] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 2008, pp. 1322-1328, doi: 10.1109/IJCNN.2008.4633969.
- [24] R. Yao, N. Wang, Z. Liu, P. Chen, D. Ma, and X. Sheng, "Intrusion detection system in the smart distribution network: A feature engineering based AE-LightGBM approach," *Energy Reports*, vol. 7, pp. 353-361, 2021, doi: 10.1016/j.egy.2021.10.024.
- [25] I. S. Thaseen, V. Mohanraj, S. Ramachandran, K. Sanapala, and S. -S. Yeo, "A hadoop based framework integrating machine learning classifiers for anomaly detection in the internet of things," *Electronics*, vol. 10, no. 16, 2021, doi: 10.3390/electronics10161955.
- [26] X. Liu *et al.*, "NADS-RA: Network anomaly detection scheme based on feature representation and data augmentation," *IEEE Access*, vol. 8, pp. 214781-214800, 2020, doi: 10.1109/ACCESS.2020.3040510.
- [27] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785-794, doi: 10.1145/2939672.2939785.
- [28] I. Benmessahel, K. Xie, and M. Chellal, "A new evolutionary neural networks based on intrusion detection systems using multiverse optimization," *Applied Intelligence*, vol. 48, no. 8, pp. 2315-2327, 2018, doi: 10.1007/s10489-017-1085-y.

BIOGRAPHIES OF AUTHORS






Januar Al Amien    completed education bachelor's degree in the Informatics Engineering Department, STMIK-AMIK Riau. And master's degree in Master of Information Technology at Putra Indonesia University Padang. Now working as a lecturer in the Department of Computer Science, University Muhammadiyah of Riau. With research interests in the field of Machine learning algorithms and AI. He can be contacted at email: januaralamien@umri.ac.id.






Hadhrami Ab Ghani    received his bachelor degree in electronics engineering from Multimedia University Malaysia (MMU) in 2002. In 2004, he completed his masters degree in Telecommunication Engineering at The University of Melbourne. He then pursued his Ph.D. at Imperial College London in intelligent network systems and completed his Ph.D. in 2011. He can be contacted at email: hadhrami.ag@umk.edu.my.






Nurul Izrin Md Saleh    obtained a bachelor's degree in information technology from Multimedia University Malaysia (MMU). He completed his master's degree in computer science at The University of Putra Malaysia. Then complete a Ph.D. at The University of Brunel London in the same field of study. She can be contacted at email: nurulizrin.ms@umk.edu.my.



Edi Ismanto    completed education bachelor's degree in the Informatics Engineering Department, State Islamic University of Sultan Syarif Kasim Riau. And master's degree in Master of Computer Science at Putra Indonesia University Padang. Now working as a lecturer in the Department of Informatics, University Muhammadiyah of Riau. With research interests in the field of Machine learning algorithms and AI. He can be contacted at email: edi.ismanto@umri.ac.id.



Rahmad Gunawan    graduated with a bachelor's degree at Gunadarma University with a major in Information Management, a master's degree with Gunadarma University majoring in Electrical Telecommunication. And now works as a lecturer at the Faculty of Computer Science, University of Muhammadiyah Riau. With research interests in the field of Machine learning algorithms and AI. He can be contacted at email: goengoen78@umri.ac.id.