Machine Learning Based Classification of IoT Traffic

Bojana VELICHKOVSKA, Ana CHOLAKOSKA, Vladimir ATANASOVSKI

Faculty of Electrical Engineering and Information Technologies, Ss. Cyril and Methodius University,
Ruger Boshkovikj 18, 1000 Skopje, North Macedonia

{bojanav, acholak, vladimir}@feit.ukim.edu.mk

Submitted January 11, 2023 / Accepted May 2, 2023 / Online first May 17, 2023

Abstract. *With the rapid expansion and widespread adoption of the Internet of Things (IoT), maintaining secure connections among active devices can be challenging. Since IoT devices are limited in power and storage, they cannot perform complex tasks, which makes them vulnerable to different types of attacks. Given the volume of data generated daily, detecting anomalous behavior can be demanding. However, machine learning (ML) algorithms have proven successful in extracting complex patterns from big data, which has led to active applications in IoT.*

In this paper, we perform a comprehensive analysis, including 4 ML algorithms and 3 neural networks (NNs), and propose a pipeline which analyzes the influence data reduction (loss) has on the performance of these algorithms. We use random undersampling as a data reduction technique, which simulates reduced network traffic data. The pipeline investigates several degrees of data loss. The results show that models trained on the original data distribution obtain accuracy that verges on 100%. XGBoost performs best from the classic ML algorithms. From the deep learning models, the 2-layered NN provides excellent results and has sufficient depth for practical application. On the other hand, when the models are trained on the undersampled data, there is a decrease in performance, most notably in the case of NNs. The most prominent change is seen in the 4-layered NN, where the model trained on the original dataset detects attacks with a success of 93.53%, whereas the model trained on the maximally reduced data has a success of only 39.39%.

Keywords

Machine learning, deep learning, Internet of Things (IoT), intrusion detection, traffic modelling

1. Introduction

The continuous and exponential growth of the Internet of Things (IoT) created an automation-driven society, where unique devices are interconnected in order to improve the quality of healthcare, industry, transportation, etc. [1], [2].

IoT networks vary from the traditional approach of networking, which reflects in their traffic patterns [3]. Namely, IoT is characterized by highly homogeneous traffic, meaning all devices running the same application will exhibit similar behavior patterns. Additionally, since IoT networks are comprised of many devices, they generate massive amounts of data. Each device in the network is sensitive to an attack that causes a variation in the standard behavior of the network and creates significant problems for end-users. This is a time-sensitive issue, and therefore, it is essential to detect these network anomalies quickly.

Techniques to detect anomalous behavior are being extensively developed for many applications [4]. Certain techniques are based on analyzing network traffic characteristics essential to understanding specific traffic patterns, such as protocol design, network management, and resource distribution. This task can be time-consuming, limiting real-time device monitoring and requiring extreme expertise and knowledge. Consequently, this becomes a problem when it is necessary to address potential traffic anomalies swiftly. In the past few years, machine learning (ML) has slowly taken its place as an alternative approach to human-assisted traditional intrusion detection systems [5]. The main reason is that ML allows development of portable algorithms.

However, next-generation security systems should be able to offer real-time anomaly detection and adjust their knowledge according to changes in daily network traffic. Therefore, ML algorithms should be reinforced through time, according to changes in the active usage of IoT devices. Considering that end devices have limited memory, storing the entirety of their network traffic information can be challenging [6]. For this reason, occasionally it is necessary to understand if particular instances of network traffic can be excluded from the ML process.

In this paper, we perform a comprehensive analysis of IoT traffic classification by evaluating the performance of 4 ML algorithms and 3 deep learning (DL) neural networks (NNs) across different sampling strategies (simulating data loss). Our approach also investigates the needed NN depth (i.e., the number of layers) for successful traffic classification. The main reason for using ML and DL algorithms is that IoT devices generate large volumes of data. Therefore, analyz-

DOI: 10.13164/re.2023.0256

ing network traffic requires technologies capable of rapidly processing, analyzing, and comprehending massive volumes of data. Given the extensive application of ML in trending topic-related research, we focus on applying the most often used algorithms in order to determine patterns of anomalous traffic [7]. Our proposed pipeline successfully classifies anomalous traffic when models are trained both on the entire network traffic data and on randomly undersampled network traffic data. Unlike previous work, we investigate the impact which different degrees of random undersampling (simulating train data loss through different sampling strategies) have on the models' performance. This serves as an indication of the algorithm's behavior when traffic information is lost, and in turn, gives an idea of the volume of network traffic data which end devices (with small storage space) must keep for training a successful anomaly detection model.

This paper is organized as follows. Section 2 gives an overview of current research in the field. Section 3 details the pipeline used in the research, from the dataset and preprocessing of the data to the models and metrics used. Section 4 provides an overview of the results. Section 5 concludes the paper.

2. Related Work

Most of the research on anomaly detection in IoT networks focuses on supervised ML performed on entire datasets gathered from IoT devices in a controlled environment. Research has proved successful in the application of ML when dealing with the big data problem in network security. Applications of ML in network traffic analysis vary from working with supervised and unsupervised algorithms, to real-time applications.

Detailed systematic overviews of available datasets and state-of-the-art approaches in analyzing network traffic are being actively researched [8–10]. Research has focused on investigating the performance of widely-used ML algorithms such as SVM, Naive Bayes, Decision Tree, and Random Forest [11–13]. Generally, best performances were obtained from SVM [14] and XGBoost [15], with accuracy ranging from 89–99%. The authors in [16] suggest a supervised ML-based intrusion detection system (IDS) for IoT networks. After normalization and dimensionality reduction on the UNSW-NB15 dataset, 6 ML models were trained, obtaining an accuracy of 99%.

DL networks have the ability to use their hierarchical structure to create high-level features from the inputted raw data [17]. Proposed DL-based IDS can use Convolutional Neural Networks (CNNs), Deep Belief Networks (DBNs), Restricted Boltzmann Machines (RBMs), Recurrent Neural Networks (RNNs), and Deep Neural Networks (DNNs). Paper [18] presents a DBN-based IDS for an IoT environment which detects anomalous traffic and offers it as a service. The method was evaluated using raw traffic data. In [19] the authors propose a CNN-based IDS using an available, well-established dataset. The approach uses a CNN to extract the network traffic features, before applying supervised learning to detect the intrusions. DNN-based IDS approaches use data transformation and normalization before feeding the data into the model [20], [21].

Researchers have also addressed the issue of class imbalances in datasets. In [22] the authors compared the performance of different ML algorithms in the CSE-CIC-IDS2018 dataset. In order to balance the dataset, the authors addressed the imbalance by using Synthetic Minority Oversampling Technique (SMOTE) [23], which improved the performance for minority class attacks. However, there are proven instances where SMOTE can cause particular issues with the data [24], which is why we use different methodologies to address data imbalance issues.

In the next section, we give an overview of our approach, i.e., the section details the specifics of the dataset and data preprocessing performed. Additionally, we present an analysis of the changes which occur in the data during the preprocessing part of the pipeline. We also give an overview of the models used in the research as well as the metrics which evaluate the models.

3. Pipeline

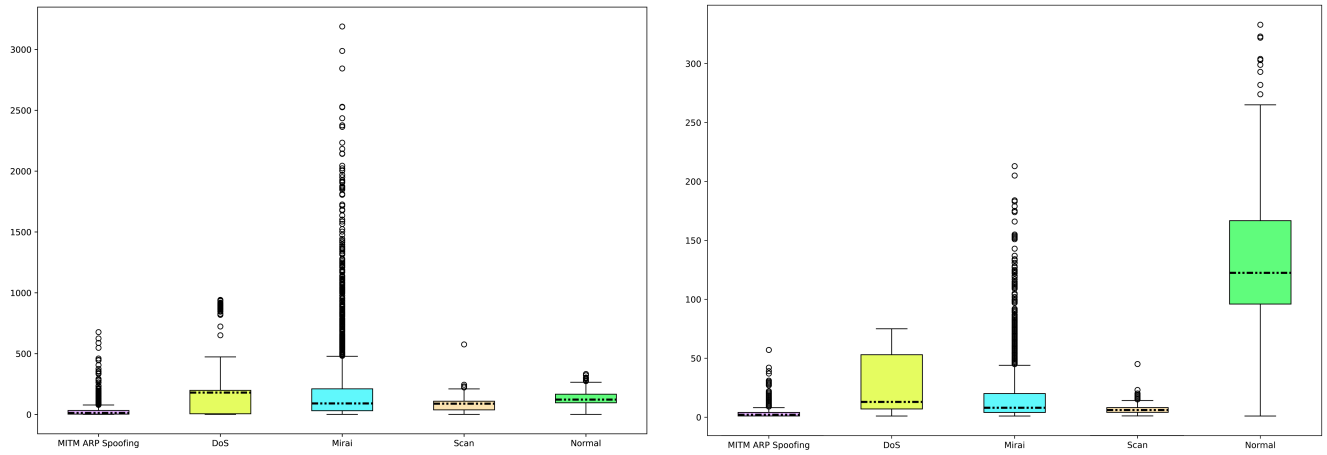
3.1 Dataset

For the purpose of this research, we use the IoTID20 dataset [25], which consists of 625,783 instances of network traffic generated in a smart home network. The dataset provides normal and anomalous network flow in a testbed environment consisting of a speaker, a smartphone, a security camera, and a laptop. The network traffic is described via 83 network features extracted from the traffic and 3 label features which illustrate binary, category, and sub-category traffic distributions.

3.2 Preprocessing

In order to prepare the dataset for the requirements of the models before training, we remove the instances with missing values. Additionally, we exclude the source and destination IPs provided, and we disregard the timestamp. From the 3 provided labels, we rely on the binary label of the network traffic data (normal or anomaly).

As can be seen from Tab. 1, the dataset contains 40,073 instances labeled as normal (or 6.4%) and 585,710 instances labeled as an anomaly (or 93.6%). The distribution of the instances shows significant data imbalance in favor of anomalous traffic. As a next step, we reduce anomalous instances and balance the normal and anomalous traffic using the RandomUnderSampler [26]. The balanced dataset, where we simulate data loss, kept the 40,073 instances of normal traffic, whereas only 40,073 randomly selected instances remained



(a) Before undersampling

(b) After undersampling

Fig. 1. Data distributions per attack type.

Binary level distribution	Type of attack	Number of instances
Normal	Normal	40,073
Anomaly	DoS	59,391
	Mirai	415,677
	MITM	35,377
	Scan	75,265

Tab. 1. Dataset overview: class distribution.

from the anomalous traffic. We investigate the change in the performance of the models between the original dataset and the balanced dataset. Additionally, we investigate the gradual changes in the models' performance by analyzing the models at several undersampling strategies (0, 0.1, 0.3, 0.5, 0.7, 1, where 0 is the original dataset, and 1 is the fully balanced dataset).

3.3 Statistical Analysis

In order to understand the impact which the RandomUnderSampler has on the data, we performed a statistical analysis of the data (before and after undersampling with a sampling strategy equal to 1), using box and whisker plots. The undersampling was applied to the anomalous traffic only. For each attack type, we plotted the time-wise data distribution in the original and the fully undersampled data. The data distribution is given in Fig. 1. Figure 1(a) shows the locality, spread, and skewness groups of the data before undersampling is performed. Figure 1(b) shows the data distribution after undersampling.

Since the undersampling process influenced only the anomalous traffic instances, there are no changes in the distribution of the normal traffic; the seeming difference results from the difference in the y-axis range. Both distributions show the DoS attacks are significantly skewed, but the skew is different between the two graphs. Another notable difference is the presence and lack of outliers in the DoS attacks. The presence of the outliers before preprocessing comes from combining two separate DoS attacks occurring on different days and analyzing them as one. Since the DoS attacks consist of flooding the target with network traffic, the attacks are an outlier in regular network traffic. However, they should not create outliers in their attack group, particularly considering the simulated nature of the dataset. Figure 1(b) shows how random undersampling contributes to removing the outliers in the DoS attacks.

The Mirai attacks have similar distributions in both stages of the analysis. However, after the undersampling, the data is more tightly grouped, as are the outliers. From both subfigures in Fig. 1 we can observe that the dispersal in both graphs is consistent for the Mirai attacks, including the skewing of the data. The changes in the MITM ARP Spoofing and the Scan attack groups are not as noticeable as with the other attack groups.

Since the undersampling is performed only on the predominant class (anomalous traffic), the distribution of the normal traffic remains unchanged. However, a difference can be noted in the relation between normal and anomalous traffic. Namely, the value range of normal traffic is smaller than both DoS and Mirai attacks before the undersampling, which is not the case after undersampling is performed.

3.4 Algorithms

ML and DL algorithms automate model building by learning from training data and identifying patterns to describe a requested outcome. Once trained, the algorithms can reapply the decision-making process on new and unseen data. The quality of the results depends on the quantity and quality of the data provided in the training process, the data preprocessing, and the algorithms used.

For the purpose of this research, we analyzed the results obtained from most frequently used ML algorithms. Additionally, we analyze the performance of DL methods, by training and evaluating 3 NN approaches. More specifically, the methods used are:

- Gaussian Naive Bayes (Gaussian NB) [27], is a supervised classification algorithm that follows the Gaussian normal distribution and is a simple classification technique, which has high functionality.
- Support Vector Classifier (SVC) [28], is an effective algorithm for localization of a distinct hyperplane for separating data points in high dimensional spaces.
- Random Forest [29], is an ensemble of decision trees, where each individual tree in the forest has a different structure and provides separate predictions. The most voted prediction is the models' final output.
- XGBoost [30], stands for Extreme Gradient Boosting and is a distributed gradient-boosted decision tree ensemble algorithm that provides a parallel tree boosting.
- NNs [31] consist of interconnected layers of nodes. Each node in a layer performs simple operations on the inputted data and later passes the results to the next layer until an output is created. The difference between the classical ML algorithms and the NNs is the interpretability of the models. Namely, NNs cannot be easily interpreted, since the model itself is a black box. Another distinction is that we use different NN architectures to analyze the depth required for successful IoT traffic classification.

The effectiveness of ML and DL algorithms is analyzed through evaluation metrics, so selecting the right metrics is important [32]. With classification methods, the metrics are extrapolated from the confusion matrix, i.e., true positives (TP), false negatives (FN), false positives (FP), and true negatives (TN). The confusion matrix gives an instance-wise explanation of the behavior of the model by means of a visual representation of each of the above-listed factors. This shows where the deterioration in results originates. An overview of the confusion matrix and its values as they are used in this paper are given in Tab. 2. Additionally, we evaluate the models using accuracy (which shows how many times the algorithm made an accurate prediction overall), precision (shows what portion of all identifications is correct), recall (shows what portion of actual values was identified correctly), and F1-score (which is a harmonic mean of precision and recall).

	Predicted Negative	Predicted Positive
Actual Negative	**TN (true negatives)** negative samples (attacks) the model predicted correctly	**FP (false positives)** negative samples (attacks) the model predicted as normal
Actual Positive	**FN (false negatives)** positive samples (normal traffic) the model predicted as attack	**TP (true positives)** positive samples (normal traffic) the model predicted correctly

Tab. 2. Confusion matrix (as used in this paper).

4. Results

The architecture of our approach is illustrated in Fig. 2. Initially, we preprocess the dataset by removing instances with missing values and scaling the data. From there, we split the data into training (80%) and testing (20%) datasets. Next, we apply sampling strategies to the training data. The selected sampling strategies are 0 (original dataset), 0.1, 0.3, 0.5, 0.7, and 1 (fully balanced dataset). With each sampled dataset we train the models again, and use the testing data is used to evaluate the trained models. This allows us to compare the difference in model performance when the original distribution of network traffic is kept and when the anomalous traffic is reduced.

We trained and evaluated 4 classic ML algorithms and 3 DL NNs (a shallow NN with 1 hidden layer and two additional NNs with 2 and 4 hidden layers respectively). Each of the NNs was optimized using stochastic gradient descent and had binary cross-entropy as a loss function. The training process for each network lasted 25 epochs. Each epoch used a fixed batch size consisting of 64 instances. We chose a dense NN because each layer provides learning features from all the combinations of the features of the previous layer. When working with tabular data, one to five hidden layers are most often enough to solve problems without overfitting. This is why we chose 3 models: shallow NN (with 1 hidden layer), 2-layered NN and 4-layered NN. Additionally, we reduce the

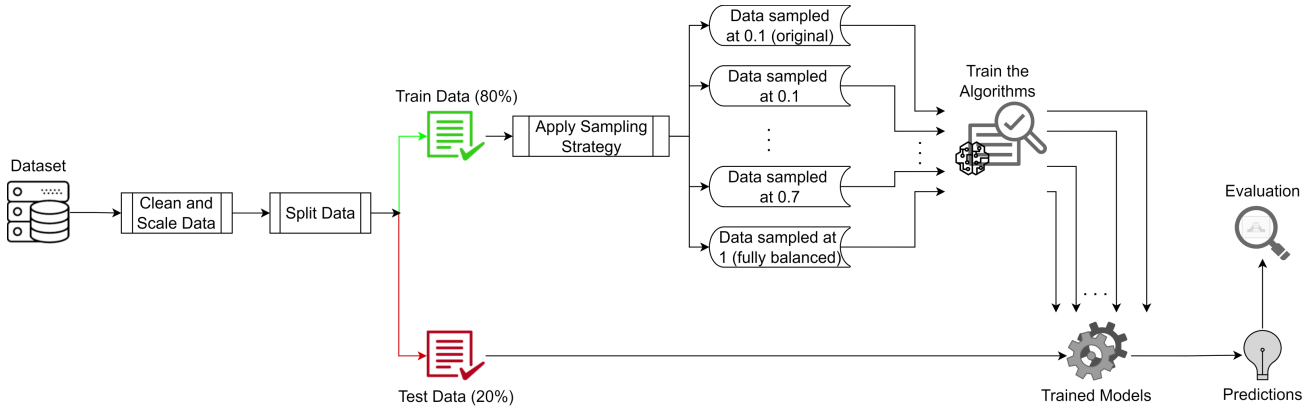


Fig. 2. Proposed pipeline architecture.

number of neurons as the layers progress, because this way the network can learn a lot of lower-level features and later use them in subsequent layers to create higher-level features.

The results obtained from the classic ML approaches and the NNs for the original and fully balanced datasets are presented in Tab. 3. The Gaussian NB showed the worst performance of all models, with nearly half of the anomalous activity being misclassified as normal traffic, from both the original and undersampled training. The best overall performance is displayed by XGBoost, with an accuracy approaching 100%, whereas Random Forest and SVC show slightly weaker performance. The overperformance which can be noted in the case of XGBoost trained with the original data is largely due to the fact that the dataset used for the study is a simulated one. Additionally, the test dataset we are using to evaluate our models is a randomly-selected portion of the original data. This significantly increases the possibility of high data correlation between the train and test data, which also contributes to the overperformance of the approach.

The performance of the 3 NNs when trained with the original dataset shows an accuracy approaching 100% in all 3 cases. This is not the case when we train the NNs on the undersampled data. Namely, when training with the undersampled data there is significant skewness in the performance of the models. The results with the undersampled data are consequently less than satisfactory for all 3 NNs.

In order to understand the drop in the performance of the NNs, we observed the behavior of the loss function. The initial loss values suggested that the model was randomly guessing the output. Since this is standard behavior for randomly initialized NNs, this did not explain the behavior of the model. Another potential issue we tested for was overfitting. Overfitting occurs when the model fits exactly against the training data and performs inaccurately against unseen data. We evaluated the performance of the trained models both on the data used for training and the data used for testing. The results showed that the model performed similarly when tested on the train and on the test data, which ruled out overfitting as an explanation for the behavior of the NNs. The only remaining cause for the obtained results rests in the structure of NNs. Namely, NNs have high complexity and a high number of parameters that need to be calculated during the training phase. Therefore, with reduced training data, the model cannot properly adjust its parameters and underperforms. Considering that all 3 NNs show worse results with the undersampled data, this phenomenon occurs with all 3 configurations.

The undersampling of the dataset does not significantly influence the overall results of the classic ML algorithms. When these results are compared to those obtained on the original data, there is a change in the confusion matrix before and after the random undersampling is performed. Namely, from the change in the upper right corner of the confusion matrix it can be seen that by undersampling the anomalous traffic, the models lose crucial information on the attacks. This results in more of the anomalous traffic being classified as normal network traffic. This distinction is pronounced in the results of the SVC and the Random Forest algorithms. However, XGBoost handles the data loss better and there are only minor differences present.

When observing the results obtained with the original distribution kept in the training dataset, the best performance can be analyzed from two distinct points. If the least number of misclassified instances is considered, then XGBoost shows the best results. However, if the aim of the model is to never miss an attack in the network traffic, the best performance can be seen by Random Forest. A disadvantage to this benefit is that the Random Forest model classifies a significant portion of normal traffic as anomalous, resulting in the detection of malicious traffic when there is none. Depending on the trade-off required here, it would be easy to argue that the 5 misclassifications by XGBoost are not as significant because these 5 instances are in accompaniment with corresponding anomalous traffic, meaning those instances would trigger the detection system.

4.1 Sampling Strategy Influence on the Models

We wanted to investigate the amount of data undersampling which can be performed before a significant distinction in misclassified anomalies can be seen. The test was performed at 6 stages: the first with no undersampling, and the remaining 5 with sampling strategies for the RandomUnderSampler (0.1, 0.3, 0.5, 0.7, 1, where 1 represents an equal number of samples in both classes or fully balanced dataset).

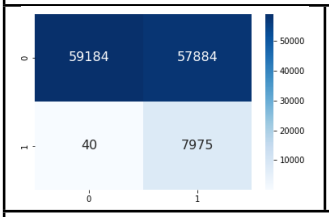
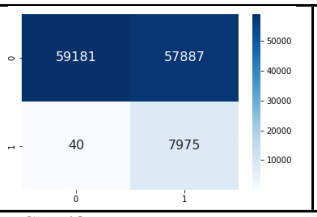
The results are given in Tab. 4. From the table, it can be seen that with the GaussianNB there is very little change in the behavior of the classifier, but that is a consequence of the classifier underperforming even when training with the entire dataset. The behavior of the SVC, RF, and XGBoost is similar. The attacks classified as attacks decrease as the sample strategy increases (which is to be expected since increasing the sampling strategy means that the number of attack samples decreases and the model has fewer attack samples to learn from). The decrease is steeper with SVC and RF, where with a sample strategy of 1 the percentage of correctly classified attacks is < 90%, whereas XGBoost reaches a minimum of 93.43% (from 93.59% total attacks). This shows that the resistance against undersampling which XGBoost has is better compared to the other two classifiers.

However, the results we obtained from the NNs show that even with minor undersampling the networks cannot successfully learn how to classify an attack and classify a significant portion of the anomalies as normal traffic. From Tab. 4 we can see that all NNs underperformed once training data was reduced. As we previously discussed, the cause of this behavior is the fact that NNs need to tune a lot of parameters, and therefore struggle with data loss. This observation is further strengthened by the steady decline in the percentages for accurately classified instances presented for

Original

Undersampled

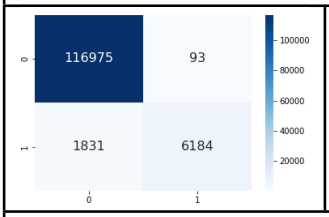
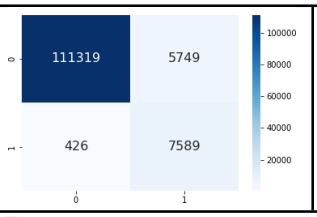
Gaussian Naive Bias

Original Confusion Matrix	Original Classification Report	Undersampled Confusion Matrix	Undersampled Classification Report
	Accuracy: 0.54		Accuracy: 0.54

	Attack	Normal
Precision	1.0	0.12
Recall	0.51	1.0
F1-score	0.67	0.22

	Attack	Normal
Precision	1.0	0.12
Recall	0.51	1.0
F1-score	0.67	0.22

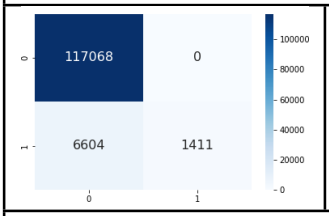
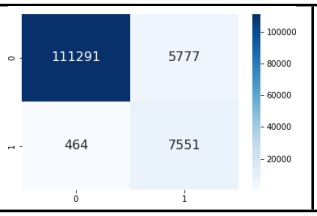
Support Vector Classifier

Original Confusion Matrix	Original Classification Report	Undersampled Confusion Matrix	Undersampled Classification Report
	Accuracy: 0.98		Accuracy: 0.95

	Attack	Normal
Precision	0.98	0.99
Recall	1.0	0.77
F1-score	0.99	0.87

	Attack	Normal
Precision	1.0	0.57
Recall	0.95	0.95
F1-score	0.97	0.71

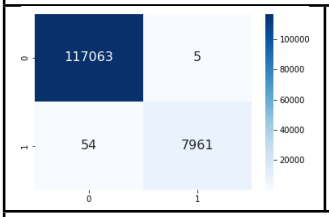
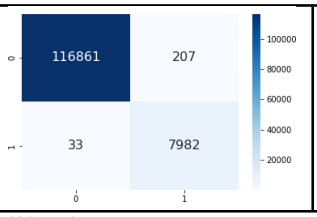
Random Forest

Original Confusion Matrix	Original Classification Report	Undersampled Confusion Matrix	Undersampled Classification Report
	Accuracy: 0.95		Accuracy: 0.95

	Attack	Normal
Precision	0.95	1.0
Recall	1.0	0.18
F1-score	0.97	0.30

	Attack	Normal
Precision	1.0	0.57
Recall	0.95	0.94
F1-score	0.97	0.71

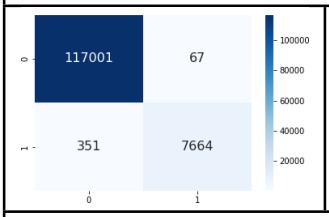
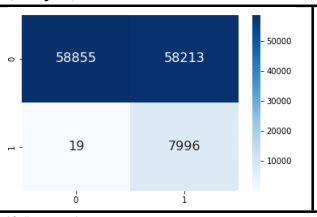
XGBoost

Original Confusion Matrix	Original Classification Report	Undersampled Confusion Matrix	Undersampled Classification Report
	Accuracy: 1.0		Accuracy: 1.0

	Attack	Normal
Precision	1.0	1.0
Recall	1.0	0.99
F1-score	1.0	1.0

	Attack	Normal
Precision	1.0	0.97
Recall	1.0	1.0
F1-score	1.0	0.99

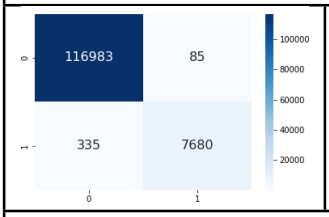
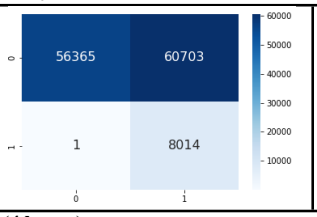
Neural Net (1 layer)

Original Confusion Matrix	Original Classification Report	Undersampled Confusion Matrix	Undersampled Classification Report
	Accuracy: 1.0		Accuracy: 0.53

	Attack	Normal
Precision	1.0	0.99
Recall	1.0	0.96
F1-score	1.0	0.97

	Attack	Normal
Precision	1.0	0.12
Recall	0.5	1.0
F1-score	0.67	0.22

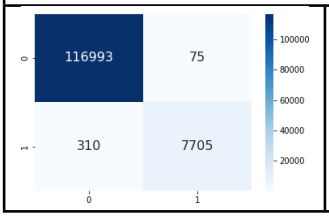
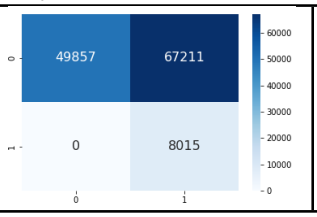
Neural Net (2 layers)

Original Confusion Matrix	Original Classification Report	Undersampled Confusion Matrix	Undersampled Classification Report
	Accuracy: 1.0		Accuracy: 0.51

	Attack	Normal
Precision	1.0	0.99
Recall	1.0	0.96
F1-score	1.0	0.97

	Attack	Normal
Precision	1.0	0.12
Recall	0.48	1.0
F1-score	0.65	0.21

Neural Net (4 layers)

Original Confusion Matrix	Original Classification Report	Undersampled Confusion Matrix	Undersampled Classification Report
	Accuracy: 1.0		Accuracy: 0.46

	Attack	Normal
Precision	1.0	0.99
Recall	1.0	0.96
F1-score	1.0	0.98

	Attack	Normal
Precision	1.0	0.11
Recall	0.43	1.0
F1-score	0.6	0.19

Tab. 3. Results from training on original (left) and undersampled (right) dataset, with a confusion matrix and a classification report for each. The test dataset contains 125,083 samples, where attack samples are 117,068 (or 93.59%) and normal samples are 8,015 (or 6.41%).

Sampling Strategy	Type	Algorithms						
		Gaussian Naive Bias	Support Vector Classifier	Random Forest	XGBoost	Neural Net (1 layer)	Neural Net (2 layers)	Neural Net (4 layers)
0	AA	47.32	93.52	93.59	93.59	93.54	93.53	93.53
	AN	46.28	0.07	0	0	0.05	0.06	0.06
	NA	0.03	1.46	5.28	0.04	0.28	0.27	0.25
	NN	6.38	4.94	1.13	6.36	6.13	6.14	6.16
0.1	AA	47.32	93.4	93.59	93.59	47.72	43.68	40.35
	AN	46.28	0.2	0	0.01	45.87	49.91	53.24
	NA	0.03	1.35	4.65	0.04	0.02	0	0
	NN	6.38	5.06	1.76	6.37	6.39	6.41	6.41
0.3	AA	47.32	91.88	93.54	93.56	46.95	45.21	41.64
	AN	46.28	1.71	0.05	0.04	46.64	48.38	51.96
	NA	0.03	0.83	2.04	0.03	0.01	0	0
	NN	6.38	5.58	4.37	6.38	6.4	6.41	6.41
0.5	AA	47.32	89.52	93.28	93.53	47.37	45.46	40
	AN	46.28	4.07	0.31	0.06	46.22	48.13	53.59
	NA	0.0	0.47	1.39	0.03	0.01	0	0
	NN	6.37	5.94	5.02	6.38	6.4	6.41	6.41
0.7	AA	47.32	89.3	91.75	93.49	46.68	42.73	41.15
	AN	46.28	4.3	1.84	0.1	46.91	50.86	52.44
	NA	0.03	0.4	0.91	0.03	0.01	0	0
	NN	6.38	6.01	5.5	6.38	6.4	6.41	6.41
1	AA	47.31	89	88.97	93.43	47.05	45.06	39.39
	AN	46.28	4.6	4.62	0.17	46.54	48.53	54.2
	NA	0.03	0.34	0.37	0.03	0.02	0	0
	NN	6.38	6.07	6.04	6.38	6.39	6.41	6.41

Tab. 4. The effect of sampling strategy in the results expressed in percentages from the total test dataset. The test dataset contains 125,083 samples, where attack samples are 117,068 (or 93.59%) and normal samples are 8,015 (or 6.41%). Type meanings: AA (attacks classified as attacks), AN (attacks classified as normal), NA (normal classified as attacks), NN (normal classified as normal).

the NNs in Tab. 4. Namely, the percentages of misclassified instances increase proportionately with the increase in sampling strategy and the increase in the complexity of the NN. To consider only one example from Tab. 4, if we observe sampling strategy 1 and the values obtained for attacks misclassified as normal traffic, we can see that the values grow as the complexity of the network increases. More specifically, with the shallow NN the value of misclassified instances is 46.54% of the entire test dataset, which grows to 54.2% with the 4-layered NN. Moreover, similar changes can be observed along sampling strategies. This shows that as the complexity of the network grows and as the data gets reduced, the models struggle with tuning their parameters.

5. Conclusion

This paper proposes an approach to investigate differences between anomalous and normal network traffic and the influence of data reduction on the performance of 4 classic ML algorithms and 3 NNs. The obtained results show that XGBoost provides the best overall performance, both before and after undersampling the training data. The performance of the NNs when trained on the original data distribution is comparable with XGBoost. However, when the NNs are trained on sampled data they misclassify a significant portion of the anomalous traffic. This matches the fact that NNs require big data for good performance.

Further research will expand on the results obtained here in two aspects. First, we will expand the research from binary to multiclass classification and observe if the same behavior can be noted there. Second, since the dataset used in this research is simulated, we will use real-time data obtained from network traffic and observe if the same accurate performance can be obtained there as well. Additionally, the approach can be expanded to include new types of attacks in order to give a more encompassing solution. The sampling strategy had intriguing results with simulated data. Therefore, we will perform an in-depth analysis of the behavior of data sampling in real-time data and see if the results will hold there.

References

[1] JASKARAN, KHAN, N., NANDINI, et al. IOT: Applications, challenges and latest trends. *1st IEEE International Conference on Industrial Electronics: Developments & Applications (ICIDeA)*. Bhubaneswar (India), 2022, p. 181–186. DOI: 10.1109/ICIDeA53933.2022.9970100

[2] GUPTA, S., TANWAR, S., GUPTA, N. A systematic review on internet of things (IoT): Applications & challenges. *10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*. Noida (India), 2022, p. 1–7. DOI: 10.1109/ICRITO56286.2022.9964892

[3] HUSSEIN, D. H., IBNKAHLA, M. An IoT traffic modeling framework and its application to autonomous edge scaling. *IEEE Global Communications Conference (GLOBECOM)*. Rio de Janeiro (Brazil), 2022, p. 5656–5661. DOI: 10.1109/GLOBECOM48099.2022.10000950

[4] THAMILARASU, G., ODESILE, A., HOANG, A. An intrusion detection system for internet of medical things. *IEEE Access*, 2020, vol. 8, p. 181560–181576. DOI: 10.1109/ACCESS.2020.3026260

[5] PRADHAN, M., MOHANTY, S., SEEMONA, A. O. Machine learning-based intrusion detection system for the internet of vehicles. In *5th International Conference on Computational Intelligence and Networks (CINE)*. Bhubaneswar (India), 2022, p. 1–6. DOI: 10.1109/CINE56307.2022.10037357

[6] LI, S., LU, Y., LI, J. CAD-IDS: A cooperative adaptive distributed intrusion detection system with fog computing. In *IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. Hangzhou (China), 2022, p. 635–640. DOI: 10.1109/CSCWD54268.2022.9776147

[7] NASSIF, A., TALIB, M., NASIR, Q., et al. Machine learning for anomaly detection: A systematic review. *IEEE Access*, 2021, vol. 9, p. 78658–78700. DOI: 10.1109/ACCESS.2021.3083060

[8] AMARUDIN, FERDIANA, R., WIDYAWAN. A systematic literature review of intrusion detection system for network security: Research trends, datasets and methods. In *Proceedings of the 4th International Conference On Informatics And Computational Sciences (ICICoS)*. Semarang (Indonesia), 2020, p. 1–6. DOI: 10.1109/ICICoS51170.2020.9299068

[9] ASHRAF, E., AREED, N., SALEM, M. H., et al. IoT based intrusion detection systems from the perspective of machine and deep learning: A survey and comparative study. *Delta University Scientific Journal*, 2022, vol. 5, no. 2, p. 367–386. DOI: 10.21608/dusj.2022.275552

[10] KUMAR, S., GUPTA, S., ARORA, S. Research trends in network-based intrusion detection systems: A review. *IEEE Access*, 2021, vol. 9, p. 157761–157779. DOI: 10.1109/ACCESS.2021.3129775

[11] TAHRI, R., BALOUKI, Y., JARRAR, A., et al. Intrusion detection system using machine learning algorithms. *ITM Web Conference*, 2022, vol. 46, p. 1–4. DOI: 10.1051/itmconf/20224602003

[12] JARADAT, A. S., BARHOUSH, M. M., BANI EASA, R. S. Network intrusion detection system: Machine learning approach. *Indonesian Journal of Electrical Engineering and Computer Science*, 2022, vol. 25, no. 2, p. 1151–1158. DOI: 10.11591/ijeecs.v25.i2.pp1151-1158

[13] JMILA, H., KHEDER, M. Adversarial machine learning for network intrusion detection: A comparative study. *Computer Networks*, 2022, vol. 214, p. 1–14. DOI: 10.1016/j.comnet.2022.109073

[14] ADITYA, R., NUHA, H. H., PRABOWO, S. Intrusion detection using support vector machine on internet of things dataset. In *IEEE International Conference on Communication, Networks and Satellite (COMNETSAT)*. Solo (Indonesia), 2022, p. 62–66. DOI: 10.1109/COMNETSAT56033.2022.9994392

[15] LE, T.-T.-H., OKTIAN, Y. E., KIM, H. XGBoost for imbalanced multiclass classification-based industrial internet of things intrusion detection systems. *Sustainability*, 2022, vol. 14, no. 14, p. 1–21. DOI: 10.3390/su14148707

[16] SAHEED, Y. K., ABIODUN, A. I., MISRA, S., et al. A machine learning-based intrusion detection for detecting internet of things network attacks. *Alexandria Engineering Journal*, 2022, vol. 61, no. 12, p. 1–15. DOI: 10.1016/j.aej.2022.02.063

[17] ALEESA, A., ZAIDAN, B., ZAIDAN, A., et al. Review of intrusion detection systems based on deep learning techniques: Coherent taxonomy, challenges, motivations, recommendations, substantial analysis and future directions. *Neural Computing and Applications*, 2020, vol. 32, no. 14, p. 9827–9858. DOI: 10.1007/s00521-019-04557-3

[18] THAMILARASU, G., CHAWLA, S. Towards deep-learning-driven intrusion detection for the Internet of Things. *Sensors*, 2019, vol. 19, no. 9, p. 1–19. DOI: 10.3390/s19091977

[19] XIAO, Y., XING, C., ZHANG, T., et al. An intrusion detection model based on feature reduction and convolutional neural networks. *IEEE Access*, 2019, vol. 7, p. 42210–42219. DOI: 10.1109/ACCESS.2019.2904620

[20] AWAJAN, A. A novel deep learning-based intrusion detection system for IoT networks. *Computers*, 2023, vol. 12, no. 12, p. 1–17. DOI: 10.3390/computers12020034

[21] IKHWAN, S., WIBOWO, A., WARSITO, B. Intrusion detection using deep neural network algorithm on the internet of things. In *IEEE International Conference on Communication, Networks and Satellite (COMNETSAT)*. Solo (Indonesia), 2022, p. 84–87. DOI: 10.1109/COMNETSAT56033.2022.9994499

[22] KARATAS, G., DEMIR, O. SAHINGOZ, O. K. Increasing the performance of machine learning-based IDSs on an imbalanced and up-to-date dataset. *IEEE Access*, 2020, vol. 8, p. 32150–32162. DOI: 10.1109/ACCESS.2020.2973219

[23] CHAWLA, N. V., BOWYER, K. W., HALL, L. O., et al. SMOTE synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 2002, vol. 16, p. 321–357. DOI: 10.1613/jair.953

[24] KUMAR, S. *Stop Using SMOTE to Handle All Your Imbalanced Data*. 2021. [Online] Cited 2023-01-04. Available at: https://towardsdatascience.com/stop-using-smote-to-handle-all-your-imbalanced-data-34403399d3be

[25] ULLAH, I., MAHMOUD, Q. A scheme for generating a dataset for anomalous activity detection in IoT networks. In *Proceedings of the 33rd Canadian Conference on Artificial Intelligence*. Ottawa (Canada), 2020, p. 508–520. DOI: 10.1007/978-3-030-47358-7_52

[26] KRAIEM, M. S., SÁNCHEZ-HERNÁNDEZ, F., MORENO-GARCÍA, M. N. Selecting the suitable resampling strategy for imbalanced data classification regarding dataset properties. An approach based on association models. *Applied Sciences*, 2021, vol. 11, no. 18, p. 1–26. DOI: 10.3390/app11188546

[27] ISLAM, R., DEVNATH, M. K., SAMAD, M. D., et al. GGNB: Graph-based Gaussian naive Bayes intrusion detection system for CAN bus. *Vehicular Communications*, 2022, vol. 33, p. 1–11. DOI: 10.1016/j.vehcom.2021.100442

[28] NANTHIYA, D., KEERTHIKA, P., GOPAL, S. B., et al. SVM based DDoS attack detection in IoT using Iot-23 botnet dataset. In *Innovations in Power and Advanced Computing Technologies (i-PACT)*. Kuala Lumpur (Malaysia), 2021, p. 1–7. DOI: 10.1109/i-ACT52855.2021.9696569

[29] KURNIABUDI, STIAWAN, D., DARMAWIJOYO, et al. Improvement of attack detection performance on the internet of things with PSO-search and random forest. *Journal of Computational Science*, 2022, vol. 64, p. 1–13. DOI: 10.1016/j.jocs.2022.101833

[30] FAYSAL, J. A., MOSTAFA, S. T., TAMANNA, J. S., et al. XGB-RF: A hybrid machine learning approach for IoT intrusion detection. *Telecom*, 2022, vol. 3, no. 1, p. 52–69. DOI: 10.3390/telecom3010003

[31] NASCIMENTO, N., ALENCAR, P., COWAN, D. A lifecycle for engineering IoT neural network-based systems. In *IEEE International Conference on Big Data (Big Data)*. Orlando (FL, USA). 2021, p. 2427–2433. DOI: 10.1109/BigData52589.2021.9671413

[32] GYAMFI, E., ANCA, J. Intrusion detection in internet of things systems: A review on design approaches leveraging multi-access edge computing, machine learning, and datasets. *Sensors*, 2022, vol. 22, no. 10, p. 1–33. DOI: 10.3390/s22103744