IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received January 12, 2020, accepted January 29, 2020, date of publication February 11, 2020, date of current version February 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2973219

Increasing the Performance of Machine Learning-Based IDSs on an Imbalanced and Up-to-Date Dataset

GOZDE KARATAS[1], **(Member, IEEE), ONDER DEMIR**[2], **AND OZGUR KORAY SAHINGOZ**[3]
[1]Department of Mathematics and Computer Sciences, Faculty of Science and Literature, Istanbul Kültür University, 34158 Istanbul, Turkey
[2]Department of Computer Engineering, Faculty of Technology, Marmara University, 34722 Istanbul, Turkey
[3]Department of Computer Engineering, Faculty of Engineering, Istanbul Kültür University, 34158 Istanbul, Turkey

Corresponding author: Ozgur Koray Sahingoz (o.sahingoz@iku.edu.tr; sahingoz@gmail.com)

This work was supported in part by Marmara University Research Project, FEN-C-DRP-110718-0408, [C-DRP], Deep Learning-Based Intrusion Detection System. The authors, therefore, acknowledge with thanks Marmara University for technical and financial support.

ABSTRACT In recent years, due to the extensive use of the Internet, the number of networked computers has been increasing in our daily lives. Weaknesses of the servers enable hackers to intrude on computers by using not only known but also new attack-types, which are more sophisticated and harder to detect. To protect the computers from them, Intrusion Detection System (IDS), which is trained with some machine learning techniques by using a pre-collected dataset, is one of the most preferred protection mechanisms. The used datasets were collected during a limited period in some specific networks and generally don't contain up-to-date data. Additionally, they are imbalanced and cannot hold sufficient data for all types of attacks. These imbalanced and outdated datasets decrease the efficiency of current IDSs, especially for rarely encountered attack types. In this paper, we propose six machine-learning-based IDSs by using K Nearest Neighbor, Random Forest, Gradient Boosting, Adaboost, Decision Tree, and Linear Discriminant Analysis algorithms. To implement a more realistic IDS, an up-to-date security dataset, CSE-CIC-IDS2018, is used instead of older and mostly worked datasets. The selected dataset is also imbalanced. Therefore, to increase the efficiency of the system depending on attack types and to decrease missed intrusions and false alarms, the imbalance ratio is reduced by using a synthetic data generation model called Synthetic Minority Oversampling TEchnique (SMOTE). Data generation is performed for minor classes, and their numbers are increased to the average data size via this technique. Experimental results demonstrated that the proposed approach considerably increases the detection rate for rarely encountered intrusions.

INDEX TERMS IDS, intrusion detection, SMOTE, machine learning, CSE-CIC-IDS2018, imbalanced dataset.

I. INTRODUCTION

Due to technological developments, most of the real-world transactions have been made available in the cyber world. Thus, many operations, such as banking, shopping, online examinations, electronic commerce, and communication are used extensively within this new environment. With the widespread use of smartphones, people can connect to this global network and perform transactions at any time and from anywhere. Although this digitalization facilitates the daily work of human beings, due to the weakness of the servers and the newly emerged intrusion techniques, networks are often attacked by the intruders who take advantage of the anonymous nature of the Internet not only to steal some information or money but also to slow down the operation of network services.

Security administrators traditionally prefer password protection mechanisms, encryption techniques, and access controls in addition to firewalls as a means of protecting the network. However, these techniques are not sufficient for protecting the system. Therefore, many administrators prefer the use of Intrusion Detection Systems (IDSs) to detect malicious attacks by monitoring network traffic, as depicted in Fig. 1.

Intrusion can be defined as any kind of unauthorized activity that causes damage to confidentiality, availability, or integrity of the data within an information system.

The associate editor coordinating the review of this manuscript and approving it for publication was Vicente Alarcon-Aquino.

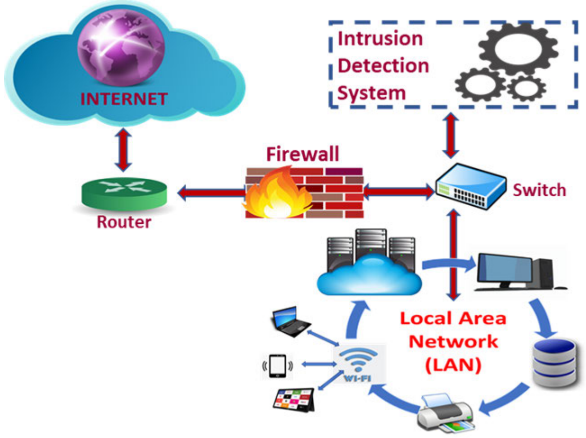


FIGURE 1. Intrusion detection systems and local area networks.

IDSs are a highly preferred means of detecting this type of activity. IDSs can be categorized into three groups: Signature-based Intrusion Detection Systems (SIDS), Anomaly-based Intrusion Detection Systems (AIDS), Hybrid Systems.

SIDSs store the signatures of the malicious activities in a knowledge base and try to detect intrusions by using pattern matching techniques. Meanwhile, AIDSs try to learn the normal behaviors of the activities and classify the others as suspicious. In this type of system, there is no need to use a signature-base, and the system can identify zero-day attacks that have not been encountered previously. Hybrid systems are composed composed by the integration of SIDS and AIDS to increase the detection rate of known malicious activities by reducing the false positive rate of zero-day attacks.

Due to the advantages of AIDSs, most current IDSs either directly use an AIDS or benefit from it within a hybrid approach. These IDSs need to be trained via machine learning model by processing the dataset. Most of the works on this topic adopted old datasets, which contain redundant information and imbalanced volumes of data types. Although we can encounter some new datasets that contain up-to-date data, the imbalanced size of data types is still a challenge for researchers.

The efficiency of and IDS is directly related to the selected learning model and the quality of the dataset used. A good quality dataset can be defined as a dataset that improves better performance metrics in real-world transactions. As mentioned in [1], [2] imbalanced datasets present a problem to researchers. A dataset is said to be imbalanced when the distribution of classes is not uniform [3]. This is a common problem in many of the classification problems due to the used datasets. Imbalanced dataset results the used classifier biases towards the majority class; however, in most of them, the aim is trying to detect the minority class [4]. This results in a large classification error over the minority class samples and main targets can be missed. To increase the quality of the dataset, it should be balanced according to data types. Therefore, in this paper, we aim to use an up-to-date dataset for training the IDS to develop a realistic knowledge base for the detection of an anomaly. To enhance the efficiency of the system, comparative work is done employing six different machine learning algorithms. To increase the detection rate of the low-sampled attack types, a synthetic data generation tool is used, and the results obtained in the present work are compared with those of previous experiments.

The rest of this paper is organized as follows. In the next section, a literature review on the topic of interest is provided. Section III depicts a comparative study of previously used datasets on IDSs. The design details of the proposed system with the chosen machine learning algorithms are explained, and the experimental results are discussed in Section IV and Section V, respectively. Finally, conclusions are drawn, and directions for future works are suggested in Section VI.

II. RELATED WORKS

Intrusion Detection Systems are striking areas not only for cybersecurity research but also for academic research. Over the past several years, many papers have been published on this topic. In this section, these noteworthy pieces of research (especially related to imbalanced datasets) are discussed briefly.

In 2019, Gao et al. used NSL-KDD dataset to test and develop an IDS by using an adaptive ensemble learning model [5]. They used four different algorithms; Decision Tree, Random Forest, K-Nearest Neighbor, and Deep Neural Networks. Also, they designed an ensemble adaptive voting algorithm. They used an NSL-KDD-Test+ file to verify their approach. The accuracy of the Decision Tree algorithm is 84.2% and the final accuracy of the adaptive algorithm is 85.2%. In the end, they compared related research papers, and they found that their ensemble adaptive model improves detection accuracy.

An online oversampling Principal Component Analysis (PCA) designed to address the anomaly detection problem is proposed in [6]. Their approach focuses on using online platforms for large-scale problems. By oversampling the minority class of the target instance, their proposed algorithm allows them to determine the anomaly of the target instance. A comparison between the PCA and other detection algorithms supported, the applicability, efficiency, and accuracy of the proposed method. Also, their algorithm reduced computational costs and memory requirements.

Yueai and Junjie proposed a two-stage strategy with a load balancing model (such as online and offline phase) to implement an IDS [7]. In the online phase, the system captured packets from the network and then detected intrusion. Meanwhile, in the offline phase, the training dataset was used to make an offline model. They used SMOTE for oversampling and made their classifications with AdaBoost and Random Forest algorithms. Their experimental results showed that SMOTE and AdaBoost did not work well.

Abdulhammed et al. (2019) used the CIDDS-001 dataset for handling imbalanced datasets to build an efficient IDS through various techniques [8]. They effectively studied the sampling methods of CIDDS-001 and evaluated this dataset through Voting, Deep Neural Networks, Variational Autoencoder, Random Forest, and stacking machine learning algorithms. This system detected attacks with 99.99% accuracy when using an imbalanced dataset.

A hybrid approach for IDSs with the NSL-KDD dataset was studied in [9]. Their approach involved a combination of SMOTE, cluster centers, and nearest neighbors. They selected important features using the leave one out method. K-Fold Cross Validation (K is 10) was used for measurement purposes. Experimental results showed that the proposed method achieved acceptable accuracy and a low false alarm rate, which don't have significant differences.

In 2019, Taher et al. proposed a supervised machine learning system to classify network traffic [10]. They used the NSL-KDD dataset for testing and training because they wanted to detect whether traffic was malicious or normal. For that purpose, they used Support Vector Machine (SVM) and Artificial Neural Network (ANN) algorithms and feature selection methods. They found that the ANN with feature selection performed better than the SVM.

Tesfahun and Bhaskari applied SMOTE to the training dataset and feature selection method based on Information Gain in NSL-KDD [11]. This study was carried out to deal with imbalanced datasets in IDSs. Random Forest algorithm was used as a classifier for the proposed method. Their experimental results showed that the Random Forest algorithm with SMOTE and information gain based feature selection performs well.

Chandra et al. proposed a hybrid model using the KDD Cup99 dataset in 2019 [12]. They used Filter-Based Attribute Selection to reduce the feature dimension of the dataset. K-Means and Sequential Minimal Optimization algorithms were used for detecting attacks in the dataset. Their proposed method significantly improves the accuracy rate.

In 2012, Qazi and Raza studied the effect attributes' selections for increasing the efficiency of the classification [13]. Also, they used undersampling and oversampling to decrease the imbalance ratio of the dataset. They used SMOTE for oversampling. They found that for the imbalanced datasets, the sampling technique is more accurate than SMOTE for classifying minority classes. It was also found that the Decision Tree and Naive Bayes algorithms are more accurate than other algorithms.

Al-issa et al. (2019) implemented Decision Tree (DT) and Support Vector Machine (SVM) algorithms for detecting attack signatures using a specific dataset [14]. The dataset contained regular profiles and several DoS scenarios in wireless sensor networks. The results showed that the DT achieved a lower false positive rate and higher true positive rate than SVM, as the DT has 99.86% and SVM has 99.62% true positive rate, and the DT has 0.05%, and SVM has 0.09%, false-negative rate.

In 2018, Ahmad et al. conducted a comparative study that resolved problems associated with accuracy and related metrics using Random Forest, Support Vector Machine, and Extreme Learning algorithms [15]. The NSL-KDD dataset was used, which is considered a benchmark for the evaluation of IDSs. The results show that the Extreme Learning Algorithm is better than other algorithms in terms of precision, recall, and accuracy.

TABLE 1. Related work.

Ref.	Dataset	Algorithms	Results	OS/US
[5]	NSL-KDD	DT, KNN, RF, DNN	Avg.=85.2	No
[6]	Real world dataset	PCA	-	OS
[7]	Online Data	Adaboost, RF	-	
[8]	CIDDS-001	DNN, RF, AE	DNN US=99.30 DNN OS = 94.27 RF US=99.99 RF OS=99.99	OS and US
[9]	NSL-KDD	CANN	CANN=99.13	OS

[10]	NSL-KDD	SVM, ANN	SVM = 82.34
			ANN = 83.68
				No

[11]	NSL-KDD	RF	-	OS
[12]	KDD Cup99	K-Means+ SMO	Avg.=99.32	No
[13]	KDD Cup99	DT, NB	-	OS and US

[14]	Specialized dataset	DT, SVM	DT = 99.86
			SVM=99.62
				No

A comparison of these related works is made in Table 1 by showing the used datasets, achieved efficiencies, and usage of oversampling (OS) and undersampling (US) methodologies.The bracketed numbers are the reference numbers of the related works. It is seen that the studies mentioned in the Table 1 use old datasets such as KDD-Cup99, NSL-KDD, or their own datasets. This makes the detection of the newest attacks challenging. Especially systems developed with older datasets as KDD-Cup99 and NSL-KDD are not suitable for detecting current attacks. To implement a more effective IDS, an up-to-date dataset is needed to be used. Additionally, most of the previous IDS implementations measure the normal accuracy of the system for showing the efficiency of them. However, this value does not give the correct performance of the system, especially in imbalanced datasets. Therefore, measuring the average accuracy, which gives the same weight for all class types, should be accepted as the primary performance metric.

III. DATASETS

IDSs can be developed either in signature-based or anomaly-based forms. To define the anomaly of a system, normal and abnormal requests should be trained by using a dataset. Researchers can use either a public dataset or they can use their own datasets. In the following subsections, most favored datasets are mentioned and then compared with their content and properties.

A. KDD CUP99

KDD Cup99 was created in 1998 by DARPA to detect anomaly network traffic and was used in the 1999 KDD Cup Challenge to test IDSs [16], [17]. In its construction, nine weeks of LAN raw data which results in a TCP dump are used. This dataset is one of the most popular datasets in the fields of data mining and machine learning. There are about 5 million data in the standard dataset. Approximately 80% of the data are attack data, while the remainings 20% are benign. There are 41 properties in the dataset that can be categorized under three headings; basic features, traffic features, and content features. Data in the dataset can be classified into five main categories, which are listed as follows;

- Normal: Non-attack data.
- DoS Attacks: These attacks are typically used to prevent users from receiving services by sending multiple connection requests to a server over security in the TCP / IP protocol structure.
- Probe Attacks: These attacks are performed to find specific information on a server or any machine.
- R2L Attacks: These are attacks made with unauthorized access as a guest or another user.
- U2R Attacks: These are the attacks of a user who is allowed to enter the system but who is not an administrator; by using this type of attack, a user can act as an administrator, and perform unauthorized operations.

There are 22 different attacks within these four main categories [18], and one normal data class. KDD data include some numerical and text-based information about operations performed, and depending on the aim, they are needed to be processed.

B. NSL-KDD

The NSL-KDD dataset was created in 2009 to solve problems related to irregular data in the KDD Cup 99 dataset [19]. The reliability of the systems developed in the previous years was questioned, as there were no accurate datasets for IDSs. The NSL-KDD dataset has important advantages over the original KDD Cup99 dataset:

- Unnecessary records in training data have been eliminated; it contains important records in the KDD Cup99 dataset,
- It doesn’t have duplicate data,
- More homogeneous distribution,
- The number of records in the training and test sets is proportionally distributed,

The NSL-KDD dataset contains a feature map with 42 features, which are grouped under four categories;

- General features,
- Content features,
- Server-based traffic features,
- Time-dependent traffic features.

Attacks in the NSL-KDD dataset are divided into four different categories: DoS, Probe, U2L, and R2L. In addition to these attacks, there is a single Normal/Benign category.

C. CIC-IDS2017

CIC-IDS2017 was created in 2017, and it includes the most recent and real-world attacks of that year. It was created by analyzing network traffic using information from the timestamp, source and destination IPs, source and destination ports, protocols, and attacks [20]. It includes 86 network-related features that also contain IP addresses and attack types.

In accordance with the last dataset evaluation framework in 2016, the criteria for establishing a reliable dataset were determined. Before the creation of CIC-IDS2017 dataset was introduced, no intrusion detection dataset had met the criteria for building a reliable dataset, which was developed in 2016. The criteria are as follows:

- Completed Network Structure / Configuration,
- Completed Traffic Structure,
- It contains tagged data,
- All network traffic is recorded,
- All protocols are included in the dataset,
- Common attacks are distributed proportionally.

D. CSE-CIC-IDS2018

The profile concept was used to create the CSE-CIC-IDS2018 dataset [21]. This is the most recent dataset available in 2018/2019 by the Canadian Institute for Cybersecurity. These profiles can be used by agents or people to create events on a network and can be applied to various network protocols with different topologies. Furthermore, the dataset was enhanced by considering the standards used in the creation of CIC-IDS2017. In addition to the basic criteria, it offers the following advantages:

- The number of duplicate data is very low,
- Uncertain data is nearly absent,
- The dataset is in a CSV format, so it ready to use without processing.

This is one of the most recent datasets currently. Two profiles were classified, and five different attack methods were used in the dataset. In addition, various scenarios were created, and data were collected daily. The dataset was edited daily, and raw data were recorded. When creating data, 80 statistical properties such as time, number of packets, number of bytes, packet length, etc. are calculated separately in the forward and reverse direction, and information is given about whether an attack is added. The final dataset is published over the Internet to the researchers, with approximately 5 million data both in PCAP and CSV format. The CSV format dataset should be used if Artificial Intelligence techniques are to be used; the unprocessed PCAP data should be used if a new feature is to be extracted. The numbers of attacks and benign types are shown in Table 2 below. Also, this table shows the IDS Datasets and their features

E. IMBALANCE RATIO OF KNOWN DATASETS

Table 3 shows the number of records of the most preferred and popular datasets, which are categorized by its classes.

As can be seen, these datasets are not balanced. For accurate calculation of the system’s efficiency, this imbalanced structure is needed to be formulated. The imbalance ratio which can be calculated as in Equation 1 can be used as the

TABLE 2. CSE-CIC-IDS2018 data distribution.

Class Label	Number	Volume (%)
Benign	2,856,035	63.111
Bot	286,191	6.324
Brute Force	513	0.011
DoS	1,289,544	28.497
Infiltration	93,063	2.056
SQL injection	53	0.001
Total	4,525,399	100

TABLE 3. Data sizes of datasets.

Dataset	Class-1	Class-2	Class-3	Class-4	Class-5	Class-6
KDD Cup99	4,113,223	553,301	45,268	18,599	112	-
NSL-KDD	77,054	53,387	14,077	4,833	119	-
CIC-IDS2017	2,358,036	453,438	15,967	1,966	36	21
CSE-CIC-IDS2018	2,856,035	1,289,544	286,191	93,063	513	53

TABLE 4. Imbalance ratio of known/recent datasets.

Dataset	Imbalance Ratio
KDD Cup99	36,725
NSL-KDD	648
CIC-IDS2017	112,287
CSE-CIC-IDS2018	53,887

metric.

$$Imbalance\ Ratio = \rho = \frac{max_i\,\{C_i\}}{min_i\,\{C_i\}} \tag{1}$$

where C_i shows the data size in the class i. In other words, imbalance ratio can be defined as the fraction between the number of instances of the majority (max) class and the minority (min) class.

According to this equation the imbalance ratio of the most popular and recent datasets are listed as in Table 4. There is a vast gap between the data classes which also affects the efficiency of the system. Additionally, sophisticated hackers focus on the development of minority data types to reach their targets. Therefore, to increase the efficiency of the system, this imbalance rate should be decreased.

IV. PROPOSED SYSTEM

Many IDS development studies have been conducted over the years, and increasing detection accuracy is the most critical metric for developers. However, if the dataset is imbalanced and a specific category composes the most significant part of the dataset, then the use of accuracy as a single metric is not much acceptable. If there is a large gap between the data size within the majority and minority categories, sophisticated attackers can focus on minority attack types to increase their efficiency. Therefore, in this paper, we focus on removing the effect of asymmetry between classes in the dataset by increasing the average accuracy of the system.

As mentioned before, many current IDSs are developed over Anomaly Detection by identifying the normal data with the use of six machine learning algorithms. As such, many helpful tools have been created over the last few decades, and currently, the Python programming language, as one of the most popular development environments, has become very important for implementing new learning-based systems. The use of new libraries, such as Scikit-Learn (Sklearn) provides excellent flexibility and ease of use not only for system development but also for testing.

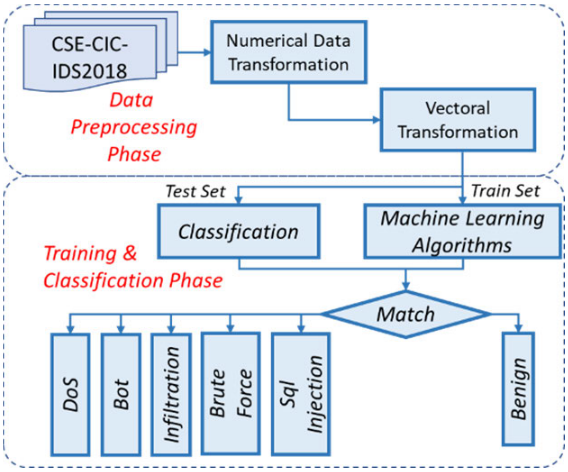


FIGURE 2. Flowchart of the IDSs with original data.

A. FLOWCHARTS OF THE PROPOSALS

Many works have used two well known datasets as KDD Cup99 and NSL-KDD which are relatively old. Therefore, in this paper, an up to date dataset CSE-CIC-IDS2018 is utilized. Along with an up to date dataset, intrusion detection was made with synthetic data production. To obtain the results of the original data in the trained system, the implemented system is executed with original data according to the flowchart depicted in Fig. 2.

However, like many other datasets, this dataset is also imbalanced. To remove the effect of the asymmetric categories, some data-driven techniques can be used. In this work, the data sampling model is used to decrease the imbalance ratio of the system. Thought this model, new data are created for minority classes, and the system is trained with them. To observe the effect of the sampled data, the flow chart of the system is modified, as depicted in Fig. 3.

B. PYTHON AND SCIKIT-LEARN

The Python programming language is easy to use and learn type general-purpose programming language, and it is currently one of the most preferred application development platforms for many application areas. Its efficient structure enables it to be quickly implemented and integrated with other systems, such as desktop and web applications, data analysis and visualization programs, network programming, database applications, and machine learning-based systems. It can be run independently of the platform and does not require a compiler. It is compatible with many operating systems such as Windows, Linux, Mac, and Symbian. It has appropriate code support for

TABLE 5. IDS datasets.

Dataset	Year	Attack Types	Attack Names	Number of Data	Advantage	Labeled	Meta-Data	Balanced	Public
CIDDS-001 [22]	2017	5	Normal, portscan, pingscan,DoS, brute force	32 M flows	Appropriate dataset for NIDS	Yes	Yes	No	Yes
AWID [23]	2016	16	authentication request, ARP flooding, injection, probe request	37 M packets	Splitted into training and test dataset	Yes	Yes	No	No
BOOTERS [24]	2013	9	9 different DDoS attack	250GB (Packets)	Real data	No	No	No	Yes
CIC-IDS2017 [21]	2017	6	Bot, brute force, DoS, infiltration, sql injection	2,829,464	Dataset that provides reliable data standards	Yes	Yes	No	Yes
CSE-CIC-IDS2018 [21]	2018	6	Bot, brute force, DoS, infiltration, sql Injection	4,525,399	Provides the specified data standards, the most recent and reliable dataset	Yes	Yes	No	Yes
DDoS2016 [25]	2016	4	DDoS (HTTP flood, SIDDOS, smurf ICMP flood, UDP flood)	2.1 M packets	Includes normal and attack traffic	Yes	No	No	Yes
IRSC [26]	2015	-	-	-	Real data, includes normal and attack traffic	Yes	No	-	Yes
KDD Cup99 [27]	1998	4	Dos, U2R, R2L, probe	4,730,503	Most used and popular dataset, split into subcategories	Yes	No	No	Yes
Kent [28]	2016	-	-	130M flows	Real data	No	No	No	Yes
NDSec-1 [29]	2016	12	Botnet, brute force, DDoS, exploits, probe, spoofing, SSL proxy, XSS/SQL injection	3.4M packet	Real data, different scenarios, includes event logs	Yes	No	No	Yes
NGIDS-DS [30]	2016	7	backdoors, DoS, exploits, generic, reconnaissance, shellcode, worms	1M packets	Includes packet data and event logs	Yes	No	No	Yes
NSL-KDD [19]	2009	4	Dos, U2R, R2L, probe	149,470	Improved version of KDD Cup99, does not contain repetitive data, Split into subcategories	Yes	No	No	Yes
PUF [31]	2018	-	DNS	300k flows	Recent dataset	Yes	No	No	No
TRAbID [32]	2018	16	DoS, port scans	460M packets	Recent dataset, split into subsets, large number of data	Yes	Yes	No	Yes
UGR'16 [33]	2016	5	botnet, DoS, port scans, SSH brute force,	16.9M flows	ISP environment attacks, splitted into training and test dataset	Yes	No	No	Yes
UNSW-NB15 [34]	2015	9	backdoors, DoS, exploits, fuzzers, generic, port scans, reconnaissance, shellcode, spam, worms	2M packets and flows	Includes packet and flow data	Yes	Yes	No	Yes

processing on one or more CPUs/GPUs with additional parallel execution libraries for increasing the performance of the system.

There is a machine learning library called Scikit-Learn, which is an open-source library, that was developed as an extension of the SciPy library in Python. It allows the implementation of various machine learning algorithms such as classification and clustering. It also provides specialized modules such as feature extraction and, model review. Scikit-Learn is very popular among researchers because it has a lot of resources and is easy to use. Therefore, in this paper, these languages and libraries are utilized to implement the proposals.

C. DATASET, PREPROCESSING AND SYNTHETIC MINORITY OVER-SAMPLING TECHNIQUE (SMOTE)

In this study, the most recent dataset available (CSE-CIC-IDS2018) is used. It is publicly accessible, and it provides CSV, PCAP and logs files. Detailed information about this dataset is provided in Section III. The operations performed during the preprocessing of the dataset are detailed below.

- Missing values, which are also referred to as Not a Number (NaN) values, have been converted to 0 to prevent value errors while working with machine learning models.
- Two columns ('Flow Bytess' and 'Flow Pktss') contain infinity values. These infinity values have been set to the maximum value in the column in which they are present + 1.
- One column ('Timestamp') has been separated into two new columns (as 'Date' and 'Time') so that there are no text values in the dataset. (Both columns are converted so that they have numeric forms such as YearMonthDay for the 'Date' column and HourMinuteSecond for the 'Time' column.)
- Two columns ('Init Fwd Win Byts' and 'Init Bwd Win Bytes') contained -1 as the value in some samples. Two new columns have been created ('Init Fwd Win Byts

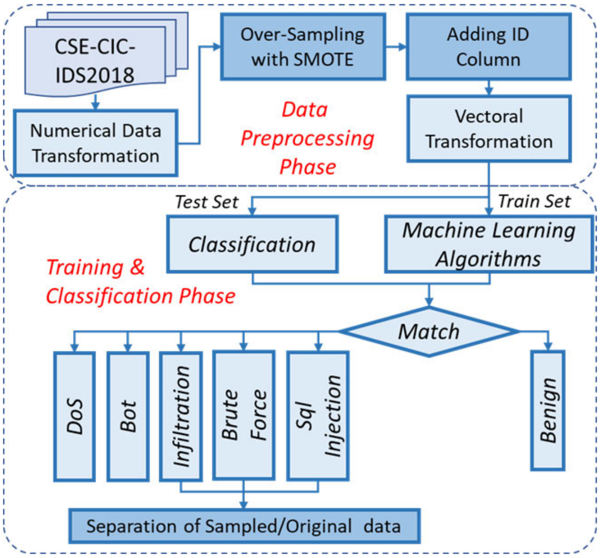


FIGURE 3. Flowchart of the IDSs with sampled data.

TABLE 6. CSE-CIC-IDS2018 labels.

Label	Value
Benign	0
Bot	1
Brute Force	2
DoS	3
Infiltration	4
SQL injection	5

Neg' and 'Init Bwd Win Byts Neg') that have 1 as the value where −1 values are present in columns with a similar name and 0 as the value where non-negative values are present.

- One column ('Label') contains the identified attack names. These attack names have been changed to numerical values as in Table 6.
- As the last step for preprocessing combined dataset is shuffled for randomness. The number of features in the dataset increased to 83 from 80 after preprocessing is completed.

Synthetic Minority Oversampling Technique (SMOTE) method was used for the generation of synthetic sample data. This method uses the K-nearest neighbor algorithm [31] to generate new samples. The literature mentions two similar methods ADASYN and RandomOverSampler. The first one, ADASYN, also generates sampled data using KNN algorithm. However, ADASYN generated data are hardly classified with the nearest neighbor approach, while SMOTE does not make any difference. Therefore, it takes a long time to produce sample data by ADASYN. In the preliminary experiments, the achieved accuracy rate by ADASYN was about 5% lower than the SMOTE generated ones. Additionally, the data generation time was also very long. So ADASYN method was not preferred for this study. The second one, RandomOverSampler, selects data randomly from the dataset and uses the same data as a sample. Similarly, the RandomOverSampler method was not preferred in our research because it samples existing data and, therefore, the results of RandomOverSampler did not reach better accuracies, especially for minority classes.

TABLE 7. CSE-CIC-IDS2018 number of sampled data.

Label	Number	Volume (%)
Benign	2,856,035	53.98
Bot	286,191	5.41
Brute Force	286,191	5.41
DoS	1,289,544	24.38
Infiltration	286,191	5.41
SQL injection	286,191	5.41
Total	5,290,343	100
Imbalance Ratio	9.98	

In the end, due to its simplicity regarding its implementation and interpretation; its efficiency in low dimensional data, and its rules for synthetic generations different from replicas, SMOTE is selected as the synthetic data generation model [6], [7], [9], [10], [13], [35]. The level of oversampling in SMOTE is directly related to the number of neighbors in the KNN algorithm, which is chosen randomly. SMOTE function creates new samples by considering the difference between the feature vector and its nearest neighbor and multiplying that difference by a random number between 0 and 1. It then adds that result to the evaluated feature vector.

D. SEPARATION OF SAMPLED DATA

Machine Learning algorithms were performed separately on a normal dataset and dataset with sampled data. Some studies showed that sampled data production should be done before the training phase [8], [36]–[39]. Therefore, the sampled data generation was carried out before all operations in the second part of the proposals. The sampled data are generation Brute Force, Infiltration, and SQL Injection attacks are initially determined to be 286,191. Data generation was performed on data containing less than 5% of the total number in the data set. And this size is chosen as the third majority class-size.

Processing sampled data is a trivial issue. Therefore, an ID column has been added to the dataset to keep track of which data is sampled. Training and tests have been performed with these sampled data using a K-fold approach to reach a realistic result by eliminating the effect of randomness. After the tests, the number of correctly predicted original sample data can also be determined easily. The ID column previously added to the dataset was used for this operation. This was done to observe the accuracy of the prediction by deleting the effect of the sampled data as test samples.

Table 7 shows the final data distribution of the dataset after running the SMOTE method by generating new samples for minority classes. Through this approach, the imbalance ratio was decreased from 53,887 to 9.98, which is an acceptable rate. The data sampling model increases the total dataset size by about 17%, which increases the training time of the system.

E. MACHINE LEARNING ALGORITHMS

As mentioned previously, for the anomaly-based intrusion detection systems, firstly, the normal behavior of the network flow should be determined. To accomplish this, the system needs to be trained using a learning algorithm. The literature offers many machine learning algorithms. To select the most suitable one for our purposes, we implemented six of them, which are detailed in the following part.

1) ADABOOST ALGORITHM

Adaptive Boosting (AdaBoost), is a community learning Boosting algorithm, that is used for classification problems [40]. "Boosting" is the process of achieving a strong result by combining weak results from the data. It dispenses the data evenly in the first step and then makes a classification. Through this classification, it finds the weakest classifier and the updates the weights. It focuses on the worst result during the updating process. After a while, it brings together several bad classifiers to form a successful classifier. Its aim is to increase its success in terms of classification. The final equation for the classification of the dataset can be shown in the following equation 2.

$$F(x) = sign(\sum_{i=1}^{N} \theta_i f_i(x)) \tag{2}$$

where f_i stands for the i^{th} weak classifier, and θ_i is the corresponding weight.

2) DECISION TREE ALGORITHM

The Decision Tree (DT) is one of the supervised learning algorithms used for the classification of numerical and class data. It has a predefined goal variable. It also has leaf nodes supported by decision-making steps to reach one of the top-down goals of the algorithm structure [41]. It takes advantage of its simple structure to process large amounts of data quickly. In some cases, more complex trees have to deal with the classification of datasets. In such cases, decision trees become more complex, and it becomes more difficult to reach any of the goals. Overfitting is another problem in decision tree algorithms. Some of the leaf nodes are pruned out of the decision tree to solve this problem. Entropy and information gain should be calculated for decision trees. Equation 3 shows how entropy is calculated.

$$E(S) = \sum_{x \epsilon X} -p(a)E(a) \tag{3}$$

where S is a dataset, X is a set of classes in S, and p is a ratio of the number of elements in class x. Equation 4 show how information gain is calculated.

$$Gain(A, S) = E(S) - \sum_{a \epsilon T} p(a)E(a) \tag{4}$$

where T is the subsets created from the dataset S.

3) RANDOM FOREST ALGORITHM

The Random Forrest (RF) is a type of supervised machine learning architecture that can be used for classification and regression problems [42]. It is effortless to use, and it creates a decision forest by using Decision Trees and performs problem-solving in this way. For this, it creates a random collection of trees. During the process, more than one Decision Tree is trained to yield the most accurate classification. Most of the time, even without the use of a hyperparameter, it can give quite good results. It is one of the most highly preferred methods because it wuickly provides speedy and accurate results even for mixed, incomplete, and noisy datasets.

4) K NEAREST NEIGHBOR ALGORITHM

K Nearest Neighbor(KNN) Algorithm is a supervised learning algorithm. Unlike other supervised learning algorithms, it does not have a training stage [43]. KNN is implemented using data from an original sample class. K data is chosen, which is the closest neighbor to the new data to be decided which sample class it should be added. The distance of the new data to be included in any of the original sample class groups is taken from the data showing the K nearest neighboring property. Euclidean, Manhattan, and Minkowski functions are used for distance calculations. The following equations 5, 6 and 7 show how these distances are calculated.

$$Euclidean = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2} \tag{5}$$

$$Manhattan = \sum_{i=1}^{N} |x_i - y_i| \tag{6}$$

$$Minkowski = (\sum_{i=1}^{N} (|x_i - y_i|)^k)^{\frac{1}{k}} \tag{7}$$

where N is the dataset size, k is a positive integer, and x_i, and y_i are the i^{th} coordinates of data. This method is highly resistant to simple and noisy training data. As such, it has the disadvantage of requiring a lot of memory space because it stores all the cases in distance calculations.

5) GRADIENT BOOSTING ALGORITHM

The Gradient Boosting Algorithm (GB) is used for regression and classification problems [44]. Similar to the Adaboosting algorithm, a combination of weak classification models generally creates a model of decision trees. The aim of increasing the gradient by updating the estimates according to the learning rate is, to reach the minimum error values.

6) LINEAR DISCRIMINANT ANALYSIS

LDA is used to reduce the number of dimensions because it makes calculation easier, takes steps to classify data in the best way, and reduces underfitting/overfitting problems [45]. Also, LDA can be used for data preprocessing before classification. It examines the distribution of classes for

TABLE 8. Machine learning algorithms hyperparameter values.

Algorithm	Hyperparameters
Adaboost	Estimator=50, Learning Rate=1.0, Algorithm="Samme R"
Decision Tree	Splitter=”Best”, Criterion=”Gini”, Minimum samples split=2, Minimum samples leaf=1
Random Forest	Estimators="Warn", Criterion="Gini", Minimum samples split=2, Minimum samples leaf=1
KNN	Class=5, Weight=”Uniform”, Distance=”Minkowski”
Gradient Boosting	Loss=”Deviance”, Learning Rate=1.0, Estimator=100, Maximum Depth=3, Validation=0.1
Linear Discriminant Analysis	Solver=”syd”

classification and finds the difference between the average values so that it creates subspaces.

Although similar to PCA, LDA maximizes the distance between classes but there is no class concept in PCA and it only tries to maximize the distance between data points. LDAs hyperparameter values of Machine Learning algorithms are shown in Table 8. These are the default values in Python's Scikit-Learn library. It is seen that hyperparameters of machine learning algorithms were set as default in the literature. Therefore, these values were left as default values in the study so that comparisons with other studies can be made.

V. EXPERIMENTAL RESULTS

In this study, the performance of machine learning algorithms in intrusion detection procedures is examined. Training and tests were conducted on the most recent dataset available (CSE-CIC-IDS2018). The parameters are selected by default in all the implemented algorithms except for KNN. In the KNN algorithm, the number of classes was determined to be six (one for non-attack type, and 5 for attack types). To decrease the variability of the performance results due to the random generation of train and test sets, the K-Fold Cross-Validation method was used in the experiments. The chosen K value was 5, in which the training and test data were divided into 80% to 20%.

Proposed systems were implemented in Keras/Tensorflow using the Python programming language, and Scikit learn libraries. To measure the performance metrics, experiments are executed on a workstation that has the properties shown in Table 9. Proposed systems were executed on the Multicore structure of the NVIDIA GeForce® GTX 1080 Ti Graphic card, whose specifications are detailed in Table 10.

To calculate the performance measure of the proposed systems; *Accuracy, Precision, Recall, F1-Score* and *Error Rate* values are used [46]. These metrics are calculated according to Equations 8- 14.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{8}$$

TABLE 9. Development environment.

Hardware	Properties
CPU	Intel(R) Core(Tm) I7-8700 Cpu @3192Mhz, 6 Cores
Op. Syst.	64 bit, Windows 10
Graphic Card	NVIDIA GeForce® GTX 1080 Ti Founders Edition 11G
L1/L2/L3Cache	384 KB/1.5 MB/12.0 MB
RAM	16.00 GB

TABLE 10. GPU specifications for development environment.

Hardware	Properties
CUDA Cores	3584
Graphics Clock	1480 MHz
Processor Clock	1582 MHz
Memory Clock	1376 MHz
Memory	11 GB
Memory Interface Width	352 bits
Memory Interface	GDDR5X
Memory Bandwidth	11 Gbps

$$Accuracy_i = \frac{TP_i + TN_i}{TP_i + FN_i + FP_i + TN_i} \tag{9}$$

$$AverageAccuracy = \frac{\sum_{i=1}^{l} \frac{TP_i+TN_i}{TP_i+FN_i+FP_i+TN_i}}{l} \tag{10}$$

$$ErrorRate = 1 - Accuracy \tag{11}$$

$$Precision = \sum_{i=1}^{l} \frac{TP_i}{TP_i + FP_i} \tag{12}$$

$$Recall = \sum_{i=1}^{l} \frac{TP_i}{TP_i + FN_i} \tag{13}$$

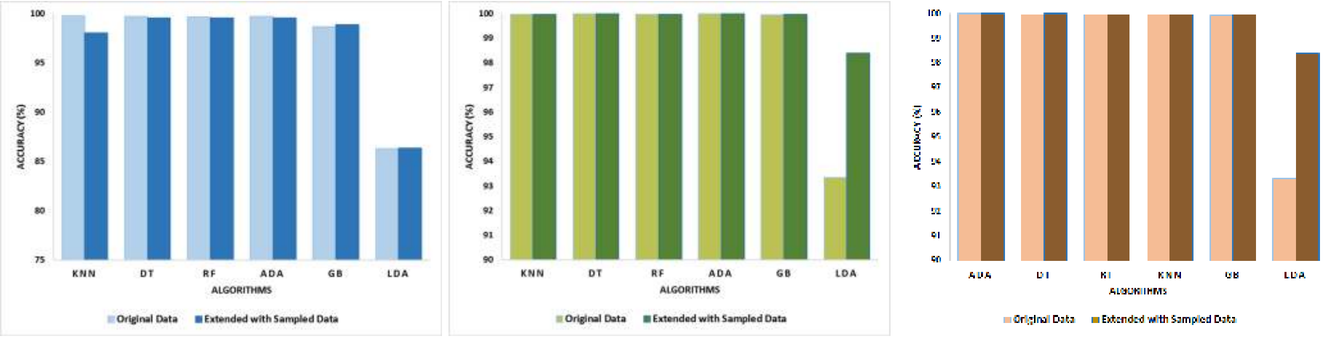
$$F1 - Score = \frac{(\beta^2 + 1)Precision * Recall}{\beta^2 Precision + Recall} \tag{14}$$

where TP_i is the i^{th} True Positive, FP_i is the i^{th} False Positive, FN_i is the i^{th} False Negative, l is the number of multiclass, and β is the balancing factor. The most common choice for β is 1,which is a harmonic mean of precision and recall.

The definition used of *accuracy* is critical because accuracy is the most vital metric used to measure the effectiveness of prediction systems. Accuracy often refers to the complete accuracy of the system, However, $Accuracy_i$ can alo refer to an individual accuracy of class i. For an imbalanced dataset, the final definition of accuracy -which is the average of the individual accuracies- is critical for researchers.

In this paper, we have implemented six different machine learning algorithms as K Nearest Neighbor, Adaboost, Random Forest, Gradient Boosting, Decision Tree, and Linear Discriminant Analysis. The performance metrics are obtained through the original dataset and extended dataset with sampled data on attack types. As the first metric, the accuracy is measured. The results of these measurements are depicted in Table 11, which also shows the execution time of the whole process.

Although accuracy is the most prominent metric used for comparing the performance of IDSs, other metrics



(a) Dos (b) Benign (c) Bot

FIGURE 4. Accuracy values for majority-class attacks.

TABLE 11. Accuracy rate and training time of the implemented algorithms.

Model	Original Dataset			Sampled Dataset		
	Acc. (%)	Error (%)	Time (sec)	Acc. (%)	Error (%)	Time (sec)
ADA	99.69	0.31	21,149	99.60	0.40	30,108
DT	99.66	0.34	163	99.57	0.43	224
RF	99.21	0.79	198	99.35	0.65	274
KNN	98.52	1.14	1,54	98.08	1.92	1,941
GB	99.11	0.89	16,739	99.29	0.71	28,759
LDA	90.80	9.20	86	91.18	8.82	135

TABLE 12. Other performance metrics of the proposed algorithms.

Model	Original Dataset			Sampled Dataset		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score
ADA	99.70	99.69	99.70	99.60	99.61	99.60
DT	99.66	99.66	99.66	99.56	99.57	99.56
RF	99.30	99.20	99.25	99.35	99.34	99.35
KNN	99.28	98.52	98.89	97.92	98.08	98.00
GB	99.51	99.11	99.29	99.30	99.29	99.30
LDA	98.90	99.11	99.00	91.96	91.18	91.57

TABLE 13. Accuracy rates according to attacks without sampled data.

Model	Accuracy (%)						
	Benign	Bot	DoS	Brute Force	Infilt.	Sql Inj.	Avg. Acc.
ADA	99.73	99.99	100	92.98	93.41	86.89	95.49
DT	99.72	99.98	100	92.78	91.66	83.01	94.53
RF	99.64	99.97	99.99	71.34	72.18	60.37	83.92
KNN	99.75	99.97	99.97	70.76	36.16	3.96	73.43
GB	98.65	99.94	99.99	25.34	98.71	33.96	76.10
LDA	86.28	93.34	99.89	52.24	95.81	33.96	76.92

TABLE 14. Accuracy rates according to attacks with sampled data.

Model	Accuracy with Sampled Data (%)						
	Benign	Bot	DoS	Brute Force	Infilt.	Sql Inj.	Avg. Acc.
ADA	99.56	99.99	99.99	99.99	96.37	100	99.32
DT	99.55	99.99	100	99.99	95.60	96.23	98.56
RF	99.53	99.97	99.99	99.99	92.63	99.99	99.19
KNN	98.07	99.97	99.98	99.85	73.91	100	95.30
GB	98.89	99.96	99.99	99.62	97.83	100	99.38
LDA	86.32	98.40	99.88	51.83	97.82	67.43	83.62

(e.g., precision, recall, and f1-score) should also be measured. Table 12 lists these metrics according to the type of machine learning algorithm used.

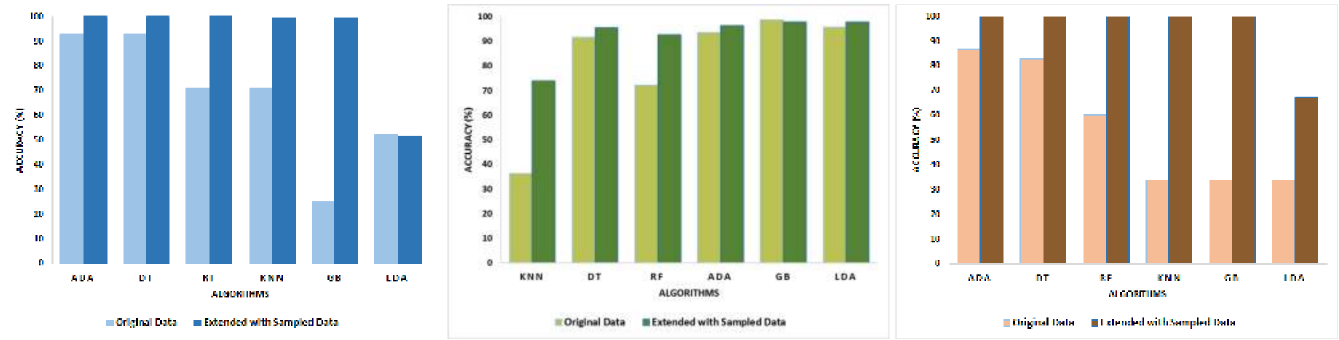
As can be seen in the table, the Adaboost algorithm is the most successful algorithm with an accuracy rate of 99.69%. The Decision Tree algorithm is the second-most efficient one with an accuracy rate of 99.66%. These are followed by the other algorithms applied to both the original dataset and the sampled dataset.

However, looking only at the complete accuracy does not yield precise comparisons. Because multi-classification IDSs are executed in this paper. The accuracy related to each attack type should be examined separately. New intruders are sophisticated. Therefore, the average accuracy of different algorithms for each attack type should be considered to determine the efficiency of the system.

Accuracies for all attack types are calculated for each machine learning algorithm, as depicted in Table 13. As can be seen from this table, three types of attacks (Brute Force, Infiltration, and SQL Injection) are associated with relatively low accuracy rates.

The low accuracy rates for these attack types stem from the data size in the dataset, as mentioned in Table 2. The total volume of them is about 3%. To increase these rates, new data are generated synthetically, and the total number of these attacks is increased by up to 16.2%, as depicted in Table 5. Then, the proposed algorithms are executed again, and the obtained accuracy rates are depicted in Table 13.

As seen in Table 13 and Table 14, the use of sampled data results in small enhancements to the first three majority data types (Benign, Bot, and Dos), the comparison of their individual accuracies are shown in Fig. 4. To compare these attacks, six machine learning algorithms and three data types are compared. For five of them, Original datasets provide the best solutions while the results are the same for six of them. For seven of these values, Sampled Data provides the best accuracy rates. However, in the minority classes, there are considerable increases, as seen in Fig. 5. In these classes, there are 72.35% accuracy increases for an average.



(a) Brute Force (b) Infiltration (c) SQL Injection

FIGURE 5. Accuracy values for minority-class attacks.

TABLE 15. Comparison table (*accuracy values are written approximately depending on the referenced paper).

Reference [15]		Normal		Sampled	
Alg.	Accuracy (%)	Alg.	Accuracy (%)	Alg.	Accuracy (%)
		ADA	99.69	*ADA*	99.60
SVM Lin	98.8*	*DT*	99.66	*DT*	99.57
SVM RBF	98.3*	*RF*	99.21	*RF*	99.35
RF	97.7*	*KNN*	98.52	*KNN*	98.58
ELM	99.5*	*GB*	99.11	*GB*	99.29
		LDA	90.80	*LDA*	91.18

Although there are small changes in these three types of data, for the resting values, there is a considerable enhancement in terms of accuracy. Fig. 5.a-c depict the accuracy levels of these three data types (e.g. Brute Force attacks, Infiltration attacks, and SQL Injection attacks). From this figure, improvements in accuracy can be seen clearly.

As discussed above, there are some comparisons of the proposed algorithms such as accuracy, time, precision, recall, f1-score. However, to measure the efficiency of a system, a comparison is made between the present study and recent work, (published in 2018) the results of which are depicted in Table 15.

The present study and the comparison study [15] have one machine-learning algorithm in common (random forest). The use of sampled data leads to, a considerable increase in the accuracy of the system, as 99.34% accuracy rate is measured. A significant difference between the papers is that in this paper instead of using the NSL-KDD dataset, which is not up to date, we use CSE-CIC-IDS2018 [21]. Additionally, a comparison with other machine learning algorithms (i.e. SVM, RBF, and ELM), shows that the trained IDSs are more efficient than these other algorithms.

The effect of sampled data size is also measured. Therefore we have tested our proposal for different dataset sizes. First, the original dataset is used for training the system.Second, the minority of dataset size is set to 93,063 (as the fourth major data class). Finally, the minority of dataset size is set to 286,191 (as the third major data class). A comparison of the average accuracies of these datasets is depicted in Fig. 6.

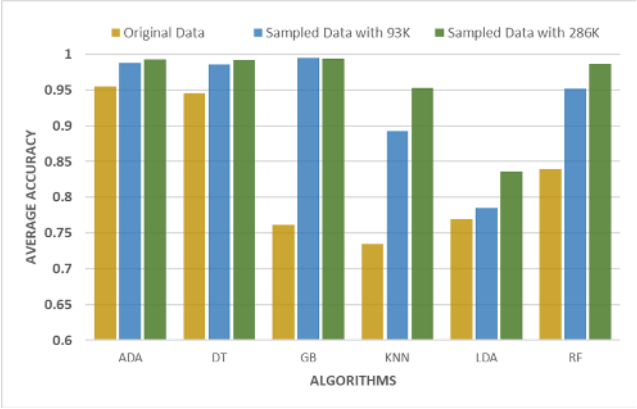


FIGURE 6. Average accuracy values according to the minimum data size.

As seen in this figure, when the size of a minority data class is increased, the average accuracy rate also increases.

VI. CONCLUSION

In recent years, due to the extended use of the Internet, computing devices can connect to a global network at any time and from anywhere. However, the anonymous form of Internet results in lots of security breaches in the network, which results in intrusions. Additionally, current attackers are more sophisticated, and with the help of automated production tools, they can generate new malwares depending on the weak detection capability of Intrusion Detection Systems(IDSs). IDSs are generally trained using pre-collected datasets. However, almost all these datasets are imbalanced with different imbalance ratios, ranging from 648 and 112,287. Imbalanced datasets result in bias towards the majority class, and in some extraordinary situations, minority classes are ignored. However, these minority classes are generally positive classes. Therefore, the imbalance ratio should be decreased to increase the efficiency of the system and to decrease its average accuracy.

In this paper, six different machine learning models (Decision Tree, Random Forest, K Nearest Neighbor, Adaboost, Gradient Boosting, and Linear Discriminant Analysis) were implemented using a recent dataset (CSE-CIC-IDS2018).

To decrease the imbalance-ratio, a data sampling model was used by increasing the data size of the minority groups. The experimental results showed that the implemented models have a very good accuracy level when compared with recent literature. The use of a sampled dataset caused the average accuracy of the models to increase between 4.01% and 30.59%.

Nowadays, due to the efficiency of big data applications, many machine learning applications are transferred to deep learning models. This paper has been a preliminary study to examine the success of deep learning algorithms in detecting small sample attacks in up to date datasets. Therefore, deep learning algorithms should be used in future work. By using a different design methodology, it is expected that the efficiency of the system will increase.

REFERENCES

[1] J. M. Johnson and T. M. Khoshgoftaar, ''Survey on deep learning with class imbalance,'' *J. Big Data*, vol. 6, no. 1, p. 27, 2019.

[2] A. Ali, S. M. Shamsuddin, and A. L. Ralescu, ''Classification with class imbalance problem: A review,'' *Int. J. Adv. Soft Comput. Appl.*, vol. 7, no. 3, pp. 176–204, 2015.

[3] F. Provost, ''Machine learning from imbalanced data sets 101,'' in *Proc. AAAI Workshop Imbalanced Data Sets*, Menlo Park, CA, USA: AAAI Press, 2000.

[4] S. Barua, M. M. Islam, X. Yao, and K. Murase, ''MWMOTE-majority weighted minority oversampling technique for imbalanced data set learning,'' *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 2, pp. 405–425, Feb. 2014.

[5] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, ''An adaptive ensemble machine learning model for intrusion detection,'' *IEEE Access*, vol. 7, pp. 82512–82521, 2019.

[6] Y.-J. Lee, Y.-R. Yeh, and Y.-C.-F. Wang, ''Anomaly detection via online oversampling principal component analysis,'' *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 7, pp. 1460–1470, Jul. 2013.

[7] Z. Yueai and C. Junjie, ''Application of unbalanced data approach to network intrusion detection,'' in *Proc. 1st Int. Workshop Database Technol. Appl.*, Apr. 2009, pp. 140–143.

[8] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. Abumallouh, ''Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic,'' *IEEE Sens. Lett.*, vol. 3, no. 1, pp. 1–4, Jan. 2019.

[9] M. R. Parsaei, S. M. Rostami, and R. Javidan, ''A hybrid data mining approach for intrusion detection on imbalanced NSL-KDD dataset,'' *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 6, pp. 20–25, 2016.

[10] K. A. Taher, B. Mohammed Yasin Jisan, and M. M. Rahman, ''Network intrusion detection using supervised machine learning technique with feature selection,'' in *Proc. Int. Conf. Robot., Electr. Signal Process. Techn. (ICREST)*, Jan. 2019, pp. 643–646.

[11] A. Tesfahun and D. L. Bhaskari, ''Intrusion detection using random forests classifier with SMOTE and feature reduction,'' in *Proc. Int. Conf. Cloud Ubiquitous Comput. Emerg. Technol.*, Nov. 2013, p. 127–132.

[12] A. Chandra, S. K. Khatri, and R. Simon, ''Filter-based attribute selection approach for intrusion detection using k-means clustering and sequential minimal optimization techniq,'' in *Proc. Amity Int. Conf. Artif. Intell. (AICAI)*, Feb. 2019, pp. 740–745.

[13] N. Qazi and K. Raza, ''Effect of feature selection, SMOTE and under sampling on class imbalance classification,'' in *Proc. UKSim 14th Int. Conf. Comput. Modeling Simulation*, Mar. 2012, pp. 145–150.

[14] A. I. Al-issa, M. Al-Akhras, M. S. Alsahli, and M. Alawairdhi, ''Using machine learning to detect DoS attacks in wireless sensor networks,'' in *Proc. IEEE Jordan Int. Joint Conf. Electr. Eng. Inf. Technol. (JEEIT)*, Apr. 2019, pp. 107–112.

[15] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim, ''Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection,'' *IEEE Access*, vol. 6, pp. 33789–33795, 2018.

[16] G. Karatas, O. Demir, and O. Koray Sahingoz, ''Deep learning in intrusion detection systems,'' in *Proc. Int. Congr. Big Data, Deep Learn. Fighting Cyber Terrorism (IBIGDELFT)*, Dec. 2018, pp. 113–116.

[17] G. Karatas and O. K. Sahingoz, ''Neural network based intrusion detection systems with different training functions,'' in *Proc. 6th Int. Symp. Digit. Forensic Secur. (ISDFS)*, Mar. 2018, pp. 1–6.

[18] D. Protić, ''Review of KDD cup '99, NSL-KDD and Kyoto 2006+ datasets,'' *Vojnotehni ki Glasnik*, vol. 66, no. 3, pp. 580–596, 2018.

[19] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, ''A detailed analysis of the KDD CUP 99 data set,'' in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.

[20] A. Gharib, I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, ''An evaluation framework for intrusion detection dataset,'' in *Proc. Int. Conf. Inf. Sci. Secur. (ICISS)*, Dec. 2016, pp. 1–6.

[21] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, ''Toward generating a new intrusion detection dataset and intrusion traffic characterization,'' in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, 2018, pp. 108–116.

[22] M. Ring, S. Wunderlich, D. Grüdl, D. Landes, and A. Hotho, ''Flow-based benchmark data sets for intrusion detection,'' in *Proc. 16th Eur. Conf. Cyber Warfare Secur. (ACPI)*, 2017, pp. 361–369.

[23] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, ''Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset,'' *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 184–208, 1st Quart., 2016.

[24] J. J. Santanna, R. van Rijswijk-Deij, R. Hofstede, A. Sperotto, M. Wierbosch, L. Z. Granville, and A. Pras, ''Booters—An analysis of DDoS-as-a-service attacks,'' in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2015, pp. 243–251.

[25] M. Alkasassbeh, G. Al-Naymat, A. Hassanat, and M. Almseidin, ''Detecting distributed denial of service attacks using data mining techniques,'' *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 1, pp. 436–445, 2016.

[26] R. Zuech, T. M. Khoshgoftaar, N. Seliya, M. M. Najafabadi, and C. Kemp, ''A new intrusion detection benchmarking system,'' in *Proc. 28th Int. Flairs Conf.*, 2015, pp. 252–255.

[27] (1999). *KDD Cup 1999 Data*. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[28] A. D. Kent, ''Comprehensive, multi-source cyber-security events data set,'' Los Alamos Nat. Lab., Los Alamos, NM, USA, Tech. Rep., 2015.

[29] F. Beer, T. Hofer, D. Karimi, and U. Bühler, ''A new attack composition for network security,'' in *10. DFN-Forum Kommunikationstechnologien*, P. Müller, B. Neumair, H. Raiser, and G. D. Rodosek, Eds. Bonn, Germany: Gesellschaft für Informatik e.V., 2017, pp. 11–20.

[30] W. Haider, J. Hu, J. Slay, B. P. Turnbull, and Y. Xie, ''Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling,'' *J. Netw. Comput. Appl.*, vol. 87, pp. 185–192, Jun. 2017.

[31] R. Sharma, R. K. Singla, and A. Guleria, ''A new labeled flow-based dns dataset for anomaly detection: PUF dataset,'' *Procedia Comput. Sci.*, vol. 132, pp. 1458–1466, 2018.

[32] E. K. Viegas, A. O. Santin, and L. S. Oliveira, ''Toward a reliable anomaly-based intrusion detection in real-world environments,'' *Comput. Netw.*, vol. 127, pp. 200–216, Nov. 2017.

[33] G. Maciá-Fernández, J. Camacho, R. Magán-Carrión, P. García-Teodoro, and R. Therón, ''UGR'16: A new dataset for the evaluation of cyclostationarity-based network IDSs,'' *Comput. Secur.*, vol. 73, pp. 411–424, Mar. 2018.

[34] N. Moustafa and J. Slay, ''UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set),'' in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.

[35] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, ''A detailed investigation and analysis of using machine learning techniques for intrusion detection,'' *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 686–728, 1st Quart., 2019.

[36] K. Matsuda and K. Murase, ''Single-layered complex-valued neural network with SMOTE for imbalanced data classification,'' in *Proc. Joint 8th Int. Conf. Soft Comput. Intell. Syst. (SCIS), 17th Int. Symp. Adv. Intell. Syst. (ISIS)*, Aug. 2016, pp. 349–354.

[37] E. Kurniawan, F. Nhita, A. Aditsania, and D. Saepudin, ''C5.0 algorithm and synthetic minority oversampling technique (SMOTE) for rainfall forecasting in Bandung regency,'' in *Proc. 7th Int. Conf. Inf. Commun. Technol. (ICoICT)*, Jul. 2019, pp. 1–5.

[38] T. E. Tallo and A. Musdholifah, ''The implementation of genetic algorithm in smote (synthetic minority oversampling technique) for handling imbalanced dataset problem,'' in *Proc. 4th Int. Conf. Sci. Technol. (ICST)*, Aug. 2018, pp. 1–4.

[39] D. Martin. (May 2019). *How to do Cross-Validation When Upsampling Data*. [Online]. Available: https://kiwidamien.github.io/how-to-do-cross-validation-when-upsampling-data.html

[40] A. J. Wyner, M. Olson, J. Bleich, and D. Mease, "Explaining the success of adaboost and random forests as interpolating classifiers," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 1558–1590, 2017.

[41] N. Frosst and G. Hinton, "Distilling a neural network into a soft decision tree," 2017, *arXiv:1711.09784*. [Online]. Available: http://arxiv.org/abs/1711.09784

[42] M. Belgiu and L. Dr gu , "Random forest in remote sensing: A review of applications and future directions," *ISPRS J. Photogram. Remote Sens.*, vol. 114, pp. 24–31, Apr. 2016.

[43] S. Zhang, X. Li, M. Zong, X. Zhu, and R. Wang, "Efficient kNN classification with different numbers of nearest neighbors," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1774–1785, May 2018.

[44] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3146–3154.

[45] L. Wu, C. Shen, and A. V. D. Hengel, "Deep linear discriminant analysis on Fisher networks: A hybrid architecture for person re-identification," *Pattern Recognit.*, vol. 65, pp. 238–250, May 2017.

[46] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manage.*, vol. 45, no. 4, pp. 427–437, Jul. 2009.

GOZDE KARATAS (Member, IEEE) was born in Istanbul, Turkey, in 1991. She received the bachelor's degree from the Mathematics and Computer Science Department, Istanbul Kültür University, in 2009, and the M.S. degree from the Computer Engineering Department, Istanbul Kültür University, in 2013. She is currently pursuing the Ph.D. degree with the Computer Engineering Department, Marmara University. She continues to work in the field of computer security. In 2015, she completed her master thesis on NoSql Database Testing in Istanbul Kültür University. During her master studies, she worked on distributed databases. She has been working as a Research Assistant with the Department of Mathematics and Computer Science, Istanbul Kültür University. Her research interests include computer networks and security, machine learning, deep learning, cryptography, python programming, statistics, and graph theory.

ONDER DEMIR received the M.S. and Ph.D. degrees in electronics and computer education from Marmara University, in 2006 and 2013, respectively. From 2003 to 2013, he worked as a Research Assistant and a Lecturer. He has been working as an Assistant Professor with the Computer Engineering Department, Marmara University. His research interests are digital image processing, biomedical image processing, cyber security, and algorithms.

OZGUR KORAY SAHINGOZ received the B.S. degree from the Computer Engineering Department, Bo aziçi University, in 1993, and the M.S. and Ph.D. degrees from the Computer Engineering Department, Istanbul Technical University, in 1998 and 2006, respectively.

He is currently working as an Associate Professor with the Computer Engineering Department, Istanbul Kültür University. He is the author of more than 100 articles. He has been working in two research projects. He graduated more than 13 M.Sc. students and supervised around six Ph.D. students. He has reviewed more than 80 national projects especially related to TUBITAK, KOSGEB-Ministry of Industry and Technology, Turkey. He is also a regular Reviewer for more than 40 Science Citation Index (/Expanded) international journals. His research interests include artificial intelligence, machine/deep learning, data science, software engineering, and UAV networking.

Dr. Sahingoz has also been very active in scientific conferences, organized and/or works as program committee members more than 100 conferences/workshops on different research areas, especially on artificial intelligence and information sciences. He has developed and taught around 20 different academic courses.