Research Article

Efficient Approach for Anomaly Detection in Internet of Things Traffic Using Deep Learning

Syed Ibrahim Imtiaz,1 Liaqat Ali Khan,1 Ahmad S. Almadhor,2 Sidra Abbas3, Shtwai Alsubai4, Michal Gregus5, and Zunera Jalil1

1Department of Cyber Security, PAF Complex, E-9, Air University, Islamabad, Pakistan
2Department of Computer Engineering and Networks, College of Computer and Information Sciences, Jouf University, Sakaka 72388, Saudi Arabia
3Department of Computer Science, COMSATS University, Islamabad, Pakistan
4College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, Al-Kharj, Saudi Arabia
5Information Systems Department, Faculty of Management Comenius University in Bratislava, Odbojárov 10, 82005 Bratislava 25, Slovakia

Correspondence should be addressed to Sidra Abbas; sidra.abbas708@gmail.com and Michal Gregus; michal.gregus@fm.uniba.sk

Received 4 August 2022; Revised 20 August 2022; Accepted 27 August 2022; Published 10 September 2022

Academic Editor: Kuruva Lakshmann

The network intrusion detection system (NIDS) is a significant research milestone in information security. NIDs can scan and analyze the network to detect an attack or anomaly, which may be a continuing intrusion or perhaps an intrusion that has just occurred. During the pandemic, cybercriminals realized that home networks lurked with vulnerabilities due to a lack of security and computational limitations. A fundamental difficulty in NIDs is providing an effective, robust, lightweight, and rapid framework to perform real-time intrusion detection. This research proposes an efficient, functional cybersecurity approach based on machine/deep learning algorithms to detect anomalies using lightweight network-based IDs. A lightweight, real-time, network-based anomaly detection system can be used to secure connected IoT devices. The UNSW-NB15 dataset is used to evaluate the proposed approach DeepNet and compare results alongside other state-of-the-art existing techniques. For the classification of network-based anomalies, the proposed model achieves 99.16% accuracy by using all features and 99.14% accuracy after feature reduction. The experimental results show that the network anomalies depend exceptionally on features selected after selection.

1. Introduction

The malicious attacks on the network are evolving at a disturbing rate [1, 2]. These attacks contain potential risks, such as distributed denial-of-service (DDoS) attacks that can exhaust services for a legitimate user by exhausting the server resources; hence, CPU, memory, disk/database bandwidth, and input-output (I/O) bandwidth become unavailable [3–5]. Similarly, worm attacks can cause infect and remotely attain unauthorized access to all the devices connected to the network. In this modern era of the connected world, security plays a crucial factor since a single successful network attack can cause losing important information [6–9]. This era is witnessing the transformation of AI for detecting anomalies, speeding the process while maintaining the effectiveness of the detection and permitting the solution to train itself autonomously. These AI-based solutions enable real-time, context-aware adaptivity, which is required by cybersecurity systems, and enable machine learning, clustering, graph mining, and entity-relationship modeling to identify potential threats [10–13].

Intrusion detection systems are a way to detect countless network attacks, especially previously unseen attack types [14, 15]. The network intrusion detection system (NIDS) is a significant research milestone in information security. NIDs can scan and analyze the network to detect an attack or anomaly, which may be a continuing intrusion or perhaps an intrusion that has just occurred. A fundamental difficulty in NIDs is providing an effective, robust, lightweight, and rapid framework to perform real-time intrusion detection [16]. This is a
crucial problem in the cyber world that needs to be resolved urgently. NIDs can detect previously unseen attacks and countless other network attacks, and it is proven as a shield to check network activity and detect intrusive events. Furthermore, NIDs emerged as the second line of defense and are utilized as a valuable framework to detect and take protective actions against sophisticated attacks [17].

With the fast-evolving technology, cybercriminals have also discovered new vulnerabilities and loopholes to violate the security premises. Therefore, security immediately became the primary and most crucial concern, whether data on a public network or a device connected to a private network. Millions of applications need security and are connected to the internet by some network. As most of the tasks are performed by integrating modern technologies, the current requirement is a reliable, fast, and appropriate security system to protect their data [18]. Network intrusion detection systems (NIDS) are commonly equipped, so signature and anomaly-based attacks can be detected [19]. In signature-based anomaly detection techniques, the network activities are compared with the database of attack signatures to identify if an attempt has been made to compromise the network, and the network administrator gets alerted with the attack details [20, 21]. In comparison, anomaly-based systems detect unknown attacks in network traffic by checking the variance in behavior from the baseline [22, 23].

During the last two years (from 2019 to 2020), COVID-19 brought a 600% plus increase in cybersecurity attacks, and in 2021, these attacks are still increasing at an abnormal rate [24, 25]. Due to the pandemic, the hand of most offices was forced to allow employees to work remotely from home. The office networks are secure with firewalls and intrusion detection prevention systems, but on the other end, cybercriminals exploit this golden opportunity to target and attack insecure home networks. These home networks are accessible gateways to gain access to corporate network data while the employees working through the home internet network are undetected. Most of the routers used by the general public are unpatched, outdated, or lack security features.

A properly efficient and robust network intrusion detection system is not a little framework [26, 27]. Particular challenges and many problems surfaced in information security for effectively and efficiently detecting anomalies in the network [28]. First, various irregular anomalies can attack the network, and these attacks comprise numerous threats and attack types. Existing network anomaly detection methods are at a loss and are unfit to survive continuously against new network malicious attacks and threats; consequently, they cannot achieve high accuracy in anomaly detection [29]. These methods are inefficient for reducing false alarms. Second, the typical machine learning algorithms utilized in intrusion detection systems show many technical errors; overfitting, imbalanced class distribution of network traffic, high bias due to irrelevant, or redundant features [30]. Another potential difficulty lies in labeling the traffic dataset for efficient and effective detection in intrusion detection systems [31]. Some research centers add updates to the NIDS datasets since substantial efforts are required to produce such labeled datasets. The last problem comprises the inability of frameworks for real-time anomaly detection on the network end. These difficulties cause NIDs to be ineffective at detecting real-world threats in large-scale environments. Furthermore, most intrusion detection systems cannot efficiently learn feature representations to build a more effective predictive model.

In recent years, numerous researches [32–39] show their results for network-based anomaly detection, but none have pushed their research to be a practical and real-time smooth framework. The main goal of this research is to derive a methodology for light real-time network-based anomaly detection. NIDS can automate and improve anomaly detection using machine learning and deep learning classifiers. Specific objectives are as follows:

(i) Propose an efficient, systematic, and functional approach based on machine/deep learning algorithms to detect anomalies using lightweight network-based IDS

(ii) Evaluate the proposed solution for judging effectiveness and efficiency with conventional machine learning models and comparing existing state-of-the-art techniques

(iii) Propose the concept of unnecessary drop-in network features based on a regressor algorithm that allows the IDs for real-time anomaly detection in-home networks while maintaining effectiveness and efficiency

(iv) For the classification of network-based anomalies, the proposed model achieves 99.16% accuracy by using all features and 99.14% accuracy after feature reduction. The experimental results show that the network anomalies depend exceptionally on features selected after selection.

The rest of our research paper is organized as follows: Section 1 provides the literature review and limitations of existing techniques. Section 2 presents information about the dataset. Section 3 explains the proposed methodology. Section 4 elucidates our proposed anomaly detection method for the proposed solution. Section 5 presents our evaluation and results. Section 6 demonstrates a detailed comparative analysis and discusses the observations made in this research. Finally, in the end, there is Section 7 that comprises conclusion and future work.

2. Literature Review

Several researchers have been working on anomaly detection in the past for different types of networks. Due to the harmful effects of anomalies and resulting attacks, several efforts have been made to curb this problem by developing a network-based intrusion detection system that can scan the network activities that could breach confidentiality, integrity, and availability and compromise network resources [40, 41]. Over the years, researchers have been conducting anomaly detection with KDD’99 and NSL-KDD datasets, and these were efficient and served the purpose well for many years. However, with emerging network-based attacks and the
new network usage patterns after COVID-19, these datasets are no more sufficient to detect these attacks.

Moustafa and Slay in [42, 43] also criticized that other datasets such as KDD’99 or NSL-KDD are limited, and these datasets are not good enough for emerging attacks in NIDS and proposed a new dataset, UNSW-NB-15. This research utilizes various KDD’99 dataset features and some more features. On the UNSW-NB15 dataset, numerous researchers have used machine learning techniques to evaluate the efficiency of the dataset. In 2020, Sarhan et al. [44] experimented on the UNSW-NB15 dataset and achieved the highest accuracy of 99.25% with binary classification without reducing any unnecessary features. The author also uses multilabel classification on this dataset and achieves the weighted accuracy of 98.19% with an $f$-score of 98%. However, the whole dataset is huge and proves to be challenging to be used in a real-time network-based intrusion detection system.

Authors in [34] explained that if the features of datasets are reduced through an algorithmic procedure (gain ratio utilized in research by author), then the dataset can be used real-time light IDs. The author evaluated the efficiency of the dataset by applying the artificial neural network (multilayer perceptron) algorithm for anomaly classification and achieved an accuracy of 76.96%. Authors in [32] used machine learning algorithms to detect cloud computing anomalies and achieved 95% accuracy with the decision tree algorithm. Other conventional techniques, correlated-adjusted decision forest (CADF), online averaged one-dependence estimators (AODE), trained artificial neural network (TNN), and Naïve Bayes (NB), achieve the accuracy of 88.2%, 83.47%, 90%, and 69.6%, respectively. However, the network-based anomaly detection accuracy rate is low and could be improved by adequately utilizing modern machine learning techniques and skills.

Authors in [45] also evaluated machine learning algorithms for anomaly detection by using gated recurrent units, random forest, Gaussian Naïve Bayes, logistic regression, adaptive boosting, $K$-nearest neighbours, decision tree, long short-term memory, convolution neural network, deep neural network, and simple recurrent neural network algorithms. In binary classification, the experiment achieved 88.5% accuracy with the decision tree algorithm on UNSW-NB15 dataset. On the other hand, with the CICIDS2017 dataset, random forest achieved 99.9% accuracy. Authors in [35] produced a novel two-stage deep learning (TSDL) model based on a stacked autoencoder with a soft-max classifier for effective NIDs. This model achieves an accuracy of 89.134% with the USW-NB15 dataset and 99.966% accuracy with the KDD99 dataset.

Authors in [38] used machine learning algorithms with hybrid optimization and proposed the method DO_IDS. This method comprises two steps, i.e., data sampling and feature selection. Data sampling uses an isolation forest (iForest) to eliminate outliers, a genetic algorithm (GA) to optimize the sampling ratio, and random forest as evaluation criteria to obtain optimal training datasets. Then, feature selection uses GA and RF to obtain the optimal feature subset. This method DO_IDS achieved an accuracy of 92.8%. In the comparative study of machine learning classifiers on NIDS from the research [36] achieved the highest accuracy of 85.34% by using the sequential minimal optimization (SMO) algorithm. Authors in [46] used the application of cyber-physical systems to receive and compute signals of human biological rhythms at the remote server. Though the research aims to receive and compute medical data from a remote server due to the high-speed internet and faster technology, it is possible to check for anomalies and validate data in real-time by a lightweight ID to secure the connected IoT devices.

Many studies unveiled an overview of the evolution of anomaly detection techniques [38, 40–43]. Some studies used machine learning and deep learning approaches to investigate an efficient method for detecting anomalies [32, 35], but faced limitations such as low anomaly detection rate. These limitations arise due to evolution, and the surface of anticipated cyberattacks and new attacks are showing new colors at an alarming rate. The focus of this paper is to address all the previously discussed limitations by providing a systematic and efficient approach for the detection of anomalies through NIDs by using a supervised learning paradigm with a minimum number of features.

## 3. Dataset Information

Many NIDs are evaluated by using a publicly available dataset. In this work, the UNSW-NB15 dataset (UNSW-NB15 Dataset: http://www.unsw.adfa.edu.au/unsw-cybersecurity/N>IDDF-UNSW-NB15-Datasets/) is used to evaluate the effectiveness of the proposed solution. The raw network packets of this dataset are created by the IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Center for Cyber Security (ACCS) to generate a hybrid of actual normal activities and synthetic recent attack behaviors. Furthermore, this dataset is updated from time to time and includes information on new modern anomalies. This dataset was released at the end of 2015 and initially had 45 features, but due to new malicious network anomaly attacks, the researchers updated this dataset to 49 features. The UNSW research group provides sample training and testing files for this dataset to evaluate machine learning techniques. Furthermore, as shown in Table 1, the complete dataset division comprises four files provided by the UNSW research center. In this research, we use the complete provided dataset to achieve more accurate evaluations.

### Table 1: UNSW-NB15 detail.

| File name       | Size   | Records | Features |
|-----------------|--------|---------|----------|
| UNSWNB15_1      | 161.2 MB | 700000  | 49        |
| UNSWNB15_2      | 157.6 MB | 700000  | 49        |
| UNSWNB15_3      | 147.4 MB | 700000  | 49        |
| UNSWNB15_4      | 91.3 MB   | 440044  | 49        |

## 4. Proposed Methodology

The proposed approach comprises data analysis, feature extraction, feature reduction, splitting training, testing data in a ratio of 8:2, and then calculating anomalies from regular traffic through machine learning and deep learning.
techniques. Figure 1 summarizes our proposed approach, which consists of the steps to achieve anomaly classification by reducing the features.

4.1. Preprocessing. Preprocessing is the initial stepping stone in machine learning so that the respective classifier can achieve the best performance without errors. The first step in the preprocessing technique is removing NaN, duplicate instances, and normalization/scaling. In this step, the rescaling of numeric attribute data is performed to a fixed range (e.g., 0 to 1) so the model does not rely on the magnitude of values. The selected dataset has low variance and ambiguities; therefore, we choose MinMax scaling for feature normalization. MinMax scaling normalizes data using the formula mentioned in 1.

\[ X_{\text{norm}} = \frac{X_i - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \]  

where \( X_i \) is the real numeric value of the feature subtracted from a minimum value of that feature and divided by the subtracted result of the maximum and minimum of the feature.

4.2. Feature Extraction. By performing feature engineering and data preparation, all the high correlation features, i.e., correlation is greater than 0.95, are dropped from the data. The complete number of features is 49 in the UNSW-NB15 dataset, and after dropping the high correlation features, the number of features drops to 41. The features dropped are "sloss," "dloss," "dpkts," "dwin," "ctime," "ct_srv_dst," "ct_src_dport_ltm," and "ct_dst_src_ltm." After this step, the features’ "sbytes" and "dbytes" are added to form a new feature, "network_bytes," that takes place in the features. This research is driven only by binary classification instead of attack category classification. So, the features that are not useful for binary classification are dropped, i.e., "srcip," "sport," "dstip," "dsport," and "attack_cat."

4.3. Feature Reduction. A supervised learning algorithm that performs effectively on numeric attribute data is random forest regressor algorithm. This algorithm utilizes the ensemble learning method for regression purposes. This regressor technique deploys many decision trees at training time and outputs the class, presenting the individual trees’ mean prediction (regression). The main advantage of this regressor model is that it performs efficiently and effectively on essential data, and it can handle thousands of input variables without variable deletion. Table 2 shows the results after dimension reduction and provides the features with their respective explained variance score. This score helps to understand the importance of features concerning the desired result. The higher the feature score, the more it affects achieving the desired decision.

4.4. Machine Learning Models. In this paper, the following machine learning and deep learning algorithms are utilized to evaluate and compare the effectiveness of our proposed approach:

- Stochastic gradient descent (SGD): Estimator implements regularized linear models with SGD learning. The loss gradient is estimated for each sample at a time, and the model is updated along the way with a decreasing strength schedule. SGD allows minibatch (online/out-of-core) learning by using partial_fit method (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model)
Table 2: Feature selection technique: random forest regressor results.

| No. | Name              | Score   |
|-----|-------------------|---------|
| 0   | Proto             | 0.00096 |
| 1   | State             | 0.00139 |
| 2   | Sttl              | 0.00280 |
| 3   | Dtl               | 0.00000 |
| 4   | Service           | 0.00189 |
| 5   | Swin              | 0.00082 |
| 6   | Trans_depth       | 0.00004 |
| 7   | Res_bdy_len       | 0.00049 |
| 8   | Stime             | 0.01324 |
| 9   | Sintpkt           | 0.00143 |
| 10  | Dintpkt           | 0.00142 |
| 11  | Tcrprtt           | 0.00191 |
| 12  | Synack            | 0.00261 |
| 13  | Ackdat            | 0.00178 |
| 14  | Is_sm_ips_ports   | 0.00000 |
| 15  | Ct_state_ttl      | 0.86503 |
| 16  | Ct_flw_http_mthd  | 0.00014 |
| 17  | Is_fip_login      | 0.00001 |
| 18  | Ct_fip_cmd        | 0.00001 |
| 19  | Ct_svc_src        | 0.01178 |
| 20  | Ct_dst_ltm        | 0.00107 |
| 21  | Ct_src_ltm        | 0.00125 |
| 22  | Ct_dst_sport_ltm  | 0.02656 |
| 23  | Dur_log1p         | 0.00191 |
| 24  | Bybytes_log1p     | 0.00518 |
| 25  | Dbytes_log1p      | 0.00586 |
| 26  | Sload_log1p       | 0.00249 |
| 27  | Dload_log1p       | 0.00144 |
| 28  | Spkts_log1p       | 0.00026 |
| 29  | Stcpb_log1p       | 0.00191 |
| 30  | Dtcpb_log1p       | 0.00195 |
| 31  | Smeansz_log1p     | 0.01195 |
| 32  | Dmeansz_log1p     | 0.00313 |
| 33  | Sjit_log1p        | 0.00122 |
| 34  | Djit_log1p        | 0.00136 |
| 35  | Network_bytes_log1p | 0.02467 |

SGDClassifier.html). For the SGD algorithm, the default parameters are used in experimentation that comprises of loss property set to “hinge” and penalty set to 12.

Decision tree (DT) is a supervised learning algorithm that continuously splits data depending upon given parameters. There are two entities of the DT, i.e., leaves and nodes. The leaves represent the outcome or the decision, and the points where data is split are nodes. The parameters used in this classifier comprise of min_samples_split value set to 2, and min_samples_leaf value set to 1.

Random forest (RF) is a supervised learning algorithm for classification and regression. This method uses a meta estimator to create nodes that implement decisions on randomly selected data samples and receive an outcome prediction from each tree. The best solutions are selected based on the outcome prediction (https://www.datacamp.com/community/tutorials/random-forests-classifier-python). To improve the predictive accuracy, random forest uses averaging and controls overfitting (https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html). The parameters for random forest algorithms that are altered during experiments are “max_depth” and “random_state” with values 8 and 0.

Extreme gradient boosting (XGB) is an extension of gradient boosted decision trees designed for both performance and speed. This algorithm is mainly used due to the efficiency of memory resources and computing time. The key algorithm implementation features comprise block structure to help parallel tree construction and sparse aware with automatic handling of missing data values (https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/). The parameter tuning setup for the XGB classifier comprises of scale_pos_weight property set to 1, the learning_rate set to 0.02, colsample_bytree property set to 0.3, the subsample property set to 0.4, the reg_alpha to 0.3, the max_depth to 2, gamma value set to 10 with 1000 n_estimators, and with objective property fixed on “binary:logistic.”

CNN: DeepNet technique comprises CNN, a deep learning algorithm with dense layers with different weights. The convolutional neural network takes an input, assigns importance by biases and learnable weights to respective aspects/objects in the input, and differentiates one from the other (https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53).

5. DeepNet

The approach DeepNet uses the CNN model is demonstrated in algorithm 1. Let D represents the dataset which contains instance $I = \{i_1, i_2, \ldots, i_n\}$, and LE represents the label encoding transformer function which changes the labels into 1-dimensional vectors. A. The mean, $\mu$, is subtracted from data for normalization and then performs normalization on the variance $\sigma$. Then, data is converted to get a 2D matrix. The library NumPy is used for this operation. Then, Gaussian variable is used to initialize the weights. $L$ denotes the total number of layers, $n$ denotes the total number of features, and $W$ denotes the matrix’s weight. $x \ast y$ denotes the dimension of the generated weight matrix. $D_2$ denotes the 2D matrix that contains the dataset of the training file, which is further processed into a 3D matrix, $D_3$. This procedure is supported by the reshape function to get the input ready for processing into the CNN model. There is one input layer, four hidden layers, and one output or flatten layer. The four hidden layer states use feature extraction by applying $32 \ast 3, 32 \ast 3, 64 \ast 3$, and $16 \ast 3$ filters and use “relu” for activation purposes. The feature map $F$ is generated by the CNN model and converted into 1-dimensional vectors $V$ after applying a max-pooling layer. This all
attained information is forwarded to the flatten layer, which converts into a 1-dimensional array. Flatten layer 1-dimensional array denoted by FCNN. This 1-dimensional data pass to the dense layer as input to predict the target labels. Last train for 30 epochs. The CNN model learns weights at every epoch to improve its accuracy by updating weights. Actual loss, validation loss, actual accuracy, and validation accuracy are measured after every epoch.

### 6. Evaluation and Results

Table 3 elaborates the features remaining after the 1st, 2nd, 3rd, and 4th instance of feature selection. In the first instance of experiments, all the features remaining after preprocessing and normalization are selected for experiment evaluation. In the second, third, and fourth instances, the features are selected based on their variance score concerning the random forest regressor algorithm. The manual threshold for selecting several features for the second, third, and fourth instance is 24, 13, and 6. After including the feature “label,” the number of features for the first, second, third, and fourth instances are 37, 25, 14, and 7, respectively. After performing feature selection techniques, it could be seen that the maximum number of features threshold to be dropped depending upon low variance is 29, and the value set of this threshold is features that have a variance more significant than 0.00999.

To compare the difference in results, machine learning analysis, and time series analysis, the factors included for this evaluation measure and comparison are accuracy, precision, recall, and f1-score.

| # | Feature name                  | Score | Instance |
|---|------------------------------|-------|----------|
| 1 | Proto                        | 0.00096 | ✔       |
| 2 | State                        | 0.00139 | ✔       |
| 3 | Stil                         | 0.00280 | ✔ ✔ ✔ ✔ |
| 4 | Dtcp                        | 0.00000 | ✔       |
| 5 | Service                      | 0.00189 | ✔       |
| 6 | Trans_depth                  | 0.00004 | ✔       |
| 7 | Res_bdy_len                  | 0.00049 | ✔       |
| 8 | Stime                        | 0.01324 | ✔ ✔ ✔ ✔ |
| 9 | Sintpkt                      | 0.00143 | ✔       |
| 10 | Dtcp                        | 0.00142 | ✔       |
| 11 | Tcpptrt                      | 0.00191 | ✔       |
| 12 | Synack                       | 0.00261 | ✔       |
| 13 | Ackdat                       | 0.00178 | ✔ ✔ ✔ ✔ |
| 14 | Is_sm_ips_ports              | 0.00000 | ✔       |
| 15 | Ct_state_ttl                 | 0.86503 | ✔     ✔   ✔ ✔ |
| 16 | Ct_flw_http_mthd             | 0.00014 | ✔       |
| 17 | Is_ftp_login                 | 0.00001 | ✔       |
| 18 | Ct_ftp_cmd                   | 0.00001 | ✔       |
| 19 | Ct_srv_src                   | 0.01181 | ✔ ✔ ✔ ✔ |
| 20 | Ct_dst_ltm                   | 0.00107 | ✔       |
| 21 | Ct_src_ltm                   | 0.00125 | ✔       |
| 22 | Ct_dst_sport_ltm             | 0.02656 | ✔ ✔ ✔ ✔ |
| 23 | Dur_log1p                    | 0.00191 | ✔ ✔ ✔ ✔ |
| 24 | Sbytes_log1p                 | 0.00518 | ✔ ✔ ✔ ✔ |
| 25 | Dbytes_log1p                 | 0.00586 | ✔ ✔ ✔ ✔ |
| 26 | Sload_log1p                  | 0.00249 | ✔ ✔ ✔ ✔ |
| 27 | Dload_log1p                  | 0.00144 | ✔       |
| 28 | Spkts_log1p                  | 0.00026 | ✔       |
| 29 | Stcp_log1p                   | 0.00191 | ✔       |
| 30 | Dtcpb_log1p                  | 0.00195 | ✔   ✔   ✔ ✔ |
| 31 | Smeansz_log1p                | 0.01195 | ✔ ✔ ✔ ✔ |
| 32 | Dmeansz_log1p                | 0.00313 | ✔ ✔ ✔ ✔ |
| 33 | Sjit_log1p                   | 0.00122 | ✔       |
| 34 | Djit_log1p                   | 0.00136 | ✔       |
| 35 | Network_bytes_log1p          | 0.02467 | ✔ ✔ ✔ ℹ |

Table 4: Results of selective machine learning and deep learning algorithms on 37 features.

| Name | Accuracy | Precision | Recall | F1-score |
|------|----------|-----------|--------|----------|
| SGD  | 0.9873   | 0.9914    | 0.9928 | 0.9902   |
| DT   | 0.9904   | 0.9903    | 0.9948 | 0.9901   |
| RF   | 0.9911   | 0.9941    | 0.9906 | 0.9903   |
| XGB  | 0.9912   | 0.9931    | 0.9901 | 0.9901   |
| DeepNet | 0.9916 | 0.9958    | 0.9936 | 0.9700   |
6.1. 1st Instance of Experiments. In Table 4, the experimental results are represented for all 37 features from Table 3 achieving the highest accuracy by DeepNet approach of 99.19%. The execution time taken for this model is 37 minutes and 58 seconds.

The SGD model is 98.73% suitable for testing concerning the UNSW-NB15 dataset. Tables 5 and 6 represent the instances of nonattack that are being confused by instances of attacks or anomalies in training and testing cases. In the training confusion matrix, 8984 of the normal instances are confused as attack instances, and 13030 of the attack instances are confused as nonattack instances. Similarly, only 3252 of the normal instances are confused as attack instances for the test confusion matrix, and 6380 of the attack instances are confused as nonattack instances. The DT model is 99.04% suitable for testing concerning the UNSW-NB15 dataset. In the training confusion matrix, only 58 of the normal instances are confused as attack instances, and 5 of the attack instances are confused as non-attack instances. Similarly, in the test confusion matrix, 2861 of the normal instances are confused as attack instances, and only 4417 are confused as nonattack instances. The RF model is 99.11% proper testing concerning the UNSW-NB15 dataset. In the training confusion matrix, 4106 of the normal instances are confused as attack instances, and 10911 of the attack instances are confused as nonattack instances. Similarly, only 1694 of the normal instances are confused as attack instances in the test confusion matrix, and only 5062 of the attack instances are confused as non-attack instances. The XGB model is 99.12% proper testing concerning the UNSW-NB15 dataset. In the training confusion matrix, 6899 of the normal instances are confused as attack instances, and 7227 of the attack instances are confused as nonattack instances. Similarly, only 3473 of the normal instances are confused as attack instances in the test confusion matrix, and only 3199 of the attack instances are confused as nonattack instances.

Figure 2(a) depicts that DeepNet achieves the highest test accuracy of 0.9919% at the 29th epoch. The convergence for accuracy training starts from 0.988% and ends at 0.992%. After the accuracy reaches 99.2%, the training accuracy convergence becomes constant. Similarly, Figure 4(b) depicts this approach achieving a minimum loss of 0.016% at the 30th epoch. The loss for training starts at 0.026 and ends at 0.016%. After the loss reaches 0.016%, the training loss convergence becomes constant. Figure 3 depicts that only 3844 of the normal instances are confused as attack instances, and only 2513 of the attack instances are confused as nonattack instances.

6.2. 2nd Instance of Experiments. In Table 7, the experimental results are represented for 25 features with the highest variance from Table 3 achieving the highest accuracy by DeepNet of 99.19%. The execution time taken for this model is 35 minutes and 44 seconds.

The SGD model is 97.89% suitable for testing concerning the UNSW-NB15 dataset. Tables 5 and 6 represent the number of instances of nonattack that are being confused as instances of attacks or anomalies in training cases testing. In the training confusion matrix, 17782 of the normal instances are confused as attack instances, and 19255 are confused as nonattack instances. Similarly, only 7386 of the normal instances are confused as attack instances in the test confusion matrix, and only 8374 of the attack instances are confused as nonattack instances. The DT model is 99.03% suitable for testing concerning the UNSW-NB15 dataset. For the training confusion matrix, 62 normal instances are confused as attack instances, and 9 are confused as nonattack instances. Similarly, only 3142 of the normal instances are confused as attack instances, and only 4323 of the attack instances are confused as nonattack instances. The RF model is proved 99.08% suitable for testing concerning the UNSW-NB15 dataset. 3756 of the normal instances are confused as attack instances for the training confusion matrix, and 11519 of the attack instances are confused as nonattack instances. Similarly, only 1953 of the normal instances are confused as attack instances in the test confusion matrix, and only 5012 of the attack instances are confused as nonattack instances. The XGB model is proved 99.11% suitable for testing concerning the UNSW-NB15 dataset. For the training confusion matrix, 6858 of the normal instances are confused as attack instances, and 7390 as nonattack instances. Similarly, only 4063 of the normal instances are confused as attack instances in the test confusion matrix, and only 2652 of the attack instances are confused as nonattack instances.

Figure 4(a) depicts that the approach DeepNet achieves the highest accuracy of 0.9918% at the 27th epoch. The convergence for accuracy training starts from 0.988% and ends at 0.992%. After the accuracy reaches 99.2%, the training accuracy convergence becomes constant. Similarly, Figure 4(b) depicts this approach achieving a minimum loss of 0.016% at the 28th epoch. The loss for training starts at 0.030 and ends at 0.016%. After the loss reaches 0.016%, the training loss

| #     | 1st instance | 2nd instance | 3rd instance | 4th instance |
|-------|--------------|--------------|--------------|--------------|
| SGD   | 13030        | 19255        | 22414        | 14529        |
| DT    | 5            | 9            | 9            | 9            |
| RF    | 10911        | 11519        | 10838        | 12134        |
| XGB   | 7227         | 7390         | 8993         | 14353        |
| DeepNet | 5298        | 8098         | 5886         | 8969         |

Table 5: Comparison of experimental results of training confusion matrix concerning confused attacks and nonattacks.
convergence becomes constant. Figure 5 depicts that only 2513 of the normal instances are confused as attack instances, and only 3722 are confused as nonattack instances.

6.3. 3rd Instance of Experiments. Table 8 presents the experimental results for 14 features with the highest variance from Table 3 achieving the highest accuracy by DeepNet approach of 99.18%. The execution time taken for this model is 33 minutes and 33 seconds.

The SGD model is proved 97.68% suitable for testing concerning the UNSW-NB15 dataset. Tables 5 and 6 represent the number of instances of nonattack that are being confused as instances of attacks or anomalies in training cases testing. It represents the number of instances of

Table 6: Comparison of experimental results of testing confusion matrix concerning confused attacks and nonattacks.

|       | 1st instance | 2nd instance | 3rd instance | 4th instance |
|-------|--------------|--------------|--------------|--------------|
|       | Attacks confused | Nonattacks confused | Attacks confused | Nonattacks confused | Attacks confused | Nonattacks confused | Attacks confused | Nonattacks confused |
| SGD   | 6380         | 3252         | 8374         | 7386         | 9627         | 7818         | 6332         | 14077         |
| DT    | 4417         | 2861         | 4323         | 3142         | 4364         | 3055         | 4230         | 3183          |
| RF    | 5062         | 1694         | 5012         | 1953         | 4786         | 2009         | 5148         | 2447          |
| XGB   | 3199         | 3473         | 2652         | 4063         | 3156         | 3945         | 4694         | 4878          |
| DeepNet | 2513       | 3844         | 3722         | 2513         | 2664         | 3824         | 3884         | 2615          |

Figure 2: Model accuracy and loss of binary classification with DeepNet using 37 features.
nonattack that are being confused as instances of attacks or anomalies. In the training confusion matrix, 18694 of the normal instances are confused as attack instances, and 22414 are confused as nonattack instances. Similarly, in the test confusion matrix, only 7818 of the normal instances are confused as attack instances, and only 9627 are confused as nonattack instances. The DT model is proved 99.00% suitable for testing concerning the UNSW-NB15 dataset.

In the training confusion matrix, 63 normal instances are confused as attack instances, and 9 are confused as nonattack instances. Similarly, only 3055 of the normal instances are confused as attack instances in the test confusion matrix, and only 4364 of the attack instances are confused as nonattack instances. The RF model is proved 99.1% suitable for testing concerning the UNSW-NB15 dataset. In the training confusion matrix, 4139 of the normal instances are confused as attack instances, and 10838 of the attack instances are confused as nonattack instances. Similarly, only 2009 of the normal instances are confused as attack instances in the test confusion matrix, and only 3156 of the attack instances are confused as nonattack instances.

Figure 6(a) depicts that the approach DeepNet achieves the highest accuracy of 0.9914% at the 30th epoch. The convergence for accuracy training starts from 0.987% and ends at 0.991%. After the accuracy reaches 0.991%, the training accuracy convergence becomes constant. Similarly, Figure 6(b) depicts this approach achieving a minimum loss of 0.016% at the 29th epoch. Training loss starts at 0.028 and ends at 0.018%. After the loss reaches 0.018%, the training loss convergence becomes constant. Figure 7 depicts that only 3824 of the normal instances are confused as attack instances, and only 2664 of the attack instances are confused as nonattack instances.

6.4. 4th Instance of Experiments. Table 9 presents the experimental results for 7 features with the highest variance from Table 3 achieving the highest accuracy by DeepNet of 99.14%. The execution time taken for this model is 29 minutes and 21 seconds.

The SGD model is proved 97.35% suitable for testing concerning the UNSW-NB15 dataset. Tables 5 and 6 represent the number of instances of nonattack that are being confused as instances of attacks or anomalies in training cases testing. It represents the number of instances of nonattack that are being confused as instances of attacks or anomalies. In the training confusion matrix, 33295 of the normal instances are confused as attack instances, and 14529 are confused as nonattack instances. The DT model is proved 99.02% suitable for testing concerning the UNSW-NB15 dataset. In the training confusion matrix, 4139 of the normal instances are confused as attack instances, and 10838 of the attack instances are confused as nonattack instances. Similarly, only 2009 of the normal instances are confused as attack instances in the test confusion matrix, and only 3156 of the attack instances are confused as nonattack instances. The RF model is proved 99.00% suitable for
testing concerning the UNSW-NB15 dataset. In the training confusion matrix, 4538 of the normal instances are confused as attack instances, and 12134 of the attack instances are confused as nonattack instances. Similarly, only 2447 of the normal instances are confused as attack instances in the test confusion matrix, and only 5148 of the attack instances are confused as nonattack instances. The XGB

Table 8: Results of selective machine learning and deep learning algorithms on 14 features.

| Name   | Accuracy | Precision | Recall | F1-score |
|--------|----------|-----------|--------|----------|
| SGD    | 0.9768   | 0.9801    | 0.9831 | 0.9811   |
| DT     | 0.9900   | 0.9901    | 0.9934 | 0.9944   |
| RF     | 0.9910   | 0.9902    | 0.9909 | 0.9900   |
| XGB    | 0.9906   | 0.9901    | 0.9905 | 0.9914   |
| DeepNet| 0.9914   | 0.9932    | 0.9941 | 0.9904   |

Figure 4: Model accuracy and loss of binary classification with DeepNet using 25 features.

Figure 5: Testing evaluation confusion matrix of 25 features.
model is proved 98.74% suitable for testing concerning the UNSW-NB15 dataset. In the training confusion matrix, 5712 of the normal instances are confused as attack instances, and 14353 are confused as nonattack instances.

Similarly, in the test confusion matrix, only 4878 of the normal instances are confused as attack instances, and only 4694 are confused as nonattack instances.

Table 9: Results of selective machine learning and deep learning algorithms on 7 features.

| Name  | Accuracy | Precision | Recall | F1-score |
|-------|----------|-----------|--------|----------|
| SGD   | 0.9735   | 0.9756    | 0.9762 | 0.9763   |
| DT    | 0.9902   | 0.9900    | 0.9943 | 0.9955   |
| RF    | 0.9900   | 0.9900    | 0.9907 | 0.9941   |
| XGB   | 0.9874   | 0.9901    | 0.9955 | 0.9921   |
| DeepNet | 0.9914  | 0.9956    | 0.9945 | 0.9903   |

Figure 8(a) depicts that the approach DeepNet achieves the highest accuracy of 0.9914% at the 30th epoch. The convergence for accuracy training starts from 0.987% and ends at 0.9914%. After the accuracy reaches 0.9914%, the training accuracy convergence becomes constant. Similarly, Figure 8(b) depicts this approach achieving a minimum loss of 0.016% at the 30th epoch. Training loss starts at 0.028 and
ends at 0.016%. After the loss reaches 0.016%, the training loss convergence becomes constant. Figure 9 depicts that only 2615 of the normal instances are confused as attack instances, and only 3884 are confused as nonattack instances.

7. Comparative Analysis and Discussion

Table 10 compares the proposed work with existing techniques and recent studies. The research [34] shows high accuracy by using a random forest algorithm, but the other parameters such as precision, recall, and $F_1$-score give 100% results due to overfitting, and noise is still present in the dataset. A similar recent research [47] achieved high accuracy of 99.57% with a random forest algorithm, but in the area of precision, recall, and $F_1$-score, they achieved a 100% score, which shows that the overfitting problem still exists in the respective research. Furthermore, in [47], feature reduction is not utilized. Another research [48] which is based on SVM with Naive Bayes feature embedding shows the approach NB-SVM that achieves 93.75% accuracy, 94.73% recall, and 7.33 FAR with UNSW-NB15 dataset. In the research [49], the concept of feature-dimensionality reduction is implemented, and by using Stacked Autoencoder, 196-dimensional data features are reduced into 32-dimensional encoded data features.
The DeepNet approach achieves the highest results in all four instances of experiments. Figure 10 depicts the comparison between the highest achieved results with DeepNet from experiments on all four instances. In instance 1, all the features from Table 3 are selected, and DeepNet achieves the highest accuracy of 99.19%. In instance 2, the number of features is reduced to 25 based on the variance score of random forest regressor, and DeepNet achieves 99.19% without any reduction in accuracy. In instance 3, only 14 features are selected, and DeepNet still maintains 99.18% accuracy. This would be significantly effective for lightweight and real-time IDs, but after further feature reduction in instance 4, DeepNet achieves the accuracy of 99.14% with only 7 most important features.

8. Conclusion

We proposed a model to detect anomalies over the network. The performance is evaluated using the state-of-the-art UNSW-NB15 dataset using machine learning and an approach combining deep learning algorithms, i.e., the CNN algorithm with dense layers for better accuracy. Also, the performance is evaluated with conventional machine learning algorithms that include SGD, DT, RF, and XGB. The experimental results demonstrated that our proposed model is most efficient for the binary classification of network anomalies. The experimental result analysis shows that our proposed model gives promising performance based on accuracy, F1-score, and FAR with efficient execution time due to an effective feature reduction procedure. The highest accuracy achieved is 99.14% by using the DeepNet approach in maximum feature reduction for binary classification without performance depletion. After evaluating the lightweight anomaly detection dataset with only 7 features, it is clear that the DeepNet approach can be utilized to secure devices connected to public routers and the IoT devices connected to CPS with a remote server. In the domain of CPS of IoT devices, a lightweight, real-time computing intrusion detection system can be utilized to detect anomalies and protect against security violations under the CIA triad. In the future, we intend to make an online real-time network-based systematic framework or service through which the network
device can see if network logs consist of benign or anomalies before an intrusion occurs. This step would contribute positively to ensuring network devices’ security, i.e., the network module can be upgraded to an intrusion prevention system to detect and prevent network anomalies. We can also make a generic dataset with different categories and types of anomaly information that will merge all categories present in the latest research datasets. This step will allow different areas to use our generic dataset instead of multiple datasets for each anomaly classification. Furthermore, we will introduce the reduced features on routers in-home and public networks for real-time anomaly detection and achieve the desired goal by securing these networks from cybercriminals.

Data Availability
The (IoT Traffic) data used to support the findings of this study are included within the article.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

References
[1] A. Al-Abassi, A. N. Jahromi, H. Karimipour, A. Dehghantanha, P. Siano, and H. Leung, “A self-tuning cyber-attacks’ location identification approach for critical infrastructures,” IEEE Transactions on Industrial Informatics, vol. 18, no. 7, pp. 5018–5027, 2022.
[2] S. I. Imtiaz, S. U. Rehman, A. R. Javed, Z. Jalil, X. Liu, and W. S. Alnumay, “DeepAMD: detection and identification of Android malware using high-efficient deep artificial neural network,” Future Generation Computer Systems, vol. 115, pp. 844–856, 2021.
[3] B. Esmaeili, A. Azmoodeh, A. Dehghantanha, B. Zolfaghari, H. Karimipour, and M. Hammoudeh, “IoT deep malware threat hunting: from adversarial example detection to adversarial scenario detection,” IEEE Transactions on Industrial Informatics, p. 1, 2022.
[4] S. Ur Rehman, M. Khalij, S. I. Imtiaz et al., “Diddos: an approach for detection and identification of distributed denial of service (ddos) cyberattacks using gated recurrent units (gru),” Future Generation Computer Systems, vol. 118, pp. 453–466, 2021.
[5] L. Zhang, Y. Li, T. Jin et al., “SPCBIG-EC: a robust serial hybrid model for smart contract vulnerability detection,” Sensors, vol. 22, no. 12, p. 4621, 2022.
[6] A. R. Javed, M. O. Beg, M. Asim, T. Baker, and A. H. Al-Bayatti, “Alphalogger: detecting motion-based side-channel attack using smartphone keystrokes,” Journal of Ambient Intelligence and Humanized Computing, pp. 1–14, 2020.
[7] A. R. Javed, F. Shahzad, S. Ur Rehman et al., “Future smart cities: requirements, emerging technologies, applications, challenges, and future aspects,” Cities, vol. 129, article 103794, 2022.
[8] A. Abbas, A. R. R. Javed, A. Yasin, Z. Jalil, N. Kryvinska, and U. Tariq, “A large-scale benchmark dataset for anomaly detection and rare event classification for audio forensics,” IEEE Access, vol. 10, pp. 38885–38894, 2022.
[9] A. R. Javed, S. U. Rehman, M. U. Khan, M. Alazab, and H. U. Khan, “Betalogger: smartphone sensor-based side-channel attack detection and text inference using language modeling and dense multilayer neural network,” Transactions on Asian and Low-Resource Language Information Processing, vol. 20, no. 5, pp. 1–17, 2021.
[10] A. K. Roy, K. Nath, G. Srivastava, T. R. Gadekallu, and J. C.-W. Lin, “Privacy preserving multi-party key exchange protocol for wireless mesh networks,” Sensors, vol. 22, no. 5, p. 1958, 2022.
[11] M. Majid, S. Habib, A. R. Javed et al., “Applications of wireless sensor networks and internet of things frameworks in the industry revolution 4.0: a systematic literature review,” Sensors, vol. 22, no. 6, p. 2087, 2022.
[12] L. Zhang, W. Chen, W. Wang et al., “CBGRU: a detection method of smart contract vulnerability based on a hybrid model,” Sensors, vol. 22, no. 9, p. 3577, 2022.
[13] A. Rehman, I. Razzak, and X. Guandong, “Federated learning for privacy preservation of healthcare data from smartphone-based side-channel attacks,” IEEE Journal of Biomedical and Health Informatics, 2022.
[14] M. Nasir, A. R. Javed, M. A. Tariq, M. Asim, and T. Baker, “Feature engineering and deep learning-based intrusion detection framework for securing edge iot,” The Journal of Supercomputing, vol. 78, no. 6, pp. 8852–8866, 2022.
[15] A. R. Javed, Z. Jalil, S. A. Moqurrab, S. Abbas, and X. Liu, “Ensemble adaboost classifier for accurate and fast detection of botnet attacks in connected vehicles,” Transactions on Emerging Telecommunications Technologies, no. article e4088, 2020.
[16] P. Kumar, R. Tripathi, and G. P. Gupta, “P2IDF: a privacy-preserving based intrusion detection framework for software defined internet of things-fog (SDIoT-Fog),” in Adjunct Proceedings of the 2021 International Conference on Distributed Computing and Networking, pp. 37–42, 2021.
[17] A. Yazdinejad, A. M. Parizi, A. Dehghantanha, and M. S. Khan, “A kangaroo-based intrusion detection system on software-defined networks,” Computer Networks, vol. 184, article 107688, 2021.
[18] N. Rathi, R. Singla, and S. Tiwari, “A novel approach for designing authentication system using a picture based p300 speller,” Cognitive Neurodynamics, vol. 15, no. 5, pp. 805–824, 2021.
[19] P. Kumar, G. P. Gupta, and R. Tripathi, “Design of anomaly-based intrusion detection system using fog computing for iot network,” Automatic Control and Computer Sciences, vol. 55, no. 2, pp. 137–147, 2021.
[20] M. S. Alnages and F. Gebali, ”A survey on some currently existing intrusion detection systems for mobile ad hoc networks,” in The Second International Conference on Electrical and Electronics Engineering, Clean Energy and Green Computing (EEECEGC2015), pp. 12–18, 2015.
[21] S. Bhattacharya, P. K. Maddikunta, R. Kaluri et al., “A novel pca-firefly based xgboost classification model for intrusion detection in networks using gpu,” Electronics, vol. 9, no. 2, p. 219, 2020.
[22] S. Zargari and D. Voorhis, “Feature selection in the corrected kdd-dataset,” in 2012 Third International Conference on Emerging Intelligent Data and Web Technologies, pp. 174–180, Bucharest, Romania, 2012.
[23] A. R. Javed, M. Usman, S. U. Rehman, M. U. Khan, and M. S. Haghhighi, “Anomaly detection in automated vehicles using
multistage attention-based convolutional neural network,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4291–4300, 2021.

[24] W. Rials, *Top Cybersecurity Trends for 2021 and Beyond*.

[25] A. R. Javed, S. U. Rehman, M. U. Khan, M. Alazab, and T. Reddy, “CANIntelliIDS: detecting in-vehicle intrusion attacks on a controller area network using cnn and attention-based gru,” *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1456–1466, 2021.

[26] K. Zheng, Z. Cai, X. Zhang, Z. Wang, and B. Yang, “Algorithms to speedup pattern matching for network intrusion detection systems,” *Computer Communications*, vol. 62, pp. 47–58, 2015.

[27] Z. Cai, Z. Wang, K. Zheng, and J. Cao, “A distributed tcam coprocessor architecture for integrated longest prefix matching, policy filtering, and content filtering,” *IEEE Transactions on Computers*, vol. 62, no. 3, pp. 417–427, 2013.

[28] Y. Yu, J. Long, F. Liu, and Z. Cai, “Machine learning combining with visualization for intrusion detection: a survey,” in *Modeling Decisions for Artificial Intelligence*, pp. 239–249, Springer, Cham, Switzerland, 2016.

[29] P. Kumar, G. P. Gupta, and R. Tripathi, “A distributed ensemble design based intrusion detection system using fog computing to protect the internet of things networks,” *Journal of ambient intelligence and humanized Computing*, vol. 12, no. 10, pp. 9555–9572, 2021.

[30] R. Sommer and V. Paxson, “Outside the closed world: on using machine learning for network intrusion detection,” in *2010 IEEE Symposium on Security and Privacy*, pp. 305–316, Oakland, CA, USA, 2010.

[31] Y. Yang, J. Long, and Z. Cai, “Network intrusion detection through stacking dilated convolutional autoencoders,” *Security and Communication Networks*, vol. 2017, Article ID 4184196, 10 pages, 2017.

[32] Z. Chkirdene, A. Erbad, R. Hamila, A. Gouissim, A. Mohamed, and M. Hamdi, “Machine learning based cloud computing anomalies detection,” *IEEE Network*, vol. 34, no. 6, pp. 178–183, 2020.

[33] A. Chehri, I. Fofana, and X. Yang, “Security risk modeling in smart grid critical infrastructures in the era of big data and artificial intelligence,” *Sustainability*, vol. 13, no. 6, p. 3196, 2021.

[34] J. Olamananmi Mbeawondu, O. D. Alowolodu, J. O. Mbeawondu, and A. O. Adetunmobi, “Network intrusion detection system using supervised learning paradigm,” *Scientific African*, vol. 9, article e00497, 2020.

[35] F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, “ATSDL: a two-stage deep learning model for efficient network intrusion detection,” *Access*, vol. 7, pp. 30373–30385, 2019.

[36] F. A. Khan and A. Gumaei, “A comparative study of machine learning classifiers for network intrusion detection,” in *International Conference on Artificial Intelligence and Security*, pp. 75–86, 2019.

[37] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, “Towards a standard feature set of nids datasets,” 2021, https://arxiv.org/abs/2101.11315.

[38] J. Ren, J. Guo, W. Qian, H. Yuan, X. Hao, and H. Jingjing, “Building an effective intrusion detection system by using hybrid data optimization based on machine learning algorithms,” *Security and Communication Networks*, vol. 2019, Article ID 7130868, 11 pages, 2019.

[39] A. Roy and K. J. Singh, “Multi-classification of unsw-nb15 dataset for network anomaly detection system,” in *Algorithms for Intelligent Systems*, pp. 429–451, Springer, 2021.

[40] G. Grispos, *Cybersecurity Practice*, Springer, 2019.

[41] H. Ghadeer, “Cybersecurity issues in internet of things and countermeasures,” in *2018 IEEE International Conference on Industrial Internet (ICI)*, pp. 195–201, Seattle, WA, USA, 2018.

[42] N. Moustafa and J. Slay, “The significant features of the unsw-nb15 and the kdd99 data sets for network intrusion detection systems,” in *2015 4th international workshop on building analysis datasets and gathering experience returns for security (BADGERS)*, pp. 25–31, Kyoto, Japan, 2015.

[43] N. Moustafa and J. Slay, “Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set),” in *2015 military communications and information systems conference (MiCIS)*, pp. 1–6, Canberra, ACT, Australia, 2015.

[44] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, “Netflow datasets for machine learning-based network intrusion detection systems,” 2020, https://arxiv.org/abs/2011.09144.

[45] N. Elmrabit, F. Zhou, F. Li, and H. Zhou, “Evaluation of machine learning algorithms for anomaly detection,” in *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, pp. 1–8, Dublin, Ireland, 2020.

[46] S. V. Bozhokin and I. B. Suslova, “Wavelet analysis of non-stationary signals in medical cyber-physical systems (mcps),” in *Lecture Notes in Computer Science*, pp. 467–480, Springer, 2014.

[47] G. Kocher and G. Kumar, “Analysis of machine learning algorithms with feature selection for intrusion detection using unsw-nb15 dataset,” *International Journal of Network Security & Its Applications*, vol. 13, no. 1, pp. 21–31, 2021.

[48] G. Jie and L. Shan, “An effective intrusion detection approach using SVM with naive Bayes feature embedding,” *Computers Security*, vol. 103, article 102158, 2021.

[49] L. Guanyu and X. Tian, “An efficient communication intrusion detection scheme in ami combining feature dimensionality reduction and improved LSTM,” *Security and Communication Networks*, vol. 2021, Article ID 6631075, 21 pages, 2021.

[50] S. M. Kasongo and Y. Sun, “Performance analysis of intrusion detection systems using a feature selection method on the unsw-nb15 dataset,” *Journal of Big Data*, vol. 7, no. 1, pp. 1–20, 2020.