Journal of the Franklin Institute 362 (2025) 107440

Contents lists available at ScienceDirect

Journal of the Franklin Institute

journal homepage: www.elsevier.com/locate/fi

Intelligent parameter-based in-network IDS for IoT using UNSW-NB15 and BoT-IoT datasets

Muhammad Luqman [a], Muhammad Zeeshan [a,1,2,*], Qaiser Riaz [a,1], Mehdi Hussain [a,1], Hasan Tahir [a,1], Noman Mazhar [b], Muhammad Saffeer Khan [c]

[a] National University of Sciences and Technology (NUST), Sector H12, Islamabad, 44000, Pakistan
[b] Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia
[c] College of Engineering, University of North Carolina at Charlotte, NC 28223, USA

ARTICLE INFO

Keywords:
In-network IDS
Internet of Things (IoT) security
BoT-IoT
NB-15
Real-time intrusion detection
Cyber Threat Intelligence (CTI)

ABSTRACT

Keeping in mind the ubiquity of Internet of Things devices and the heterogeneity of protocols, intruders can easily meddle in an IoT network and jeopardize its security. This work proposes an intelligent intrusion detection system for IoT networks by applying machine learning algorithms, namely Random Forest and Support Vector Machine. We achieved high efficiency and reduced overfitting by balancing 11 classes in the BoT-IoT dataset. The dataset was prepared using preprocessing techniques that include handling missing values, one-hot encoding, and scaling the features for both binary and multi-class classification. An important feature selection and extraction process was used for RF and SVM model training with two hyperparameter tuning techniques. The proposed model achieves efficient training times for SVM and RF with three cross-validation techniques using 10 folds. In machine learning, an accuracy of 99.60% for binary and 98.31% for multi-class classification was achieved. For deep learning algorithms, the abnormal traffic in the UNSW BoT-IoT dataset consists of user datagram protocol and transmission control protocol packets. To address this imbalance, we merged two important network datasets, UNSW BoT-IoT and NB-15, using feature engineering based on common features, and used the deep learning model LSTM for training. After applying preprocessing techniques and important feature extraction, we performed multiple iterations with 10 and 16 epochs for model validation and achieved an accuracy of 99.89% and 99.97% for multi-class classification.

1. Introduction

Intruders penetrate a network to perform unauthorized and illegal activities like stealing valuable data and gaining access to network resources. To protect against network intrusions, an Intrusion Detection System (IDS) is required to monitor network traffic to identify intrusions and safeguard networks. The intrusion detection system can be either anomaly-based or signature-based. A signature-based IDS identifies attacks whose pattern or signature is already known by the system. It detects attacks based on predetermined patterns or rules, such as byte sequences. For large and distributed networks like the IoT, a significant number of rules are vital for an efficient signature-based IDS. A consequence of this is that zero-day attacks cannot be identified because their

* Corresponding author.
E-mail address: muhammad.zeeshan@seecs.edu.pk (M. Zeeshan).
[1] SMIEEE.
[2] Researcher.

https://doi.org/10.1016/j.jfranklin.2024.107440
Received 30 September 2023; Received in revised form 9 June 2024; Accepted 3 December 2024
Available online 11 December 2024

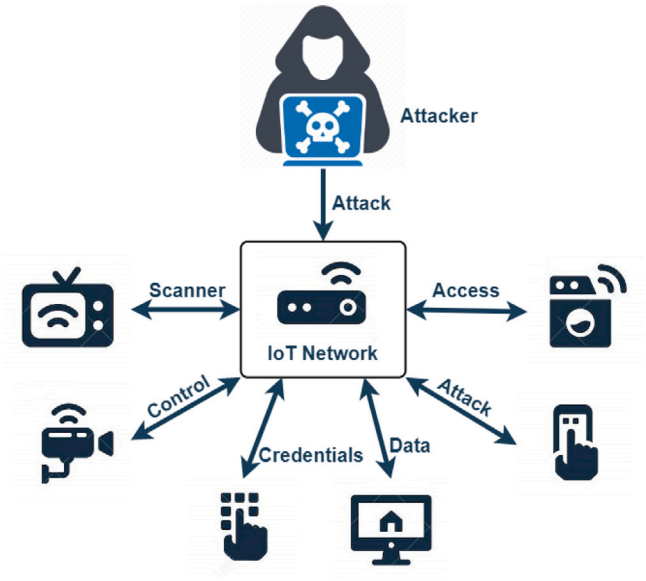

Fig. 1. Attacks in IoT.

signatures are not available. To overcome these issues, this work proposes an anomaly-based IDS, which can detect unknown attacks by studying abnormalities in regular network traffic conditions. Thus, a well-trained machine learning model is required to sense threats without human intervention.

IoT comprises a network of interconnected smart devices, encompassing sensors, actuators, software, and various computing devices. These smart devices are utilized to sense a range of physical stimuli and communicate/exchange data among themselves. The correct and reliable exchange of data is essential in an IoT network, but unfortunately, there are numerous opportunities for attackers to steal information and compromise system integrity. Due to the ubiquity of IoT devices and the heterogeneity of protocols, intruders can easily meddle in an IoT network and jeopardize its security. With the massive upsurge in IoT devices, the volume of network traffic has also grown, which has increased the number of network attacks. Cyber attackers can gain unauthorized access to IoT infrastructure, broadcast messages, and inject false information into the network as shown in Fig. 1.

In recent years, Machine Learning (ML) has proved very effective in detecting malicious traffic in IDS. ML algorithms are trained with significant features from available attack datasets so that they can predict similar malicious data. Several supervised and unsupervised ML algorithms can be applied to discover the source, pattern, and nature of attacks [1]. Machine learning algorithms and techniques are efficient in recognizing possible attacks [2]. For the development of an IoT IDS, the training dataset must be carefully chosen, and IoT traffic must be present in the input dataset [3]. There are several benchmark network datasets available that academics have used to evaluate and analyze their suggested IDS systems. Some of the proposed tactics identify network assaults efficiently, but within a limited context, and even fewer of these works have evaluated their algorithms on real-time IoT information. Recent studies on various datasets do not accurately represent IoT scenarios [2]. However, in this work, we utilize the UNSW Bot-IoT [4] and UNSW NB-15 [5] datasets in our experiments. Bot-IoT is acquired from an actual IoT platform and consists of realistic IoT benign and attack traffic.

In supervised learning, this research applies data balancing and preprocessing to the UNSW BoT-IoT dataset, handling missing values and using one-hot encoding on the balanced dataset. Parameter optimization is achieved via Grid Search CV and Randomized Search CV. Using the random forest feature importance technique, we identify the 27 most critical features. Support Vector Machine (SVM) and Random Forest (RF) models are utilized for binary and multi-class classification, employing cross-validation methods such as K-Fold, Stratified K-Fold, and Stratified Shuffle Split to train and assess model performance. Post-training, the model is tested on realistic IoT datasets.

In unsupervised learning, this research studies a combined dataset created by merging attack traffic from Bot-IoT and normal traffic from NB-15 based on feature comparison and engineering. This unique technique achieves data balancing for a large dataset. Preprocessing involves imputing missing values and one-hot encoding. The Long Short-Term Memory (LSTM) model is used for training, leading to better accuracy with two neural network layers and one dense layer with three and five-class parameter settings. After training, testing is performed on a realistic dataset, and results are reported through matrices and graphs. This research contributes the following:

- Large datasets were balanced by merging malicious and normal records from Bot-IoT and NB-15, reducing overfitting and improving model generalization.

- A novel intelligent IDS was proposed, employing unique preprocessing methods, feature extraction and engineering from Bot-IoT and NB-15, and feature importance using Random Forest algorithms, resulting in enhanced accuracy and reduced training time.
- Cross-validation and hyperparameter tuning were performed to validate the training of RF, SVM, and LSTM models, optimizing model accuracy and reducing computational cost.
- The proposed intelligent IDS demonstrated outstanding accuracy: 99.60% for binary and 98.31% for multi-class classification in machine learning after dataset balancing. Additionally, merging the BoT-IoT and NB-15 datasets yielded exceptional multi-class classification accuracy of 99.89% and 99.97% using LSTM in deep learning.

The rest of the paper is structured as follows: Section 2 reviews the related literature. Section 3, IIDS-IoT, introduces the proposed intrusion detection system, detailing the datasets, preprocessing steps, data merging, handling data imbalance, feature extraction, and one-hot encoding. Section 4, Experiments and Results, discusses the outcomes of both machine learning and deep learning approaches. Finally, Section 5 provides the conclusion.

2. Related work

2.1. IDS: a general review

Intrusion Detection Systems (IDS) can be classified into two types: Host-based IDS and Signature-based IDS [6,7]. Host-based Intrusion Detection Systems (HIDS) are installed on a host device to detect suspicious activities on that device. HIDS can be further divided into two subtypes: Signature-based IDS and Anomaly-based IDS. Signature-based IDS detects attacks by matching them against known patterns or signatures. Earlier researchers developed signature-based IDS [8], but recent advancements have led to more sophisticated signature-based NIDS [9] to address evolving IoT scenarios. Researchers recommend using an ensemble or hybrid approach in ML-based IDS, combining multiple classifiers for efficient performance [10]. Recent work on NIDS focuses on achieving greater accuracy and developing precise feature selection methods. Selecting contributory features from the dataset is crucial, as it significantly impacts model training. A hybrid feature selection technique ensures accurate feature selection [11] and builds IDS using ensemble feature selection techniques. Combining different feature selection methods results in higher accuracy and reduced training time.

2.2. AI-based intrusion detection systems

Artificial Neural Networks (ANNs) are the most widely used classifiers in intrusion detection systems, with numerous NIDS models already developed using ANNs [12,13]. Various Neural Network models exist, including Multi-Layer Perceptron (MLP) [14], Recurrent Neural Network, and Convolutional Neural Network [15]. Recently, intrusion detection models have been formulated using supervised SVM algorithms with different datasets [16,17]. The Random Forest algorithm is a significant classifier for building IDS [18]. Researchers have trained multiple supervised learning algorithms to analyze the accuracy and performance of different classifiers [18,19]. Using a single dataset and feature set, several supervised learning algorithms have been trained, and performance comparisons have been made using popular metrics like F1 score, precision, recall, and specificity [20]. Intrusion detection models have also been developed using deep learning techniques [21], which typically require substantial amounts of data for effective training.

Zhu et al. [22] created multi-class deep learning models utilizing Adam as an optimizer in their neural network and implemented an early-stop strategy to enhance training. They used datasets such as BoT-IoT, ToN-IoT, and ISCX, achieving over 98% accuracy on ToN-IoT and BoT-IoT, and over 91% on ISCX. Sheeraz Ahmed et al. [23] designed a DDoS attack detection and prevention system using deep learning on the CTU-13 dataset. They trained, validated, and tested their models with a 70% training, 15% validation, and 15% testing split, achieving 98.99% accuracy with the MLP algorithm. Wang et al. [24] applied deep neural networks (DNN), convolutional neural networks (CNN), and other models on the CIC-IDS2018 dataset, obtaining over 98% accuracy in multiclass classification. CIC-IDS2018 and CIC-IDS2017 datasets have been utilized in similar studies [25,26]. Research by Romeiras et al. [25] focuses on how Peregrine manages computational resources in switch data planes under constraints, ensuring efficient performance. Meanwhile, Jafri's [26] solution enables packet classification in the data plane, promoting self-driving networks capable of tasks like abnormality recognition and performance measurement.

Ankit and Ritika [27] present an intrusion detection system based on a deep neural network (DNN), utilizing the NSL-KDD, IDS-2017, UNSW-NB15 and BoT-IoT datasets. To address class imbalance in the datasets, they implemented the bagging ensemble learning method, resulting in over 98% accuracy for IDS-2017, NSL-KDD and BoT-IoT, and approximately 96.7% accuracy for NB-15. Romeiras et al. [25] and Jafri et al. [26] propose ML based detector for malicious traffic detection, Peregrine which suits to Terabit network and achieved more then 80% against most attacks. Anushiya et al. [28] suggest a unique genetic algorithm and faster recurrent convocation neural network for intrusion detection systems; moreover, the authors describe an effective feature selection approach that obtained 93.78 percent accuracy using the BoT-IoT dataset. Kemp et al. [29] integrate four datasets: HTTP POST, Slow Read, Apache Range Header and Slowloris to prevent DoS attacks on the application layer and identify patterns using PCA. To successfully defend against several DOS assaults, authors train data using machine learning techniques. Taher et al. [30] studies focus on data streaming mining algorithms, and these algorithms will handle and solve the problem of unbalanced datasets and outperform others.

Chauhan and Atulkar [31] explored the prevention and identification of DDoS attacks on Software Defined (SD) IoT traffic through feature extraction and machine learning using the LGBM model. Diao et al. [32] presented a deep-learning EC-GCN model that converts encrypted communication into graphs for detecting and classifying encrypted traffic flow, achieving a 5%–20% increase in accuracy. Prajapati et al. [33] proposed a network intrusion detection system (NIDS) using the NSL-KDD dataset, with their deep learning (CNN) model outperforming existing approaches. Liu et al. [34] investigated unsupervised deep learning-based methods for time series anomaly detection in IoT applications, utilizing data preprocessing, feature extraction, clustering, and pattern inference.

Vitorino et al. [35] developed an IoT network intrusion detection system based on IoT-23 and Bot-IoT datasets, employing unsupervised (Isolation Forest) and supervised (Random Forest and Extreme Gradient Boosting) machine learning algorithms, with XGB achieving the highest accuracy. Awajan at al. [36] aimed to develop an intrusion detection and mitigation solution for IoT traffic, achieving over 93% accuracy using a deep learning (DNNs) model and feature extraction module. Kalutharage et al. [37] proposed an AI-based model (RF, DT, and DNN) for DDoS attacks on layer three traffic in IoT networks, achieving high accuracy using the USB-IDS dataset. Bouke et al. [38] conducting an empirical evaluation and comparison of thirteen distinct machine learning models for intrusion detection in 5G networks using 5G-NIDD dataset.

2.3. IDSs for IoT environments

Considering the security vulnerabilities highlighted in [39], it is evident why there is a crucial requirement for an effective IDS to address the concerns in IoT networks, and the researchers utilized the BoT-IoT and NSL-KDD datasets. When identifying numerous threats in IoT environments, supervised learning algorithms have exhibited promising performance as several scholars have employed these advanced supervised classifiers to construct intrusion detection models for IoT environments [40,41]. In [42], the authors have leveraged five different supervised ML algorithms along with the MLlib library of Apache Spark for intrusion detection in IoT. The work in [43] has utilized SVM, and the study in [44] has employed Random Forest in their IoT-based IDS models. The model proposed in work [45] uses a decision tree to detect Botnet attacks, whereas [46] focuses on identifying man-in-the-middle attacks in IoT by employing four supervised machine learning algorithms. The objective of Ali et al. [47] is to provide an overview of the significance of concentrating and mitigating botnet attack detection in the present context of attentive development of IoT devices.

Dos and DDoS attacks are the most prevalent and severe types of network attacks, and many researchers have made efforts to detect and mitigate these attacks. To protect IoT from DoS and DDoS attacks, numerous models have been constructed [48,49]. ANN is trained in a manner that enables them to detect multifarious attacks in IoT. The works of [50,51] presented such intrusion detection models that leverage an ANN classifier. Deep learning algorithms used for intrusion detection for IoT environments achieve high performance when the data is diverse. Deep learning models are typically trained and tested to detect various types of attacks in IoT [52–54]. Tongtong Su et al. [55] developed a network intrusion detection system with an accuracy of 84.25% using the NSL-KDD dataset and the LSTM algorithm. Ali et al. [47] developed a machine learning and deep learning algorithm using the NB-15 dataset for the effective detection of botnet attacks in the IoT environment and achieved 96.98% accuracy. Amine et al. [56] studies primary objective is to discuss security issues in AI and IoT, notably in the context of 6G communication networks, and utilized deep learning classifiers like DNNs, CNNs, and RNNs, and the LSTM and GRUs neural network models to train on the Edge-IIoTset dataset.

Sharmila et al. [57] approach to designing a quantized autoencoder (QAE) model for IDS in IoT devices by utilizing the RT-IoT2022 dataset, which contains normal and attack traffic from several IoT devices. The principal objective of the study is to reduce the computational costs of implementing AI models in such devices without compromising the accuracy rate. Farouk et al. [58] research aims at establishing the optimal machine learning-based approach that would aid in the identification of network threats, especially insider attacks on Local Area Networks (LANs), using the NSL-KDD dataset. The concern marked as an insider threat challenge in cybersecurity was the primary research question of the paper, where the goal was to identify a stable, accurate machine learning classifier for detecting such threats. Shafi et al. [59] propose a multi-layered DDoS detection model using a new cloud-based DDoS dataset to bridge the existing gaps and have thus developed the BCCC-cPacket-Cloud-DDoS-2024 dataset, which offers more than eight user activities and 17 DDoS-formulated assaults. The author's main idea is to enhance DDoS detection performance by introducing a set of large datasets and a proper detection model.

2.4. IDSs based on UNSW BoT-IoT

Constructed from a pragmatic test-bed setup, the BoT-IoT dataset comprises genuine and emulated IoT network traffic that incorporates various attack types. X. Wang et al. [60] leverage deep learning for traffic classification. They have utilized generic features derived from header field information of data packets and performed binary and multi-class classification. The model successfully identifies DoS/DDoS, Reconnaissance, and data theft attacks. Ahmad et al. [61] have employed a Bijective soft set method to select the most effective machine learning algorithm for intrusion detection in IoT. Ahmad et al. [21] have developed a network intrusion detection system using supervised machine learning algorithms using the BoT-IoT data-set, and feature selection is performed using the wrapper-based method, and their model achieves an accuracy of 95%.

McDermott et al. [62] compared the efficacy of ML classifiers in intrusion detection using the BoT-IoT dataset. Alkadi et al. [63] have developed a system to detect intrusions in IoT and cloud-based environments. The system's performance has been validated using NB-15 and BoT-IoT datasets. Giampaolo Bovenzi et al. [64] propose a model that predicts anomalies using a MultiModal Deep AutoEncoder and classifies attacks using soft output classifiers. Model validation is done using the BoT-IoT dataset. Authors in [65]

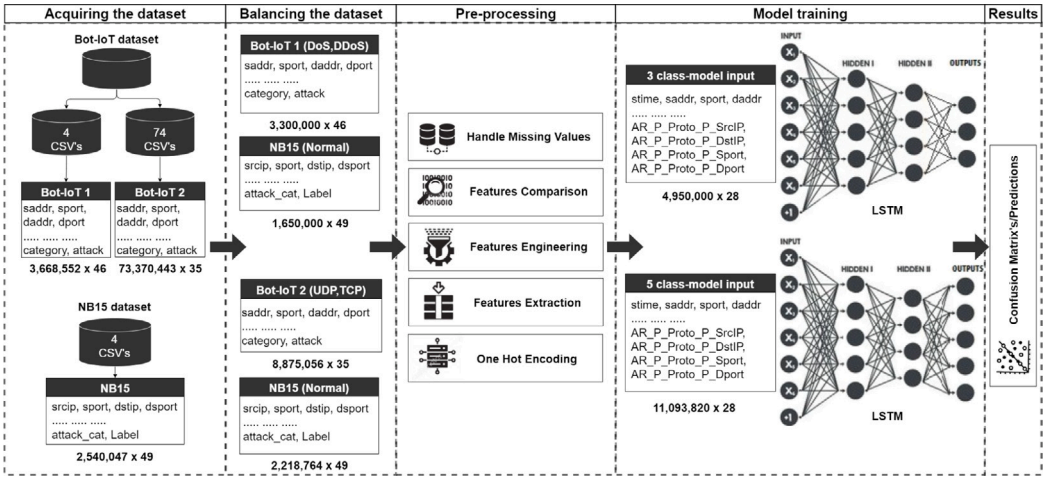

Fig. 2. IIDS-IoT proposed model.

have studied their intrusion detection model for IoT using a deep learning approach. The authors have used the transfer learning rule to design binary and multi-class classification models.

Alkadi et al. [63] have proposed a blockchain-based framework for intrusion detection systems using smart contracts in IoT networks. For training, they used BoT-IoT, with 60 hidden nodes, which yields an accuracy of 98.91%. Ge et al. [60] developed a novel deep learning-based intrusion detection framework for IoT networks and obtained an accuracy of 88% to 99%. Leevy et al. [66] compared eight state-of-the-art machine learning algorithms and evaluated their performance for intrusion detection systems using the BoT-IoT dataset. Their multi-class classification models achieve an accuracy of 88% to 98%. Lanitha et al. [67] have emphasized detecting DoS attacks through their intrusion detection model. They have used deep learning techniques with the BoT-IoT dataset.

2.5. IDSs based on UNSW NB-15

Machine learning techniques like Support Vector Machine, Decision Tree, Random Forest, and Naive Bayes have been utilized to develop NIDS using NB-15 [26,49,68–70], and Ali et al. [47] employed ANN, CNN, LSTM, and RNN to address the ACLR problem. Numerous researchers have leveraged Deep learning techniques with NB-15 to construct a network intrusion detection system [71–73]. Hyeokmin et al. [74] in their work have put forth a NIDS model which operates by utilizing linear information using NB-15 and LSTM, yielding reasonable accuracy. Pooja et al. [75] have used KDDCUP-99 and UNSW-NB15 datasets in their experiments to design an intrusion detection system. Their model employs Bi-directional LSTM, achieving an accuracy of 99%. NB-15 and KDD datasets lack realistic IoT traffic, making BoT-IoT the first with such data. Beyond accuracy, model correctness requires a confusion matrix to identify overfitting [75]. However in this work we address class imbalance by merging the unbalanced UNSW NB-15 and BoT-IoT datasets based on common features, maximizing the use of packets for balanced training. Yu Yan et al. [76] developed a Network Intrusion Detection model based on a stacked auto-encoder. For the experiment, NB-15 and LSTM have been used, giving an accuracy of about 92%. More recently, Yakub et al. [77] used the min–max concept with PCA to implement machine learning models and achieved an accuracy of 99.98% for NB-15. Zheng et al. [70] focuses on Planter to enable automatic in-network model and compute transfer based on trained machine-learning models, including PCA for programmable network devices. Yin et al. [78] used 10% of the NB-15 dataset and employed random forest feature importance to select and extract 23 features, achieving an accuracy of 84.24% through MLP.

3. Intelligent IDS for IoT networks (IIDS-IoT)

This section presents details and the design of the proposed intelligent intrusion detection system for the IoT (IIDS-IoT) shown in Fig. 2. This includes dataset description, preprocessing, and both machine learning and deep learning models.

3.1. Acquiring the datasets for IIDS

We used the BoT-IoT and NB-15 datasets for the purpose model (IIDS-IoT).

3.1.1. UNSW Bot-IoT

BoT-IoT is a realistic dataset that is composed of IoT traffic traces. It has been generated by Nickolaos Koroniotis et al. [4] at the University of New South Wales, Canberra, Australia. This is the first publicly available dataset that contains realistic IoT-generated

Table 1
Bot-IoT Dataset.

Class	No. of records	% of total dataset
Normal	9543	0.0130
Abnormal/Malicious	73,360,791	99.9868
Total	**73,370,334**	

Table 2
NB-15 Dataset.

Class	No. of records	% of total dataset
Normal	2,218,764	87.3514
Abnormal	321,283	12.6487
Total	**2,540,044**	

Table 3
Balanced Bot-IoT dataset, Used for machine learning.

Class	No. of records
Normal	9543
Reconnaissance_OS_Fingerprint	11,250
Reconnaissance_Service_Scan	11,250
DDoS_TCP	11,250
DDoS_UDP	11,250
DDoS_HTTP	11,250
DoS_TCP	11,250
DoS_UDP	11,250
DoS_HTTP	11,250
Theft_Keylogging	1469
Theft_Data_Exfiltration	118
Total	**101,130**

traffic. The IoT devices used in the test-bed of the BoT-IoT dataset include a weather station, a smart refrigerator, a smart thermostat that controls cooling systems, motion-activated smart lights, and a smart garage door. This smart home configuration has been used to generate real-time IoT traffic into which diverse types of attacks have been incorporated. 3 types of attacks have been introduced into the system: information gathering, denial of service, and information theft attacks. Each of the attacks has its subcategories. The dataset is available in CSV files containing more than seventy-two million records and 46 extracted features. The complete dataset records contain many subcategories in the malicious class along with the normal class; however, Table 1 shows normal and malicious traffic and the number of records in each.

3.1.2. UNSW NB-15

NB-15 has been generated by Nour Moustafa and Jill Slay [5] at the University of New South Wales, Canberra, Australia. The NB-15 dataset is also huge but highly imbalanced. Normal traffic comprises about 87.35% of the total, whereas abnormal traffic is about 12.65%. Moreover, the distribution of abnormal traffic is also highly uneven. Only 5% of the abnormal traffic on NB 15 consists of DoS packets, while 67% of the abnormal traffic consists of generic packets. Table 2 shows normal and abnormal traffic and the number of records in them.

3.2. Machine learning based IDS

3.2.1. Dataset balancing

Upon obtaining the Bot-IoT dataset, we explore and find that the normal packets from Bot-IoT are only 9543 (i.e., 0.013%) in number. hence, it is a highly imbalanced dataset due to a very small number of normal traffic packets. We under-sample the other classes and take 11,250 samples for all the other eight classes, except the two classes of *Theft* category packets because they are already small in number as shown in the Table 3. So, the 11-class dataset after performing the packet balancing is used for binary and multi-class machine learning models.

3.2.2. Preprocessing

In Bot-IoT, there are more than 550 missing values in both *sport* and *dport* columns, as shown in Fig. 3. The machine learning models and algorithms used in this work do not support missing values in training data. Missing values can be handled in a variety of ways during the pre-processing data cleaning setup, including deleting rows with missing values or imputed missing values. So, we opt for the imputation method, which prevents the loss of data. As a result, we assigned *0 (zero)* to all null values in the source port and destination port columns in our dataset. In terms of data type resolution, we performed conversions on *proto*, *flgs*, *state*, *saddr*, *sport*, *daddr*, and *dport* columns through one-hot encoding.

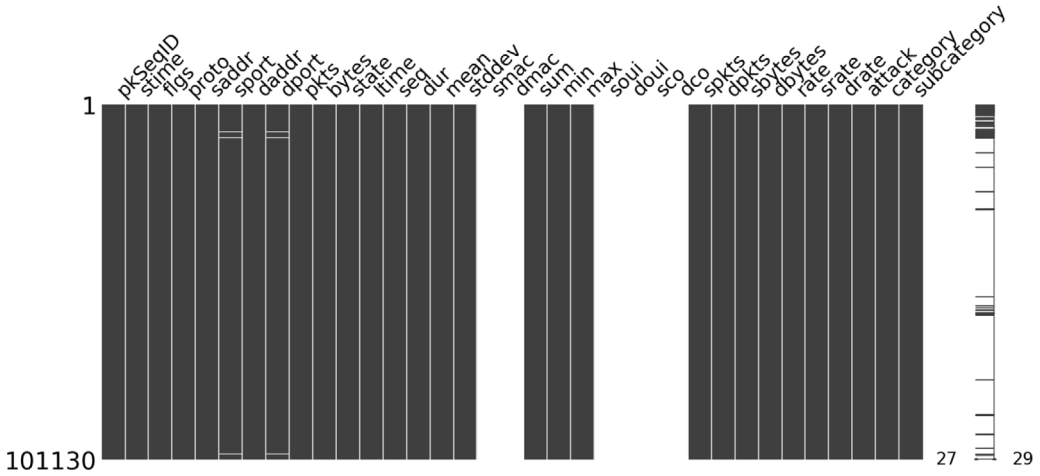

Fig. 3. Missing values in Bot-IoT dataset.

Table 4
Balanced and Merged two data-set for three class Deep Learning.

Class	No. of records	Records from dataset
Normal	1,650,000	NB-15
DDoS	1,650,000	BoT-IoT
DoS	1,650,000	BoT-IoT
Total	**4,950,000**	

3.2.3. Feature extraction and selection

To reduce the model complexity and improve the training of models, prominent 28 features are extracted through the random forest feature importance graph. So the feature *daddr* is contributing more than 0.25 towards classification. This over-contributing feature is the cause of over-fitting. After removing *daddr* we are left with 27 features.

3.2.4. Parameters tuning and cross-validation

For hyper-parameter tuning, *Randomized Search CV* and *Grid Search CV* has been used and it is noticed that both methods gave us the same best parameters. *n_estimators = 40*, *max_features = sqrt*, *criterion = entropy*, *max_depth = 50*, *min_samples_leaf = 1*, *min_samples_split = 10*, *bootstrap = False* are the best parameters for RF, and *C = 0.1*, *kernel = poly*, *probability = True*, *gamma = 1* are the best parameters for SVM. For better training and removal of over-fitting, *k-fold*, *stratified k-fold*, and *stratified shuffle split* cross-validation techniques are used. Furthermore, the dataset is split into three parts: 10% for the actual testing, 10% for validation of the model, and 80% for the training of the model.

3.3. Deep learning based IDS

3.3.1. Dataset balancing

A contribution of this research is in terms of balancing the dataset by merging malicious traffic from Bot-IoT and normal traffic from NB-15. To achieve data balancing for three class classifications, this research focuses on 97.5% of malicious traffic from Bot-IoT, consisting of DoS and DDoS. Hence 1,650,000 records are extracted from normal traffic in NB-15 and merged with malicious records (1,650,000 records for DoS and 1,650,000 records for DDoS) from Bot-IoT, as shown in Table 4.

For five class classifications, the focus is on 99.93% of malicious traffic from Bot-IoT, consisting of TCP and UDP. Hence all normal records are extracted from NB-15 which are 2,218,764 in number and merge with the Bot-IoT malicious records (DoS_TCP, DoS_UDP, DDoS_TCP, and DDoS_UDP). Hence all classes are equally balanced for better training, as shown in the Table 5.

3.3.2. Preprocessing

The missing values in which *smac*, *dmac*, *soui*, *doui*, *sco*, and *dco* are null features, as shown in Fig. 4 have been dropped from Bot-IoT. More than 200 null values in Bot-IoT are filled by putting special characters (0) in *sport* and *dport*. In NB-15, *attack_cat* is null for normal traffic since only the normal packets are being considered hence *attack_cat* has been eliminated.

Table 5
Balanced and Merged two datasets for five class Deep Learning.

Class	No. of records	Records from dataset
Normal	2,218,764	NB-15
DDoS_UDP	2,218,764	BoT-IoT
DDoS_TCP	2,218,764	BoT-IoT
DoS_UDP	2,218,764	BoT-IoT
DoS_TCP	2,218,764	BoT-IoT
Total	**11,093,820**	

Table 6
Complexity of algorithm's during preprocessing of dataset.

Algorithms	Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 4	Algorithm 5
Complexity	$O(n)$	$O(n)$	$O(n^2)$	$O(n^2)$	$O(n^2)$

Algorithm 1 Return the index positions based on value for data-frame.

Require: $dfObj$ can be a part or complete pandas data frame.
Require: $value$ for comparison in the column.
Require: $nameofCol$ on the bases need to find the index.
procedure GETINDEXES($dfObj, value, nameofCol$)
  Initialize $listOfPositions \leftarrow list()$
  for all $rows \in dfObj$ **do**
    if value of $nameofCol == value$ **then**
      $listOfPositions \leftarrow$ index for each $row \in rows$ ▷ Hint: append() in python
    end if
  end for
  return $listOfPositions$
end procedure
Ensure: $dfObj$, $value$ and $nameofCol \neq null$

Algorithm 2 Calculate the two column values in the dataset.

Initialize $newdataframe \leftarrow NULL$
procedure COLUMNDATAADD( )
$seg, listsegdata, columnforindex, csum1, csum2, newcolname$
  for all $segdata \in listsegdata$ **do**
    $getlistofPositions \leftarrow$ GETINDEXES($seg, segdata, columnforindex$) ▷ Calling Algo 1.
    for all $pos \in getlistofPositions$ **do**
      $newdataframe[newcolname] \leftarrow csum1[pos] + csum2[pos]$
    end for
  end for
  return $newdataframe$
end procedure
Ensure: seg, $listsegdata$, $columnforindex$, $csum1$, $csum2$ and $newcolname \neq null$

3.3.3. Features comparison and engineering

In feature comparison, it has been noticed that 12 attributes are common in both the Bot-IoT and NB-15 datasets which can be used for feature engineering. In a classifier, features or attributes play a vital role in classifier accuracy, so creating those attributes that are relevant to an IoT network from the given data leads to improved accuracy. After comparison, calculating the values of attribute, total number of bytes per source ip (*TnBPSrcIP*) using Algorithm 3 and base attribute *saddr*, *bytes* in the dataset, total number of bytes per destination ip (*TnBPDstIP*) using Algorithm 3 and *daddr*, *bytes*, total number of packets per source ip (*TnP_PSrcIP*) using Algorithm 3 and *saddr*, *pkts*, total number of packets per destination ip (*TnP_PDstIP*) using Algorithm 3 and *daddr*, *pkts*, total number of packets per protocol (*TnP_PerProto*) using Algorithm 3 and *proto*, *pkts*, total number of packets per dport (*TnP_Per_Dport*) using Algorithm 3. We computed *dports*, *pkts*, (*N_IN_Conn_P_SrcIP*) i.e, the average rate per protocol per source ip using Algorithm 4 and *saddr*, *bytes*, *pkts*, *dur*, the average rate per protocol per destination ip (*N_IN_Conn_P_DstIP*) using Algorithm 4 and *daddr*, *bytes*, *pkts*, *dur*.

Algorithm 3 Calculate the required columns sum in the dataset.

Initialize $newdataframe \leftarrow NULL$
procedure ColumnData( )
$seg, listsegdata, columnforindex, columnforsum, newcolname$
  Initialize $sumofValue \leftarrow 0$
  for all $segdata \in listsegdata$ **do**
    $getlistofPositions \leftarrow$ getIndexes($seg, segdata, columnforindex$) ▷ Calling Algo 1.
    for all $position \in getlistofPositions$ **do**
      $sumofValue$ += values of $seg[columnforsum]$ where equal to $position$
    end for
    for all $pos \in getlistofPositions$ **do**
      where position of $newdataframe[newcolname]$ is equal to $pos \leftarrow sumofValue$
    end for
  end for
  return $newdataframe$
end procedure
Ensure: seg, $listsegdata$, $columnforindex$, $columnforsum$ and $newcolname \neq null$

Algorithm 4 Calculate the required column count in the dataset.

Initialize $newdataframe \leftarrow NULL$
procedure ColumnDataCount( )
$seg, listsegdata, columnforindex, columnforsum, newcolname$
  Initialize $countofIndex \leftarrow 0$
  for all $segdata \in listsegdata$ **do**
    $getlistofPositions \leftarrow$ getIndexes($seg, segdata, columnforindex$) ▷ Calling Algo 1.
    for all $position \in getlistofPositions$ **do**
      $countofIndex$ += 1 for all $position$
    end for
    for all $pos \in getlistofPositions$ **do**
      where position of $newdataframe[newcolname]$ is equal to $pos \leftarrow countofIndex$
    end for
  end for
  return $newdataframe$
end procedure
Ensure: seg, $listsegdata$, $columnforindex$, $columnforsu$ and $newcolname \neq null$

Number of inbound connections per source ip (*AR_P_Proto_P_SrcIP*) are computed using Algorithm 5 and *saddr*, *pkts*, *dur*, number of inbound connections per destination ip (*AR_P_Proto_P_DstIP*) are computed using Algorithm 5 and *daddr*, *pkts*, *dur*, average rate per protocol per sport (*AR_P_Proto_P_Sport*) using Algorithm 5 and *sport*, *pkts*, *dur*, and average rate per protocol per dport (*AR_P_Proto_P_Dport*) using Algorithm 5 and *dport*, *pkts*, *dur* are derived attributes from Bot-IoT dataset. Furthermore, Algorithm 2 to Algorithm 5 are using Algorithm 1 as a nested call. The entire procedure is repeated for the next dataset, total count of packets in transaction (*pkts*) using Algorithm 2 and base attribute *spkts*, *dpkts* in the dataset, total number of bytes in transaction (*bytes*) using Algorithm 2 and *sbytes*, *dbytes*, total number of bytes per source ip (*TnBPSrcIP*) using Algorithm 3 and *srcip*, *bytes*, total number of bytes per destination ip (*TnBPDstIP*) using Algorithm 3 and *dstip*, *bytes*, total number of packets per source ip (*TnP_PSrcIP*) using Algorithm 3 and *srcip*, *pkts*.

Total number of packets per destination ip (*TnP_PDstIP*) computed using Algorithm 3 and *dstip*, *pkts*, total number of packets per protocol (*TnP_PerProto*) using Algorithm 3 and *proto*, *pkts*, total number of packets per dport (*TnP_Per_Dport*) using Algorithm 3 and *dports*, *pkts*, average rate per protocol per source ip (*N_IN_Conn_P_SrcIP*) using Algorithm 4 and *srcip*, *bytes*, *pkts*, *dur*, average rate per protocol per destination ip (*N_IN_Conn_P_DstIP*) using Algorithm 4 and *dstip*, *bytes*, *pkts*, *dur*, number of inbound connections per source ip (*AR_P_Proto_P_SrcIP*) using Algorithm 5 and *srcip*, *pkts*, *dur*, number of inbound connections per destination ip (*AR_P_Proto_P_DstIP*) using Algorithm 5 and *dstip*, *pkts*, *dur*, average rate per protocol per sport (*AR_P_Proto_P_Sport*) using Algorithm 5 and *sport*, *pkts*, *dur*, and average rate per protocol per dport (*AR_P_Proto_P_Dport*) using Algorithm 5 and *dport*, *pkts*, *dur* are 14 derived attributes from NB-15 dataset. The computational complexity of Algorithm 3, 4 and 5 is $O(n^2)$ as shown in Table 6. Attributes values to be computed based on attribute values, so Algorithm 1 gives us the exact position of that value. While feature engineering, Algorithm 1 provides us with functionality to find the position of the values that we need for computation from base features and the computational complexity of Algorithm 1 and 2 is $O(n)$.

Algorithm 5 Calculate the average of the required column in the dataset.

Initialize $newdataframe \leftarrow NULL$
procedure ColumnDataAvg( )
$seg, listsegdata, columnforindex, csum1, csum2, newcolname$
  Initilize $sumofValue1 \leftarrow 0$
  Initialize $sumofValue2 \leftarrow 0$
  Initialize $res \leftarrow 0$
  for all $segdata \in listsegdata$ **do**
    $getlistofPositions \leftarrow$ getIndexes($seg, segdata, columnforindex$) ▷ Calling Algo 1.
    for all $position \in getlistofPositions$ **do**
      $sumofValue1$ += values of $seg[csum1]$ where equal to $position$
      $sumofValue2$ += values of $seg[csum2]$ where equal to $position$
    end for
    $res \leftarrow \frac{sumofValue1}{sumofValue2}$
    for all $pos \in getlistofPositions$ **do**
      where position of $newdataframe[newcolname]$ is equal to $pos \leftarrow res$
    end for
  end for
  return $newdataframe$
end procedure
Ensure: seg, $listsegdata$, $columnforindex$, $csum1$, $csum2$ and $newcolname \neq null$

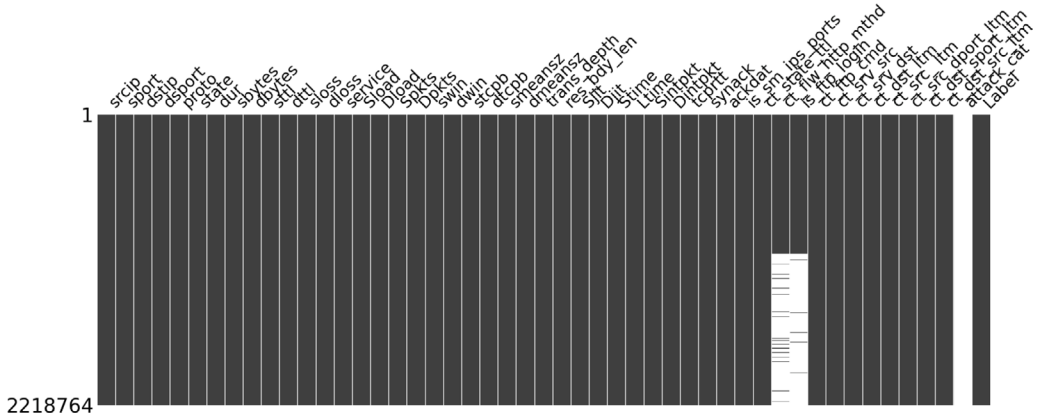

Fig. 4. Missing values in NB-15 dataset.

3.3.4. Features extractions and selection

In feature extraction, we extract 27 common features from the combined dataset for deep learning models as shown in Table 7. In addition to these, the *category* column contains nominal data for multi-class classification, and the *attack* column contains the binary representation that can be used for binary-class classification.

3.3.5. One-hot encoding

After the selection of features, one hot encoding technique is used and performed (object to int) on the combined dataset, and the features are: *saddr*, *sport*, *daddr*, *dport*, and *proto*. Furthermore, for the LSTM model, direct class labels are not suitable, owing to which vector encoding for each class has been performed.

3.3.6. Hyper-parameters tuning and model training

In deep learning experiments, two LSTM layers are used with *output shape = 20*, *parameters = 1760*, *activation = tanh*, *recurrent_activation = sigmoid*, and *return_sequences = True* for the first layer. In the dense layer setting for the LSTM model, the *output shape = 3* is used for three, and *output shape = 5* is used for five-class classification with the *activation = softmax*. Thus the total parameters for three classes are *5103* and five classes are *5145*, as shown in the Table 8. Furthermore, the data-set is split into three parts: 15% for the actual testing, 15% for model validation, and 70% for model training in deep learning.

Table 7
Selected features.

#	Features	Description
1	stime	Record start time
2	saddr	Source IP address
3	sport	Source port number
4	daddr	Destination IP address
5	dport	Destination port number
6	ltime	Record last time
7	dur	Record total duration
8	pkts	Total count of packets in transaction
9	bytes	Totan number of bytes in transaction
10	proto	transaction protocols presents.
11	spkts	Source-to-destination packet count
12	dpkts	Destination-to-source packet count
13	sbytes	Source-to-destination byte count
14	bytes	Destination-to-source byte count
15	TnBPSrcIP	Total No. of bytes per Src IP
16	TnBPDstIP	Total No. of bytes per Dst IP.
17	TnP_PSrcIP	Total No. of packets per Src IP.
18	TnP_PDstIP	Total No. of packets per Dst IP.
19	TnP_PerProto	Total No. of packets per protocol.
20	TnP_Per_Dport	Total No. of packets per dport
21	N_IN_Conn_P_SrcIP	Avg. rate per protocol per Src IP
22	N_IN_Conn_P_DstIP	Avg. rate per protocol per Dst IP
23	AR_P_Proto_P_SrcIP	Several inbound conn. per Src IP
24	AR_P_Proto_P_DstIP	inbound conn. per Dst IP
25	AR_P_Proto_P_Sport	Avg. rate per protocol per sport
26	AR_P_Proto_P_Dport	Avg. rate per protocol per dport
27	Classification_Label	Attack category

Table 8
LSTM best parameters are used for 3-class and 5-class model training.

Layers	Output shape	Parameters	Activation		
			activation	return_sequences	recurrent_activation
LSTM	20	1760	tanh	True	sigmoid
LSTM	20	3280	tanh	–	sigmoid
Dense	3/5*	63/105*	softmax	–	–
Total	–	**5103/5145***	–	–	–

4. Experiments & results

This section details the experiments and outcomes of the proposed intelligent intrusion detection system for the IoT (IIDS-IoT). During the experimentation, we used Windows 10 Pro and Google Colab's free version with an architecture of 64-bit AMD EPYC 7B12 CPU 2-Core 2-Threads. We allocated 12.7 GB RAM and 107.7 GB disk space. While performing learning experimentation we also used scikit-learn, numpy, and pandas libraries in Python version 3.7. The deep learning machine is an Intel Core i7 12700K CPU with 12 cores and 20 threads, 32 GB of RAM, and 512 GB of disk space, with Windows 10 Pro installed. Furthermore, the Jupyter Notebook on Anaconda Navigator 1.9.12 with TensorFlow and Keras back-ends is used with the help of Python version 3.7.

4.1. Machine learning

Model parameters have a significant influence on learning because they guide the model during training on a dataset. For the best parameters, both Randomized Search CV and Grid Search CV have been used. Both methods gave us the same best parameters for SVM and RF. Furthermore, a wide range of parameters have been used and the optimized parameters for each of the supervised machine learning models have been obtained. Through a balanced dataset, shown in Table 3, better training for all classes of the Bot-IoT dataset has been achieved. Both models are cross-validated through k-fold, stratified k-fold, and stratified shuffle split with 10 folds, and the model execution time with respect to folds is shown in Table 9. In experiments, it was noted that the accuracy for SVM binary classification is 91.02%, almost the same for K-Fold and Stratified K-Fold. But significant improvement in accuracy has been seen when using Stratified Shuffle Split with 10 folds. Through parameter tuning, the reported accuracy is 99.60% and the execution time is reduced by 40%. The accuracy of RF binary classification through K-Fold and Stratified K-Fold is 98.30% and 98.32%, and through Stratified Shuffle Split 98.33% accuracy has been achieved and a more than 50% reduction in model training time through parameter tuning is also achieved.

In Figs. 5 and 6, we reported the results of the actual testing of binary classification models through Stratified Shuffle Split (Cross Validation) after removing the over-fitting and effectively selecting the features. In binary classification, the false negative (FN) for

Table 9
Machine learning model execution time for binary and multi-class classification.

Model	Classification type	CV folds	CV methods with execution time (HH:MM:SS)		
			K-fold	Stratified_K-fold	Stratified_ShuffleSplit
RF	Binary	10	00:00:46	00:00:46	00:00:54
RF	Multi	10	00:00:44	00:00:46	00:00:55
SVM	Binary	10	00:02:48	00:02:45	00:03:05
SVM	Multi	10	00:40:23	01:58:33	00:52:57

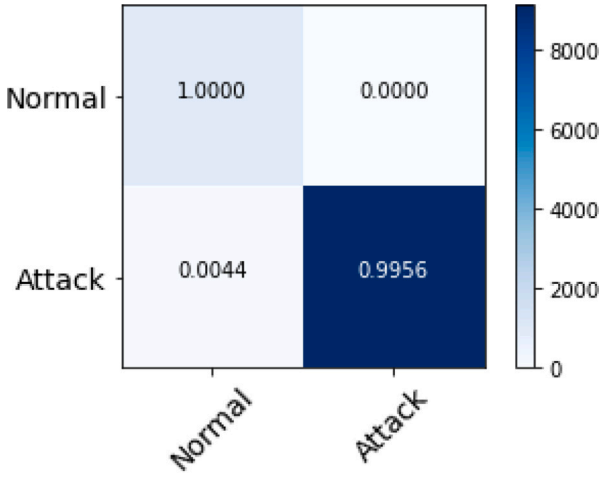

Fig. 5. SVM binary classification with 99.60% accuracy.

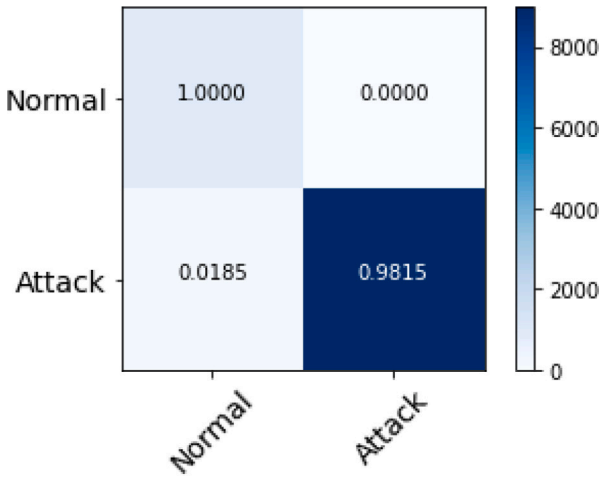

Fig. 6. RF binary classification with 98.33% accuracy.

the attack packets is only 0.44%, and all normal packets are classified by looking at the true positive (TP) shown in Fig. 5, where a total of 99.60% accuracy is achieved by Support Vector Machines (SVM). By looking at the false negative (FN) shown in Fig. 6, we found that the misclassification rate of attack packets in Random Forest (RF) is 1.85%, giving a total of 98.33% accuracy overall, which is a little lower than SVM. As a result, in binary classification, SVM outperforms RF significantly.

The same balanced dataset is used for multi-class machine learning models and cross-validated through k-fold, stratified k-fold, and stratified shuffle split with 10 folds. In experiments, we found the accuracy for SVM multi-classification was 96.99% for K-Fold and 96.50% for Stratified K-Fold. But there is a lot of improvement in accuracy when we use Stratified Shuffle Split with 10 folds. Through parameter tuning, the reported accuracy is 98.07% and the execution time is reduced by 30%. The accuracy of RF multi-classification through K-Fold is 90.64% and Stratified K-Fold is 92.52% and through Stratified Shuffle Split 98.32% accuracy has been achieved and a more than 80% reduction in model training time through parameter tuning has been noted.

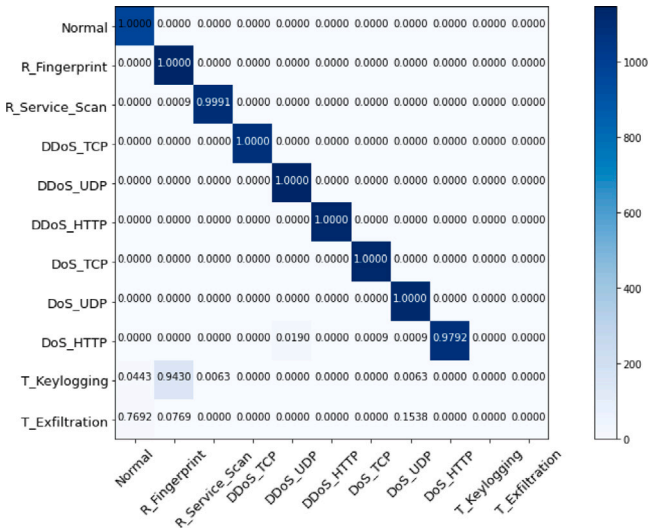

Fig. 7. SVM multi-class classification with 98.07% accuracy.

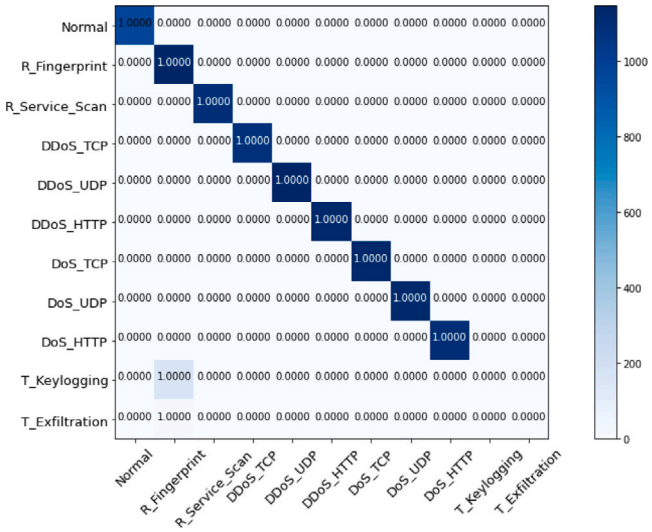

Fig. 8. RF multi-class classification with 98.31% accuracy.

In Figs. 7 and 8, we reported the results of the actual testing of multi-classification, 11 classes are present for prediction, where the normal traffic belongs to the normal class and all malicious packets are in other classes. The false negative (FN) for Service_Scan is 0.09%, DoS_HTTP is only 2.08%, but Keylogging and Exfiltration packets are misclassified due to limited packets in training data, and all other classes are classified by looking at the true positive (TP) shown in Fig. 7, where a total of 98.07% accuracy is achieved by Support Vector Machines (SVM). By looking at the false negative (FN) shown in Fig. 8, we found that Keylogging and Exfiltration are misclassified in Random Forest (RF), and all other classes are classified, which gives it a total of 98.31% accuracy overall, which is a little higher than SVM. As a result, in multi-classification, RF performs better than SVM.

4.2. Deep learning

Analysis shows that 99.93% of abnormal traffic in Bot-IoT consists of UDP and TCP records. By merging normal records from NB-15 and balancing the dataset shown in the Table 5, the focus is on the records for five class classifications and their utilization for deep learning experiments. Furthermore, more than 97.5% of the abnormal traffic in Bot-IoT consists of DoS and DDoS records, so we created an effective training dataset by combining Bot-IoT with normal records from NB-15 and balancing the classes shown in the Table 4 for three class classifications to train the deep learning model.

Table 10
Deep learning (LSTM) model execution time for 3 class and 5 class classification.

Model	Classification type	Epochs	LSTM avg. execution time (HH:MM:SS)		
			Batch size	System Ram (GBs)	Execution time
LSTM	3-class multi	10	3092	32	01:13:33
LSTM	3-class multi	16	3092	32	01:47:13
LSTM	5-class multi	10	3092	32	01:58:12
LSTM	5-class multi	16	3092	32	03:18:09

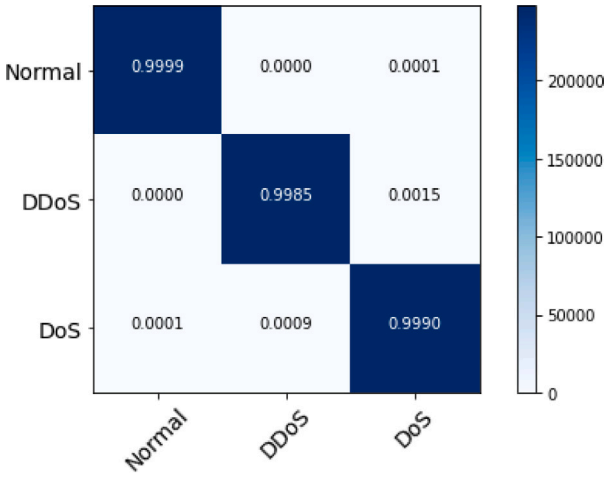

Fig. 9. LSTM Three-class classification iteration-1 with 10 epochs and 99.92% accuracy.

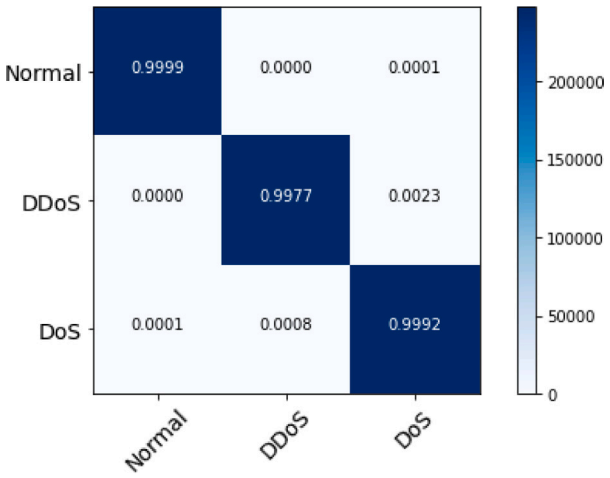

Fig. 10. LSTM three-class classification iteration-2 with 10 epochs and 99.89% accuracy.

The LSTM model is used for three class and five class classifications, and the best parameters for the deep learning model are shown in the Table 8. Furthermore, several iterations are performed on three class and five class classifications to validate our model accuracy and monitor the training loss and validation loss. The epochs, batch size of data, system ram during the execution of experiments, and average execution time are shown in Table 10. All experiment iterations on deep learning are shown in Figs. 21–24.

In the experiments, three iterations are performed to validate three class LSTM model accuracy, as shown in Figs. 9–11. By looking at the 1st iteration in Fig. 9, the TP for the Normal class is 99.99%, so the FN is only 0.01%. So there are only a few Normal records that are misclassified as DoS. The TP for DDoS is 99.85%, and a few packets of DDoS are misclassified as DoS, so the TN is 0.15%. The last rows of the confusion matrix represent the DoS class, where the TP for DoS is 99.90% and 0.09% of DoS records are misclassified as DDoS and 0.01% are misclassified as Normal, so the overall accuracy of iteration-1 is 99.92%.

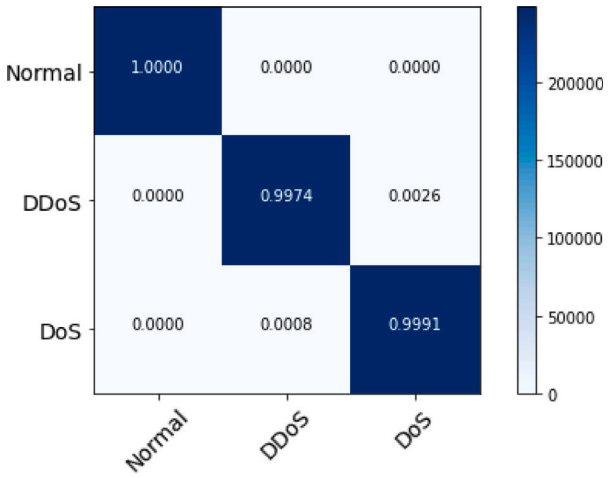

Fig. 11. LSTM three-class classification iteration-3 with 10 epochs and 99.88% accuracy.

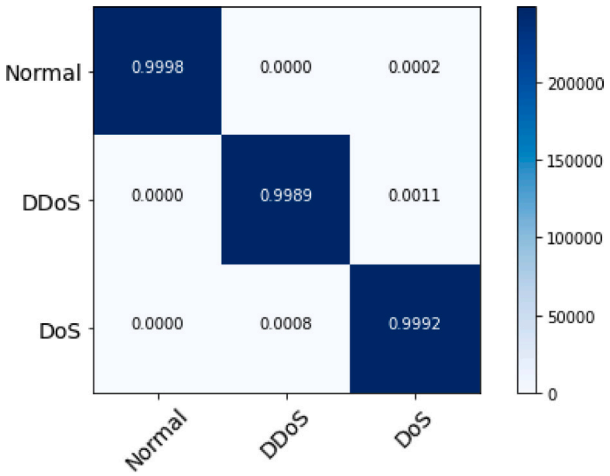

Fig. 12. LSTM three-class classification iteration-1 with 16 epochs and 99.93% accuracy.

Iteration-2's experiment results are shown in Fig. 10, the TP for the Normal class is 99.99% and the FN is only 0.01%. So there are only a few Normal records that are misclassified as DoS. The TP for DDoS is 99.77%, and a few packets of DDoS are misclassified as DoS so the TN is 0.23%. The last rows of the confusion matrix represent the DoS class, where the TP for DoS is 99.92% and 0.08% of DoS records are misclassified as DDoS and 0.01% are misclassified as Normal, so the overall accuracy of this iteration is 99.89%.

The results of the third experiment iteration are shown in Fig. 11 shows iteration-3. In the experiment, there is no misclassification for Normal records. The TP for DDoS is 99.74%, and a few packets of DDoS are misclassified as DoS so the TN is 0.26%. The last rows of the confusion matrix represent the DoS class, where the TP for DoS is 99.91% and 0.08% of DoS records are misclassified as DDoS and 0.01% are misclassified as Normal, so the overall accuracy of this iteration is 99.88%.

In the experiments, three iterations are performed to validate the five-class LSTM model's accuracy, as shown in Figs. 12–14. By looking at the 1st iteration in Fig. 12, the TP for the Normal class is 99.98%, so the FN is only 0.02%. So there are only a few Normal records that are misclassified as DoS. The TP for DDoS is 99.89%, and a few packets of DDoS are misclassified as DoS, so the TN is 0.11%. The last rows of the confusion matrix represent the DoS class, where the TP for DoS is 99.92% and 0.08% of DoS records are misclassified as DDoS, so the overall accuracy of iteration-1 is 99.93%.

Results of the second experiment iteration are shown in Fig. 13 shows. In the experiment, there is no misclassification for Normal records. The TP for DDoS is 99.88%, and a few packets of DDoS are misclassified as DoS, so the TN is 0.12%. The last rows of the confusion matrix represent the DoS class, where the TP for DoS is 99.47% and 0.53% of DoS records are misclassified as DDoS, so the overall accuracy of this iteration is 99.78%.

Iteration-3's experiment results are shown in Fig. 14. In the experiment, there is no misclassification for Normal records. The TP for DDoS is 99.85%, and a few packets of DDoS are misclassified as DoS so the TN is 0.15%. The last rows of the confusion

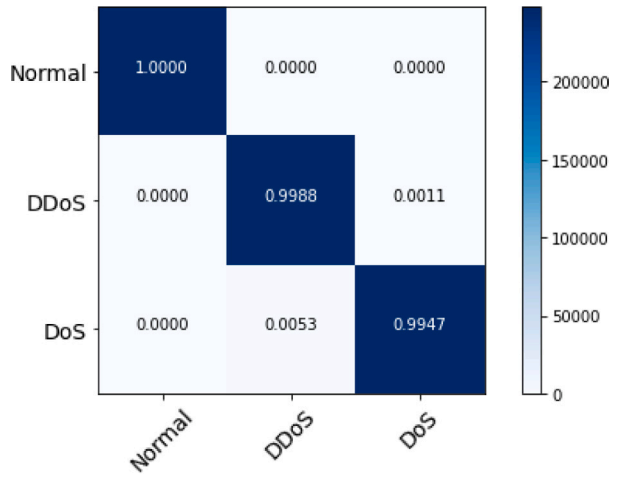

Fig. 13. LSTM three-class classification iteration-2 with 16 epochs and 99.78% accuracy.

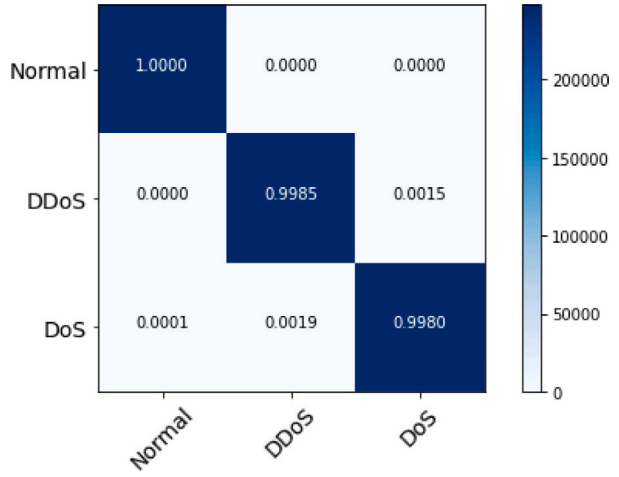

Fig. 14. LSTM three-class classification iteration-3 with 16 epochs and 99.80% accuracy.

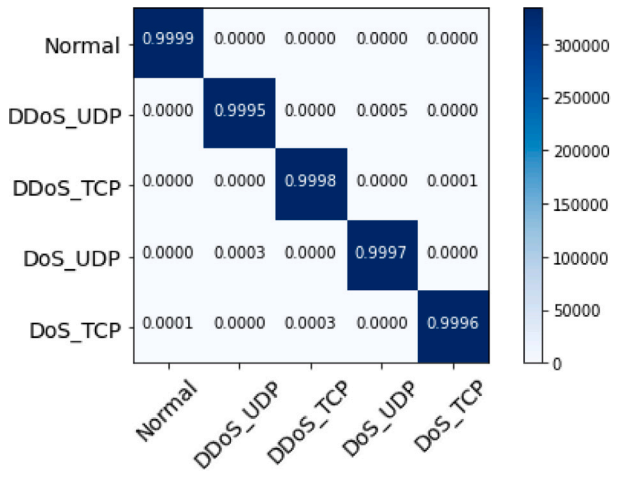

Fig. 15. LSTM five-class classification iteration-1 with 10 epochs and 99.97% accuracy.

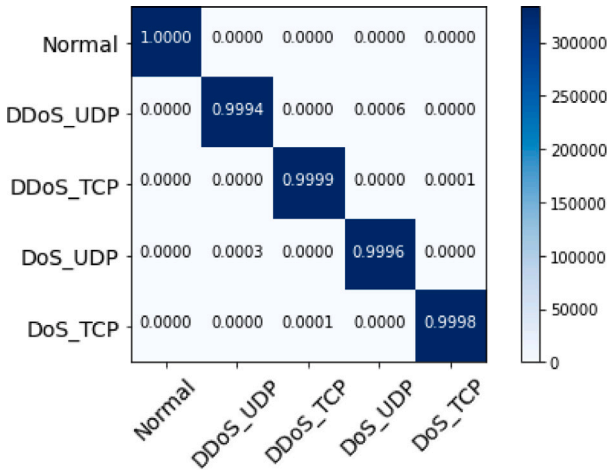

Fig. 16. LSTM five-class classification iteration-2 with 10 epochs and 99.98% Accuracy.

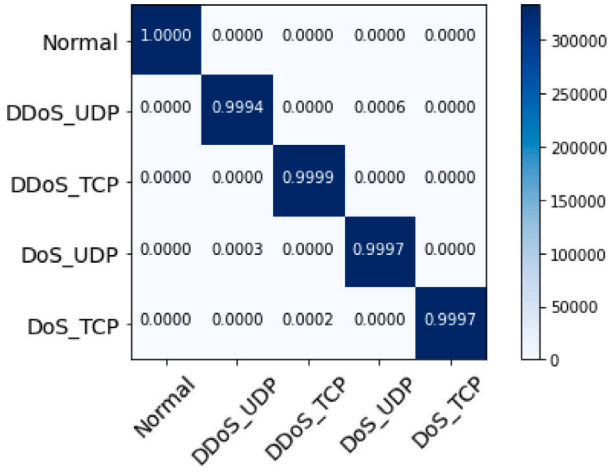

Fig. 17. LSTM five-class classification iteration-3 with 10 epochs and 99.97% accuracy.

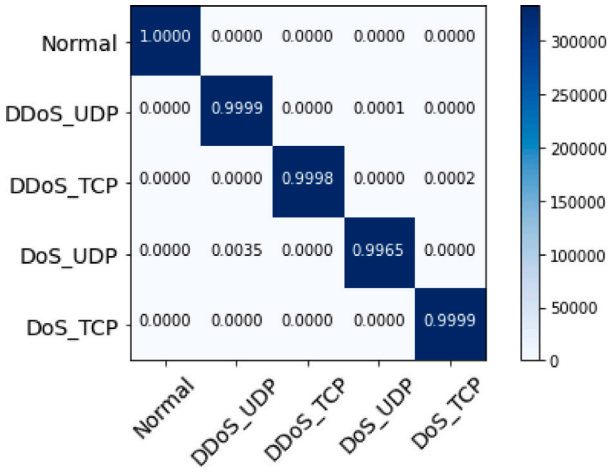

Fig. 18. LSTM three-class classification iteration-1 with 16 epochs and 99.92% accuracy.

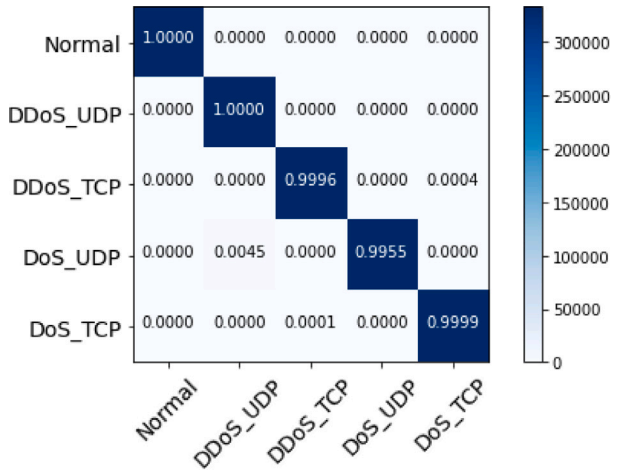

Fig. 19. LSTM three-class classification iteration-2 with 16 epochs and 99.90% accuracy.

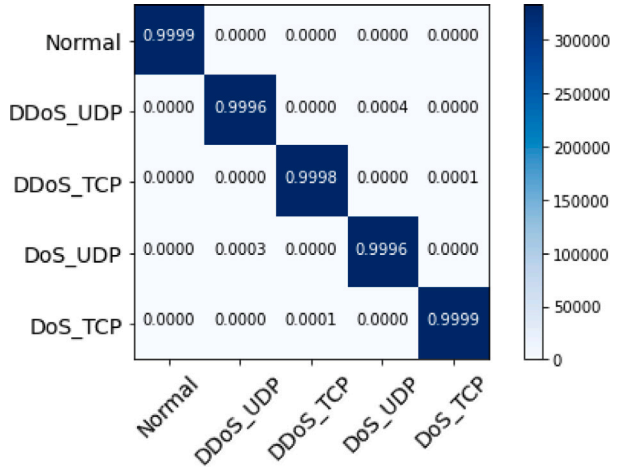

Fig. 20. LSTM three-class classification iteration-3 with 16 epochs and 99.98% accuracy.

matrix represent the DoS class, where the TP for DoS is 99.80% and 0.19% of DoS records are misclassified as DDoS and 0.01% are misclassified as Normal, so the overall accuracy of this iteration is 99.80%.

In the experiments, again three iterations are performed to validate the five-class LSTM model's accuracy, as shown in Figs. 15–17. By looking at the 1st iteration in Fig. 15, the TP for the Normal class is 99.99%, so the FN is only 0.01%. The TP for DDoS_UDP is 99.95%, and a few packets of DDoS_UDP are misclassified as DoS_UDP, so the TN is 0.05%. The 3rd row of the confusion matrix represents the DDoS_TCP, where the TP for DDoS_TCP is 99.98%, meaning 0.02% of DDoS_TCP records are misclassified. The TP for DoS_UDP is 99.97%, and a few packets of DoS_UDP are misclassified as DDoS_UDP, so the TN is 0.03%. The last row represents the DoS_TCP class, where the TP for DoS_TCP is 99.96%, and 0.03% of DoS_TCP records are misclassified as DDoS_TCP and 0.01% are misclassified as Normal hence the overall accuracy of iteration-1 is 99.97%.

Fig. 16 shows iteration-2. In the experiment, there is no misclassification for Normal records. The TP for DDoS_UDP is 99.94%, and a few packets of DDoS_UDP are misclassified as DoS_UDP, so the TN is 0.06%. The 3rd row of the confusion matrix represents the DDoS_TCP, where the TP for DDoS_TCP is 99.99%, meaning 0.01% of DDoS_TCP records are misclassified as DoS_TCP. The TP for DoS_UDP is 99.96%, and a few packets of DoS_UDP are misclassified as DDoS_UDP, so the TN is 0.04%. The last row represents the DoS_TCP class, where the TP for DoS_TCP is 99.98%, and 0.02% of DoS_TCP records are misclassified, so the overall accuracy for this iteration is 99.98%.

Fig. 17 shows iteration-3. In the experiment, there is no misclassification for Normal records. The TP for DDoS_UDP is 99.94%, and a few packets of DDoS_UDP are misclassified as DoS_UDP, so the TN is 0.06%. The 3rd row represents DDoS_TCP, where the TP for DDoS_TCP is 99.99%, meaning 0.01% of DDoS_TCP records are misclassified. The TP for DoS_UDP is 99.97%, and a few packets of DoS_UDP are misclassified as DDoS_UDP, so the TN is 0.03%. The last row represents the DoS_TCP class, where the TP for DoS_TCP is 99.97%, and 0.03% of DoS_TCP records are misclassified, so the overall accuracy for this iteration is 99.97%.

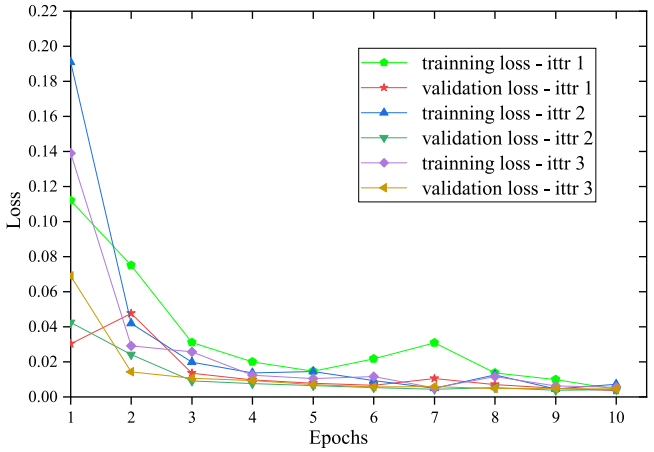

Fig. 21. Three-class, three iteration of LSTM training and validation loss with 10 epochs.

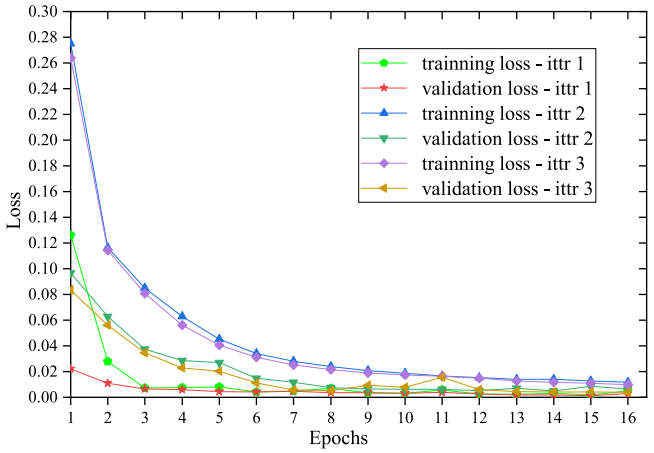

Fig. 22. Three-class, three iteration of LSTM training and validation loss with 16 epochs.

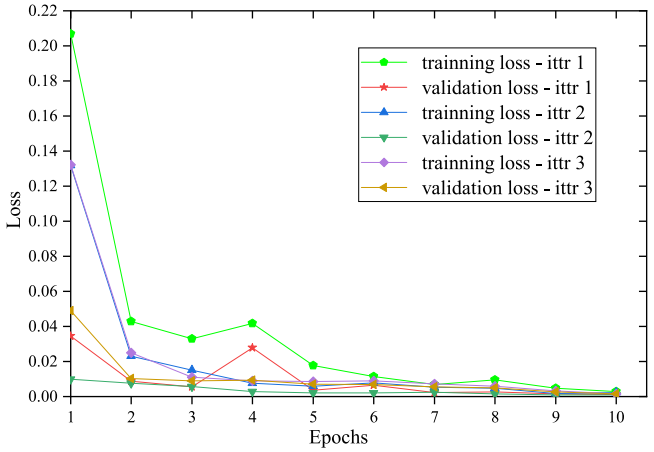

Fig. 23. Five-class, three iteration of LSTM training loss and validation loss with 10 epochs.

In the experiments, we performed three iterations to validate our five-class LSTM model's accuracy, shown in Figs. 18–20. By looking at the 1st iteration in Fig. 18, there is no misclassification for Normal records. The TP for DDoS_UDP is 99.99%, and a

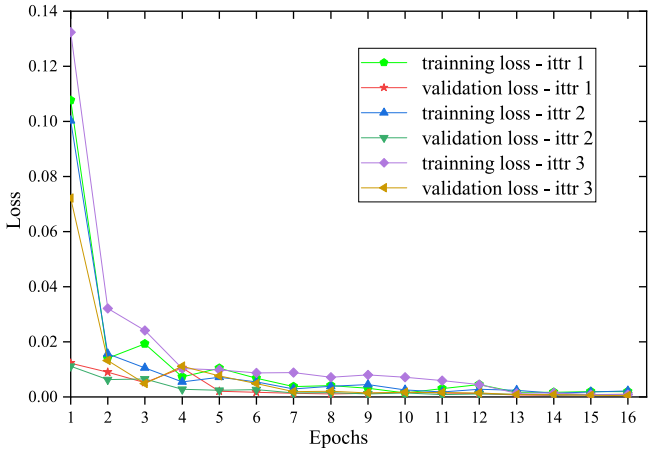

Fig. 24. Five-class, three iteration of LSTM training loss and validation loss with 16 epochs.

Table 11
Comparison with state-of-the-art works.

Papers with years	Dataset for research	Algo. used	Dataset used for experiment	Imbalanced dataset	No. of features	Classified classes	Accuracy in (%)
[63], (2020)	Bot-IoT NB-15	BiLSTM	Subset	Yes	Full	16	99.41 98.91
[79], (2020)	Bot-IoT	RF SVM	Subset	Yes	5 best	8	99.98 97.80
[67], (2021)	Bot-IoT	RF	DoS, DDoS Packets	Yes	Full	3	94.00
[66], (2021)	Bot-IoT	RF	5%	Yes	37	Binary	97.18
[80], (2021)	Bot-IoT NB-15	LSTM	96% 87%	No	26	3	96.30
[78], (2022)	NB-15	MLP	10%	Yes	23	6	84.24
[57], (2023)	RT-IoT2022	LSTM	Subset	Yes	Full	3	98.00
[26], (2024)	NB-15	RF	Full	Yes	Full	9	94.21
[47], (2024)	NB-15	LSTM	Subset	Yes	Full	9	96.98
[39], (2024)	Bot-IoT	LSTM	Subset	Yes	Full	11	98.01
Purposed (IIDS-IoT)	**Bot-IoT** **NB-15**	**SVM** **RF** **LSTM**	**100%** **87%**	No	**26** **27**	**Binary** **3** **5** **11**	**99.60** **98.07** **98.33** **98.31** **99.89** **99.97**

few packets of DDoS_UDP are misclassified as DoS_UDP, so the TN is 0.01%. The third row represents DDoS_TCP, where the TP for DDoS_TCP is 99.98%, meaning 0.02% of DDoS_TCP records are misclassified as DoS_TCP. The TP for DoS_UDP is 99.65%, and a few packets of DoS_UDP are misclassified as DDoS_UDP, so the TN is 0.35%. The last row represents the DoS_TCP class, where the TP for DoS_TCP is 99.9%, and 0.01% of DoS_TCP records are misclassified, so the overall accuracy of iteration-1 is 99.92%.

Fig. 19 shows iteration-2. In the experiment, there is no misclassification for Normal and DDoS_UDP records. The third row represents DDoS_TCP, where the TP for DDoS_TCP is 99.96%, meaning 0.04% of DDoS_TCP records are misclassified as DoS_TCP. The TP for DoS_UDP is 99.55%, and a few packets of DoS_UDP are misclassified as DDoS_UDP, so the TN is 0.45%. The last row represents the DoS_TCP class, where the TP for DoS_TCP is 99.99%, and 0.01% of DoS_TCP records are misclassified as DDoS_TCP, so the overall accuracy for this iteration is 99.90%.

Fig. 20 shows iteration-3. In the experiment, the TP for the Normal class is 99.99%, so the FN is only 0.01%. The TP for DDoS_UDP is 99.96%, and a few packets of DDoS_UDP are misclassified as DoS_UDP, so the TN is 0.04%. The third-row represents DDoS_TCP, where the TP for DDoS_TCP is 99.98%, meaning 0.02% of DDoS_TCP records are misclassified. The TP for DoS_UDP is 99.96%, and a few packets of DoS_UDP are misclassified, so the TN is 0.04%. The last row represents the DoS_TCP class, where the TP for DoS_TCP is 99.99%, and 0.01% of DoS_TCP records are misclassified as DDoS_TCP, so the overall accuracy for this iteration is 99.98%.

The conducted experiments and results show equally trained classes, and the misclassification rate is very low. We validate the model through multiple iterations and through the training and validation loss graphs shown in Figs. 21–24. The results show that the purposed solution (IIDS-IoT), with a balanced dataset and effective feature selections, outperforms other state-of-the-art methods.

We utilized 100% Bot-IoT dataset with 26 features and 87% NB-15 dataset with 27 features. The purposed solution is compared with other state-of-the-art works in Table 11 and it is evident that these findings outperform other state-of-the-art studies, as shown in (Table 11). The proposed model achieves higher accuracies for both binary and multi-class classifications.

5. Conclusion

In this research, two datasets are integrated together, i.e., BoT-IoT and NB-15, with appropriate feature selection. For training purposes, machine learning and deep learning models, including RF, SVM, and LSTM, are used. Different classes within the dataset have been balanced along with cross-validation to train the algorithms better and overcome the over-fitting problem. The techniques used in this research have a significant impact on enhancing the accuracy of models. For our two different intrusion detection systems, the accuracy for machine learning is 99.60% for binary and 98.32% for multi-class, while LSTM with combined datasets brings an accuracy of 99.89% for three-class classification and 99.97% for five-class classification. We found out that stratified shuffle split gives us better accuracy as well as less model execution and training time in both binary class and multi-class classification.

CRediT authorship contribution statement

Muhammad Luqman: Investigation, Software, Writing – original draft. **Muhammad Zeeshan:** Conceptualization, Methodology, Supervision, Writing – review & editing. **Qaiser Riaz:** Validation. **Mehdi Hussain:** Data curation, Formal analysis. **Hasan Tahir:** Writing – review & editing. **Noman Mazhar:** Writing – review & editing. **Muhammad Saffeer Khan:** Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] P. Laskov, P. Düssel, C. Schäfer, K. Rieck, Learning intrusion detection: supervised or unsupervised? in: Image Analysis and Processing–ICIAP 2005: 13th International Conference, Cagliari, Italy, September 6-8, 2005. Proceedings 13, Springer, 2005, pp. 50–57.
[2] B. Kaur, S. Dadkhah, F. Shoeleh, E.C.P. Neto, P. Xiong, S. Iqbal, P. Lamontagne, S. Ray, A.A. Ghorbani, Internet of things (IoT) security dataset evolution: Challenges and future directions, Internet Things (2023) 100780.
[3] A.R. Gad, A.A. Nashat, T.M. Barkat, Intrusion detection system using machine learning for vehicular ad hoc networks based on ToN-IoT dataset, IEEE Access 9 (2021) 142206–142217.
[4] N. Koroniotis, N. Moustafa, E. Sitnikova, B. Turnbull, Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset, Future Gener. Comput. Syst. 100 (2019) 779–796.
[5] N. Moustafa, 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, MilCIS, IEEE, 2015, pp. 1–6.
[6] U. Oktay, O. Sahingoz, et al., Attack types and intrusion detection systems in cloud computing, in: Proceedings of the 6th International Information Security & Cryptology Conference, 2013, pp. 71–76.
[7] U.A. Sandhu, S. Haider, S. Naseer, O.U. Ateeb, A survey of intrusion detection & prevention techniques, in: 2011 International Conference on Information Communication and Management, IPCSIT, Vol. 16, 2011, pp. 66–71.
[8] V. Kumar, O.P. Sangwan, Signature based intrusion detection system using SNORT, Int. J. Comput. Appl. Inf. Technol. 1 (3) (2012) 35–41.
[9] S. Jin, J.-G. Chung, Y. Xu, Signature-based intrusion detection system (IDS) for in-vehicle CAN bus network, in: 2021 IEEE International Symposium on Circuits and Systems, ISCAS, IEEE, 2021, pp. 1–5.
[10] A. Rai, Optimizing a new intrusion detection system using ensemble methods and deep neural network, in: 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), IEEE, 2020, pp. 527–532.
[11] W. He, H. Li, J. Li, Ensemble feature selection for improving intrusion detection classification accuracy, in: Proceedings of the 2019 International Conference on Artificial Intelligence and Computer Science, 2019, pp. 28–33.
[12] K.A. Taher, B.M.Y. Jisan, M.M. Rahman, Network intrusion detection using supervised machine learning technique with feature selection, in: 2019 International Conference on Robotics, Electrical and Signal Processing Techniques, ICREST, IEEE, 2019, pp. 643–646.
[13] E. Hodo, X. Bellekens, A. Hamilton, P.-L. Dubouilh, E. Iorkyase, C. Tachtatzis, R. Atkinson, Threat analysis of IoT networks using artificial neural network intrusion detection system, in: 2016 International Symposium on Networks, Computers and Communications, ISNCC, IEEE, 2016, pp. 1–6.
[14] S. Moghanian, F.B. Saravi, G. Javidi, E.O. Sheybani, GOAMLP: Network intrusion detection with multilayer perceptron and grasshopper optimization algorithm, IEEE Access 8 (2020) 215202–215213.
[15] M. Ahsan, K.E. Nygard, Convolutional neural networks with LSTM for intrusion detection., in: CATA, Vol. 69, 2020, pp. 69–79.
[16] P. Hadem, D.K. Saikia, S. Moulik, An SDN-based intrusion detection system using SVM with selective logging for IP traceback, Comput. Netw. 191 (2021) 108015.
[17] P. Nagar, H.K. Menaria, M. Tiwari, Novel approach of intrusion detection classification deeplearning using SVM, in: First International Conference on Sustainable Technologies for Computational Intelligence, Springer, 2020, pp. 365–381.
[18] B.M. Khammas, Ransomware detection using random forest technique, ICT Express 6 (4) (2020) 325–331.
[19] B.A. Tama, M. Comuzzi, K.-H. Rhee, TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system, IEEE Access 7 (2019) 94497–94507.
[20] A. Vikram, et al., Anomaly detection in network traffic using unsupervised machine learning approach, in: 2020 5th International Conference on Communication and Electronics Systems, ICCES, IEEE, 2020, pp. 476–479.
[21] M. Shafiq, Z. Tian, Y. Sun, X. Du, M. Guizani, Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city, Future Gener. Comput. Syst. 107 (2020) 433–442.
[22] S. Zhu, X. Xu, H. Gao, F. Xiao, CMTSNN: A deep learning model for multi-classification of abnormal and encrypted traffic of internet of things, IEEE Internet Things J. (2023).

[23] S. Ahmed, Z.A. Khan, S.M. Mohsin, S. Latif, S. Aslam, H. Mujlid, M. Adil, Z. Najam, Effective and efficient ddos attack detection using deep learning algorithm, multi-layer perceptron, Future Internet 15 (2) (2023) 76.
[24] Y.-C. Wang, Y.-C. Houng, H.-X. Chen, S.-M. Tseng, Network anomaly intrusion detection based on deep learning approach, Sensors 23 (4) (2023) 2171.
[25] J. Romeiras Amado, F. Pereira, D. Pissarra, S. Signorello, M. Correia, F. Ramos, Peregrine: ML-based malicious traffic detection for terabit networks, 2024, arXiv e-prints, arXiv–2403.
[26] S.U. Jafri, S. Rao, V. Shrivastav, M. Tawarmalani, Leo: Online {ML-based} traffic classification at {Multi-Terabit} line rate, in: 21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24), 2024, pp. 1573–1591.
[27] A. Thakkar, R. Lohiya, Attack classification of imbalanced intrusion data for IoT network using ensemble learning-based deep neural network, IEEE Internet Things J. (2023).
[28] R. Anushiya, V. Lavanya, A new deep-learning with swarm based feature selection for intelligent intrusion detection for the internet of things, Meas.: Sens. 26 (2023) 100700.
[29] C. Kemp, C. Calvert, T.M. Khoshgoftaar, J.L. Leevy, An approach to application-layer DoS detection, J. Big Data 10 (1) (2023) 22.
[30] T.M. Ghazal, N.A. Al-Dmour, R.A. Said, A. Omidvar, U.Y. Khan, T.R. Soomro, H.M. Alzoubi, M. Alshurideh, T.M. Abdellatif, A. Moubayed, et al., Ddos intrusion detection with ensemble stream mining for IoT smart sensing devices, in: The Effect of Information Technology on Business and Marketing Intelligence Systems, Springer, 2023, pp. 1987–2012.
[31] P. Chauhan, M. Atulkar, An efficient centralized ddos attack detection approach for software defined internet of things, J. Supercomput. (2023) 1–37.
[32] Z. Diao, G. Xie, X. Wang, R. Ren, X. Meng, G. Zhang, K. Xie, M. Qiao, EC-GCN: A encrypted traffic classification framework based on multi-scale graph convolution networks, Comput. Netw. 224 (2023) 109614.
[33] G. Prajapati, P. Singh, Anomaly based network intrusion detection system for IoT, in: Proceedings of International Conference on Data Science and Applications: ICDSA 2022, Volume 2, Springer, 2023, pp. 693–706.
[34] Y. Liu, Y. Zhou, K. Yang, X. Wang, Unsupervised deep learning for IoT time series, IEEE Internet Things J. (2023).
[35] J. Vitorino, I. Praça, E. Maia, Towards adversarial realism and robust learning for IoT intrusion detection and classification, Ann. Telecommun. (2023) 1–12.
[36] A. Awajan, A novel deep learning-based intrusion detection system for IoT networks, Computers 12 (2) (2023) 34.
[37] C.S. Kalutharage, X. Liu, C. Chrysoulas, N. Pitropakis, P. Papadopoulos, Explainable AI-based DDOS attack identification method for IoT networks, Computers 12 (2) (2023) 32.
[38] M.A. Bouke, A. Abdullah, An empirical assessment of ML models for 5G network intrusion detection: A data leakage-free approach, e-Prime-Adv. Electr. Eng. Electron. Energy (2024) 100590.
[39] M. Nanjappan, K. Pradeep, G. Natesan, A. Samydurai, G. Premalatha, Deeplg SecNet: utilizing deep LSTM and GRU with secure network for enhanced intrusion detection in IoT environments, Cluster Comput. (2024) 1–13.
[40] M.A. Rahman, A.T. Asyhari, L. Leong, G. Satrya, M.H. Tao, M. Zolkipli, Scalable machine learning-based intrusion detection system for IoT-enabled smart cities, Sustainable Cities Soc. 61 (2020) 102324.
[41] R.A. Ramadan, Efficient intrusion detection algorithms for smart cities-based wireless sensing technologies, J. Sens. Actuator Netw. 9 (3) (2020) 39.
[42] V. Morfino, S. Rampone, Towards near-real-time intrusion detection for IoT devices using supervised learning and apache spark, Electronics 9 (3) (2020) 444.
[43] C. Ioannou, V. Vassiliou, Experimentation with local intrusion detection in IoT networks using supervised learning, in: 2020 16th International Conference on Distributed Computing in Sensor Systems, DCOSS, IEEE, 2020, pp. 423–428.
[44] I. Alrashdi, A. Alqazzaz, E. Aloufi, R. Alharthi, M. Zohdy, H. Ming, Ad-iot: Anomaly detection of iot cyberattacks in smart city using machine learning, in: 2019 IEEE 9th Annual Computing and Communication Workshop and Conference, CCWC, IEEE, 2019, pp. 0305–0310.
[45] R.T. Wiyono, N.D.W. Cahyani, Performance analysis of decision tree c4. 5 as a classification technique to conduct network forensics for botnet activities in internet of things, in: 2020 International Conference on Data Science and Its Applications (ICoDSA), IEEE, 2020, pp. 1–5.
[46] K.S. Kiran, R.K. Devisetty, N.P. Kalyan, K. Mukundini, R. Karthi, Building a intrusion detection system for iot environment using machine learning techniques, Procedia Comput. Sci. 171 (2020) 2372–2379.
[47] M. Ali, M. Shahroz, M.F. Mushtaq, S. Alfarhood, M. Safran, I. Ashraf, Hybrid machine learning model for efficient botnet attack detection in IoT environment, IEEE Access (2024).
[48] R. Gopi, V. Sathiyamoorthi, S. Selvakumar, R. Manikandan, P. Chatterjee, N. Jhanjhi, A.K. Luhach, Enhanced method of ANN based model for detection of ddos attacks on multimedia internet of things, Multimedia Tools Appl. (2021) 1–19.
[49] P. Kumar, R. Kumar, G.P. Gupta, R. Tripathi, A distributed framework for detecting ddos attacks in smart contract-based blockchain-IoT systems by leveraging fog computing, Trans. Emerg. Telecommun. Technol. 32 (6) (2021) e4112.
[50] M. Almiani, A. AbuGhazleh, A. Al-Rahayfeh, S. Atiewi, A. Razaque, Deep recurrent neural network for IoT intrusion detection system, Simul. Model. Pract. Theory 101 (2020) 102031.
[51] S. Hanif, T. Ilyas, M. Zeeshan, Intrusion detection in IoT using artificial neural networks on UNSW-15 dataset, in: 2019 IEEE 16th International Conference on Smart Cities: Improving Quality of Life using ICT & IoT and AI (HONET-ICT), IEEE, 2019, pp. 152–156.
[52] M. Ge, N.F. Syed, X. Fu, Z. Baig, A. Robles-Kelly, Towards a deep learning-driven intrusion detection approach for internet of things, Comput. Netw. 186 (2021) 107784.
[53] J. Arshad, M.A. Azad, M.M. Abdeltaif, K. Salah, An intrusion detection framework for energy constrained IoT devices, Mech. Syst. Signal Process. 136 (2020) 106436.
[54] X. Kan, Y. Fan, Z. Fang, L. Cao, N.N. Xiong, D. Yang, X. Li, A novel IoT network intrusion detection approach based on adaptive particle swarm optimization convolutional neural network, Inform. Sci. 568 (2021) 147–162.
[55] T. Su, H. Sun, J. Zhu, S. Wang, Y. Li, BAT: Deep learning methods on network intrusion detection using NSL-kdd dataset, IEEE Access 8 (2020) 29575–29585.
[56] M. Amine Ferrag, O. Friha, B. Kantarci, N. Tihanyi, L. Cordeiro, M. Debbah, D. Hamouda, M. Al-Hawawreh, K.-K.R. Choo, Edge learning for 6G-enabled internet of things: A comprehensive survey of vulnerabilities, datasets, and defenses, 2023, arXiv e-prints, arXiv–2306.
[57] B. Sharmila, R. Nagapadma, Quantized autoencoder (QAE) intrusion detection system for anomaly detection in resource-constrained IoT devices using RT-IoT2022 dataset, Cybersecurity 6 (1) (2023) 41.
[58] M. Farouk, R.H. Sakr, N. Hikal, Identifying the most accurate machine learning classification technique to detect network threats, Neural Comput. Appl. 36 (16) (2024) 8977–8994.
[59] M. Shafi, A.H. Lashkari, V. Rodriguez, R. Nevo, Toward generating a new cloud-based distributed denial of service (DDoS) dataset and cloud intrusion traffic characterization, Information 15 (4) (2024) 195.
[60] M. Ge, X. Fu, N. Syed, Z. Baig, G. Teo, A. Robles-Kelly, Deep learning-based intrusion detection for IoT networks, in: 2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing, PRDC, IEEE, 2019, pp. 256–25609.
[61] M. Shafiq, Z. Tian, A.K. Bashir, X. Du, M. Guizani, IoT malicious traffic identification using wrapper-based feature selection mechanisms, Comput. Secur. 94 (2020) 101863.
[62] A. Churcher, R. Ullah, J. Ahmad, F. Masood, M. Gogate, F. Alqahtani, B. Nour, W.J. Buchanan, et al., An experimental analysis of attack classification using machine learning in IoT networks, Sensors 21 (2) (2021) 446.

[63] O. Alkadi, N. Moustafa, B. Turnbull, K.-K.R. Choo, A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks, IEEE Internet Things J. 8 (12) (2020) 9463–9472.
[64] G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, A. Pescapé, A hierarchical hybrid intrusion detection approach in IoT scenarios, in: GLOBECOM 2020-2020 IEEE Global Communications Conference, IEEE, 2020, pp. 1–7.
[65] I. Ullah, Q.H. Mahmoud, Design and development of a deep learning-based model for anomaly detection in IoT networks, IEEE Access 9 (2021) 103906–103926.
[66] J.L. Leevy, J. Hancock, T.M. Khoshgoftaar, J. Peterson, Detecting information theft attacks in the Bot-IoT dataset, in: 2021 20th IEEE International Conference on Machine Learning and Applications, ICMLA, IEEE, 2021, pp. 807–812.
[67] H. Azath, D.B. David, E.C. Blessie, A. Jayapradha, S.S. Rani, et al., BoT-IoT based denial of service detection with deep learning, in: 2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), IEEE, 2021, pp. 221–225.
[68] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, F. Ahmad, Network intrusion detection system: A systematic study of machine learning and deep learning approaches, Trans. Emerg. Telecommun. Technol. 32 (1) (2021) e4150.
[69] S. Moualla, K. Khorzom, A. Jafar, Improving the performance of machine learning-based network intrusion detection systems on the UNSW-NB15 dataset, Comput. Intell. Neurosci. 2021 (2021).
[70] C. Zheng, M. Zang, X. Hong, R. Bensoussane, S. Vargaftik, Y. Ben-Itzhak, N. Zilberman, Automating in-network machine learning, 2022, arXiv preprint arXiv:2205.08824.
[71] A. Aleesa, M. Younis, A.A. Mohammed, N. Sahar, Deep-intrusion detection system with enhanced UNSW-NB15 dataset based on deep learning techniques, Journal of Engineering Science and Technology 16 (1) (2021) 711–727.
[72] V. Kanimozhi, P. Jacob, UNSW-NB15 dataset feature selection and network intrusion detection using deep learning, Int. J. Recent Technol. Eng. 7 (5S2) (2019) 443–446.
[73] L. Zhiqiang, G. Mohi-Ud-Din, L. Bing, L. Jianchao, Z. Ye, L. Zhijun, Modeling network intrusion detection system using feed-forward neural network using unsw-nb15 dataset, in: 2019 IEEE 7th International Conference on Smart Energy Grid Engineering, SEGE, IEEE, 2019, pp. 299–303.
[74] H. Gwon, C. Lee, R. Keum, H. Choi, Network intrusion detection based on lstm and feature embedding, 2019, arXiv preprint arXiv:1911.11552.
[75] T. Pooja, P. Shrinivasacharya, Evaluating neural networks using bi-directional LSTM for network ids (intrusion detection systems) in cyber security, Glob. Transitions Proc. 2 (2) (2021) 448–454.
[76] Y. Yan, L. Qi, J. Wang, Y. Lin, L. Chen, A network intrusion detection method based on stacked autoencoder and LSTM, in: ICC 2020-2020 IEEE International Conference on Communications, ICC, IEEE, 2020, pp. 1–6.
[77] Y.K. Saheed, A.I. Abiodun, S. Misra, M.K. Holone, R. Colomo-Palacios, A machine learning-based intrusion detection for detecting internet of things network attacks, Alex. Eng. J. 61 (12) (2022) 9395–9409.
[78] Y. Yin, J. Jang-Jaccard, W. Xu, A. Singh, J. Zhu, F. Sabrina, J. Kwak, IGRF-RFE: A hybrid feature selection method for MLP-based network intrusion detection on UNSW-NB15 dataset, 2022, arXiv preprint arXiv:2203.16365.
[79] M. Shafiq, Z. Tian, A.K. Bashir, X. Du, M. Guizani, CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine-learning techniques, IEEE Internet Things J. 8 (5) (2020) 3242–3254.
[80] M. Zeeshan, Q. Riaz, M.A. Bilal, M.K. Shahzad, H. Jabeen, S.A. Haider, A. Rahim, Protocol-based deep intrusion detection for DoS and DDoS attacks using UNSW-NB15 and Bot-IoT data-sets, IEEE Access 10 (2021) 2269–2283.