Design and Development of RNN-based Anomaly Detection Model for IoT Networks

Imtiaz Ullah, (Member, IEEE), Qusay H. Mahmoud, (Senior Member, IEEE)
Department of Electrical, Computer and Software Engineering, Ontario Tech University, Oshawa, ON, L1G 0C5 Canada
Corresponding author: Imtiaz Ullah (e-mail: imtiaz.ullah@ontariotechu.net)

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

ABSTRACT Cybersecurity is important today because of the increasing growth of the Internet of Things (IoT), which has resulted in a variety of attacks on computer systems and networks. As the number of various IoT devices and services grows, cyber security will become an increasingly difficult issue to manage. Malicious traffic identification using deep learning techniques has emerged as a key component of network-based intrusion detection systems (IDS). Deep learning methods have been a research focus in network intrusion detection. A recurrent neural network is useful in a wide range of applications. This paper proposes a novel deep learning model for detecting anomalies in IoT networks using recurrent neural networks. The proposed model is implemented in IoT networks utilizing LSTM, BiLSTM, and GRU-based approaches for anomaly detection. A convolutional neural network can analyze input features without losing important information, making them particularly well suited for feature learning. In addition, we propose a hybrid deep learning model based on convolutional and recurrent neural networks. Finally, employing LSTM, BiLSTM, and GRU-based techniques, we propose a lightweight deep learning model for binary classification. The proposed deep learning models are validated using NSLKDD, BoT-IoT, IoT-NI, MQTT, MQTTset, IoT-23, and IoT-DS2 datasets. Our proposed binary and multiclass classification model achieved high accuracy, precision, recall, and F1 score compared to current deep learning implementations.

INDEX TERMS Internet of Things, anomaly detection, recurrent neural network, convolutional neural network, LSTM, BiLSTM, GRU.

I. INTRODUCTION

The fast Internet growth has facilitated the creation of the IoT. A common element contributing to this development is the ease with which IoT devices are available, affordable, and convenient in our everyday lives. Due to the fast advancement of wireless communication technologies, developers have built extremely low-cost IoT nodes that support data collection, data analysis, and wireless transmission [1]. IoT is a network of linked physical devices and sensors that enables information sharing via the Internet. IoT networks have shown particularly efficient in collecting, analyzing, reporting, and predicting information for incorporation into future plans. Many technologies, including protocols, software, and component elements, make up IoT networks. An IoT architecture is a collection of multiple components such as sensors, actuators, protocols, cloud services, and layers of an IoT communication network. The layer is usually made up of many levels that let administrators identify, analyze, and manage the system's consistency.

There is no one agreed-upon design for the IoT, with many designs suggested by various developers. The primary types of architecture are three, four, and five-layer structures. A three-layer structure, comprised of perception, network, and application layers, is the most basic configuration for IoT implementation. The perception layer is equipped with sensors, actuators, and computational hardware to detect and collect information from the environments. The physical layer handles tasks such as setting a frequency, manipulating the signal, encrypting the signal, and transmitting and receiving data. This layer has several issues, including power consumption, security, and compatibility. The network layer connects IoT devices to other smart objects, network equipment, and services. This layer receives data from the perception layer and passes the data to the application for analytics and smart services. The network layer needs to deal
with reliability, network capacity, energy usage, and security. The application layer provides user-specific services and applications [2][3].

Four layers of architecture consist of the application layer, data processing layer, network layer, and perception layer. The four-layer IoT architecture application layer specifies all IoT applications and connects end IoT devices and the network. This layer grants access to different services depending on the information gathered by sensors. The data processing layer gets information from the perception layer and ensures that data is sent on from genuine users while remaining safe from attacks. The network layer bridges the physical and the virtual worlds by acting as a medium for collecting data from sensors and sending the information to other network devices and networks. The perception layer, also known as the sensor layer, identifies and collects data from IoT devices. A network of sensors will likely be composed of distinct sensors of various types.

The perception layer has to distinguish between these different sensors and recognize their diverse operating procedures. A five-layer IoT architecture adds processing and business layers to the three-layer model design's physical, network, and application layers. The function of the perception and application layers in this approach is identical to that of the three-layered architecture. The transport layer sends and receives data from IoT sensors between the perception and processing layers. The processing layer, often called the middleware layer, handles the storage, analysis, and processing of large data volumes delivered by the transport layer. The business layer oversees the IoT system's general administration, including applications, business and business models, and user privacy. The IoT network can be Low Power Wide Area Network (LPWAN), Local and Personal Area Network (LAN / PAN), cellular network, or mesh network [4].

Recently, the Industrial Internet of things (IIoT) has emerged as the most rapidly developing revolutionary technology, with the ability to digitize and integrate numerous sectors, resulting in significant economic benefits for all relevant stakeholders. IIoT offers significant potential for developing a wide range of industrial applications, but they are also vulnerable to cyberattacks and need higher levels of security. The wide range of sensors in the IIoT network creates a significant volume of data, which has caught the interest of hackers worldwide. When it comes to protecting IIoT applications from cyberattacks, the intrusion detection system (IDS), which monitors network traffic and identifies network behavior, is regarded as one of the most important security measures [5]. As the IoT network technology continues to advance, cyber-attack detection measures are becoming more important in assuring the security of IoT networks. However, with the continuous development of IoT network traffic, standard IDSs are incapable of rapidly and reliably identifying complex and varied IoT network attacks, particularly those using low-frequency attacks [6].

IoT systems and cloud computing are among the growing internet-connected services, making cyber-attack prevention increasingly difficult. Cyberattacks have evolved into severe dangers to security and privacy since their influence on Internet of Things systems would result in financial losses and seriously harm human life. A crucial part of keeping a network safe is network intrusion detection to monitor and identify potential threats, events, and breaches. Security systems, such as firewalls and intrusion detection systems, are vulnerable to recent cyber threats since their techniques are focused on static attack signatures and cannot identify new attack variations [7].

Due to IoT’s significant economic effect and extensive influence on our lives is a desirable target for cybercriminals, and cybersecurity has risen to the top of the priority list for the IoT infrastructure. Even though cybersecurity has been studied for years, the massive IoT network and the introduction of new threats have rendered conventional approaches ineffective. The study conducted by Tsimenidis et al. [8] provides a comprehensive assessment of deep learning models that have been developed for Internet of Things intrusion detection. In a detailed, organized examination of how deep learning has been used for IoT cybersecurity and its unique contributions to creating efficient IoT intrusion detection systems, solutions have been classed by model.

It is becoming more concerning for most service providers as the number of computer networks and Internet threats continues to grow. It has prompted the development and deployment of IDSs to help prevent and mitigate the threats produced by network intruders. An important role in detecting network cyberattacks and abnormalities has been performed and continues to be played by intrusion detection systems. Many intrusion detection systems (IDSs) have been suggested by researchers worldwide to tackle the danger of network intruders. However, most previously suggested intrusion detection systems have significant rates of false alarms [9].

The primary aim of this paper is to develop an RNN based model for IoT networks. We effectively used kernel regularizer, bias regularizer, activity regularizer at LSTM, BiLSTM, and GRU layers, layer normalization, and activity regularization layers to develop a novel RNN and CNNRNN based model for multiclass and binary class classification. In contrast to Batch Normalization, the layer normalization method normalizes the activations of the previous layer for each given sample in a batch individually rather than across the entire batch. Activity regularization makes changes to the cost function that is based on input activities. Overfitting of the RNN model was assessed using drop layer, early stopping, and 5-fold cross-validation techniques.

We first used class weights during the training phase to resolve class imbalances in the datasets. The weights assigned to classes were determined by the number of instances of each class, so a minority class with a small number of instances will receive a high weight. New synthetic samples were created using the borderline SMOTE algorithm. The random state controls the algorithm’s randomization. The borderline SMOTE is being used to ensure the training set is balanced. The set feature we extracted from pcap files is generalized features that can be used for any IoT network. We evaluate our proposed multiclass and binary classification model using NSLKDD, BoT-IoT, IoT Network Intrusion, IoT-23, MQTT-IoT-
IDS2020, MQTTset, and IoT-DS2 datasets. The contributions of this paper are:

- Design, develop, and evaluate an anomaly detection model for IoT network using a recurrent neural network.
- A hybrid model of convolutional and recurrent neural networks for anomaly detection in IoT Networks.
- A lightweight recurrent neural network model for IoT network anomaly detection.
- Performance improvements of binary and multiclass classification models.

The remainder of this paper is organized as follows: Section II provides a discussion of the related work. Section III explains the design and development of the recurrent and convolutional neural networks model for multiclass and binary classification anomaly detection model for IoT networks. Section IV discusses the data collecting process for evaluating the proposed models. Section V presents the evaluation of the results, followed by a discussion of the findings in Section VI. Finally, Section VII concludes the paper and offers recommendations for future research.

II RELATED WORK

Millions of IoT devices are embedded in smart cities, enabling crucial applications such as smart homes, autonomous vehicles, and communications. The smart city is based on millions of heterogeneous sensors that do not support traditional security frameworks. Several manufacturers use inadequate protection mechanisms for their devices and fail to upgrade their firmware in response to recently discovered operational security breaches. To achieve comprehensive management of sensor operating systems while maintaining perfect security, smart cities need a common architecture that combines soft computing and Deep Learning [10].

While many machine learning methods have been employed to identify anomaly-based invasions, relatively few attempts have been made to classify recurrent neural network. The IoT has evolved fast in recent years, and cyber cyberattacks on IoT devices are becoming common. It has become more necessary to have an effective approach for identifying malicious attacks in the IoT domain to reduce security threats on IoT devices. Alkahtani et al. [11] developed a hybrid deep learning approach based on CNN and LSTM for detecting botnet assaults on nine market IoT devices. Their methodology proved effective in identifying botnet assaults from various IoT devices with an average accuracy of 90%.

Autonomous Vehicles (AVs) are prone to safety and security issues, risking human life. The Internet of Vehicles (IoVs) is a network of manually operated vehicles connected to the Internet. If cyber attackers get access to these vehicles, they might be utilized for malicious purposes. Khan et al. [12] have developed a multistage intrusion detection system to identify Autonomous Vehicles and Internet of vehicle intrusions while minimizing the number of false alarms. The proposed framework uses a BiLSTM architecture to detect intrusions from AVs' network gateways and communication networks. Additionally, the suggested system can detect zero-day outbreaks in networks of IoVs.

Smart Home Network IoT devices are very susceptible to sophisticated botnet assaults. Popoola et al. [13] examine the performance of RNNs in properly classifying network traffic samples belonging to minority groups is severely unbalanced network traffic data. To learn hierarchical representations of highly unbalanced network traffic data with different degrees of abstraction, many layers of RNN are stacked. To effectively capture the classifying properties of severely unbalanced network traffic samples, the stacked RNN model was used instead of the RNN model. The SRNN model also showed excellent generalization abilities when recognizing network traffic samples from minority groups.

A significant increase is being seen worldwide in the number of computer-controlled automobiles, which is on the rise at an alarming rate. Even though this improves the driving experience, it introduces a new security vulnerability in the automobile business. Desta et al. [14] suggested an LSTM-based intrusion detection system based on arbitration ID sequences. They used LSTM for IDS implementation. Their accuracy was just 60% using this strategy. They were only able to get an accuracy of 0.6 using this strategy. Because applying this finding to a real car would result in a large number of false negatives, they designed a second strategy that utilizes log loss as an anomalous indicator.

Malicious traffic identification using deep learning approaches has become a key component of network intrusion detection research and development. Most IDS need packets to be gathered into certain flows and then analyzed, incurring processing delays, even the most sophisticated ones. In order to identify malicious traffic at the packet level, Wang et al. [15] offer a deep learning strategy, employing hierarchical networks, which can learn the properties of communication using basic data packets. They also explored how data balance affects classification performance and time efficiency between the LSTM and GRU models.

Hao et al. [16] utilize the encoder to automatically process and analyze network packets to get properties that appropriately reflect the network packets. The variant-gated recurrent units dynamically understand data packets content and header attributes to increase the IDS detection rate significantly. The experimental findings from ISCX2012 indicate that intrusion detection using the proposed variant-gated recurrent units provides a greater level of accuracy and detection rate. Using binary weights and activation functions, their proposed model provides a greater representation of the data than the original raw data and helps to minimize the amount of memory and access time. An overview of recent developments in deep learning for intrusion detection is presented in Table 1. In Table 1, DR mean detection rate, Acc represents the accuracy, F1 represents the F1 score, and pr represents precision.

Recent years have seen a tremendous influence on industrial production because of the fast development and widespread use of new technologies, resulting in smart manufacturing (SM). However, industrial systems based on the IoT are currently one of the most targeted sectors for various cyberattacks. Anomaly Detection is proposed by
## TABLE 1. An overview of the related work in the area of deep learning for intrusion detection.

| Article | Year | Model | Dataset | Classification | Performance |
|---------|------|-------|---------|----------------|-------------|
| [17]    | 2017 | CNN   | KDD     | Binary         | Acc = 99.60 |
| [18]    | 2017 | CNN   | ISCX2012| Multiclass    | Acc = 99.69 |
| [19]    | 2017 | RNN   | NSLKDD  | Multiclass    | Acc = 83.28 |
| [20]    | 2017 | LSTM  | NSLKDD  | Binary         | Acc = 98.85 |
| [21]    | 2018 | LSTM  | AWID    | Binary         | Acc = 99.22 |
| [22]    | 2018 | BiLSTM| UNSW-NB15| Binary   | Acc = 95.71 |
| [23]    | 2018 | DBN   | NSLKDD  | Multiclass    | Acc = 82.08 |
| [24]    | 2018 | BGRU+MLP | KDD99   | Multiclass    | Acc = 99.34 |
| [25]    | 2018 | RNN   | NSLKDD  | Multiclass    | Acc = 81.29 |
| [26]    | 2018 | LSTM  | NSLKDD  | Multiclass    | Acc = 89.00 |
| [27]    | 2018 | LSTM  | NSLKDD  | Multiclass    | Acc = 73.18 |
| [28]    | 2018 | CNN   | NSLKDD  | Multiclass    | Acc = 80.13 |
| [29]    | 2019 | LSTM  | ISCX2012| Binary         | Acc = 99.99 |
| [30]    | 2019 | GRU   | Personal Dataset | Multiclass | Acc = 96.08 |
| [31]    | 2019 | GRU   | Personal Dataset | Multiclass | Acc = 95.60 |
| [32]    | 2019 | RNN   | BoT-IoT | Multiclass    | Acc = 98.20 |
| [33]    | 2019 | CNN   | NSLKDD  | Binary         | Acc = 99.67 |
| [34]    | 2019 | CNN   | NSLKDD  | Multiclass    | Acc = 89.41 |
| [35]    | 2019 | DNN   | KDD99   | Multiclass    | Acc = 92.90 |
| [36]    | 2019 | DNN   | KDD99   | Binary         | Acc = 93.00 |
| [37]    | 2019 | DNN   | UNSW-NB15| Binary       | Acc = 99.19 |
| [38]    | 2019 | DNN   | UNSW-NB15| Multiclass | Acc = 97.04 |
| [39]    | 2019 | DNN   | CICIDS2017| Binary   | Acc = 97.73 |
| [40]    | 2019 | CNN-LSTM | NSLKDD  | Binary         | Acc = 98.90 |
| [41]    | 2019 | LSTM  | KDD99   | Binary         | Acc = 99.43 |
| [42]    | 2019 | GRU   | KDD99   | Binary         | Acc = 99.06 |
| [43]    | 2019 | Bi-LSTM| KDD99   | Binary         | Acc = 82.20 |
| [44]    | 2019 | GRU   | NSLKDD  | Binary         | Acc = 82.87 |
| [45]    | 2019 | GRU   | Gas Pipeline | Binary  | Acc = 91.70 |
| [46]    | 2019 | GRU   | ISCX2012| Binary         | Acc = 99.90 |
| [47]    | 2019 | DNN   | BoT-IoT | Multiclass    | Acc = 98.09 |
| [48]    | 2020 | CNN-LSTM| CISC-2010| Binary   | Acc = 91.54 |
| [49]    | 2020 | CNN-LSTM| CICISC2017| Binary   | Acc = 93.00 |
| [50]    | 2020 | CNN-LSTM| CICIDS2017| Binary   | Acc = 99.03 |
| [51]    | 2020 | LSTM  | KDD99   | Binary         | Acc = 98.94 |
| [52]    | 2020 | LSTM-CNN | CICIDS2017| Multiclass   | Acc = 98.60 |
| [53]    | 2020 | CNN   | BoT-IoT | Binary         | Acc = 91.00 |
| [54]    | 2020 | DNN   | BoT-IoT | Multiclass    | Acc = 99.79 |
| [55]    | 2020 | RNN   | BoT-IoT | Multiclass    | Acc = 98.79 |
| [56]    | 2020 | CNN, GRU | Multiple | Binary       | Acc = 99.42 |
| [57]    | 2020 | CNN–WDSLSTM | UNSW-NB15| Binary       | Acc = 97.17 |
| [58]    | 2020 | SNN   | BoT-IoT | Multiclass    | Acc = 98.73 |
| [59]    | 2020 | RNN   | BoT-IoT | Binary         | Acc = 98.31 |
| [60]    | 2021 | DSSTE+LSTM | NSLKDD  | Multiclass    | Acc = 81.78 |
| [61]    | 2021 | LSTM Autoencoder | UNSW-NB15| Binary       | Acc = 98.00 |
| [62]    | 2021 | LSTM  | CIC-IDS2018| Multiclass | Acc = 94.00 |
| [63]    | 2021 | LSTM  | CICIDS2017| Binary      | Acc = 99.55 |
| [64]    | 2021 | BiLSTM| KDD     | Binary         | Acc = 99.70 |
| [65]    | 2021 | IE-DBN| KDD     | Multiclass    | Acc = 98.12 |
| [66]    | 2021 | IE-DBN| NSLKDD  | Multiclass    | Acc = 98.79 |
| [67]    | 2021 | LSTM–GRU | BoT-IoT | Binary       | Acc = 99.76 |
| [68]    | 2021 | LSTM–GRU | NSL     | Binary         | Acc = 99.14 |
| [69]    | 2021 | LSTM  | KDD99   | Binary         | Acc = 98.88 |
| [70]    | 2021 | BiLSTM| NSLKDD  | Binary         | Acc = 94.26 |
| [71]    | 2021 | BiLSTM| NSLKDD  | Multiclass    | Acc = 91.36 |
| [72]    | 2021 | CNN   | InSDN   | Binary         | Acc = -97.50 |
| [73]    | 2021 | CNN   | InSDN   | Multiclass    | Acc = -97.50 |
| [74]    | 2021 | ANN   | CTU-13  | Binary         | Acc = 99.94 |
| [75]    | 2021 | Reinforcement | NSLKDD  | Multiclass    | Pr = 96.50 |
| [76]    | 2021 | LSTM  | UNSWNB15| Binary       | Acc = 98.88 |
| [77]    | 2021 | GE    | MQTT    | Binary         | Acc = 97.94 |
| [78]    | 2021 | DT    | MQTTset | Binary         | Acc = 99.47 |
| [79]    | 2021 | RF    | MQTTset | Binary         | Acc = 99.68 |
Huong et al. [65] to identify industrial control systems cyberattacks. The framework outperforms existing time series data detection solutions in terms of detection performance. In the Industrial IoT (IIoT), a vast quantity of data processing takes place in the Cloud and at the edge to perform various types of analytics. IIoT routing attacks can be detected using Nayak et al. [66] deep learning-based routing attack detection technique. The proposed solution uses parallel learning and detection to facilitate deep learning on IIoT devices with limited processing power. The parallel model output is tested in an IIoT network to compare the performance of distributed and centralized threat detection in an RPL network. Training time is significantly reduced when a parallel GAN model is used.

Although various relevant research has utilized deep learning for NIDS, most of these techniques fail to account for the effect of overfitting when deep learning algorithms are implemented. Therefore, the anomaly detection system’s resilience may be compromised, making it less effective in detecting zero-day cyberattacks. Convolutional neural networks (CNNs) and a novel regularizer technique are proposed by Elsayed et al. [59] to categorize flow traffic into normal and attack categories. Additionally, they propose a lightweight NIDS based on training CNN-based models with fewer features without sacrificing model performance significantly. Advanced information and communication technologies have facilitated the dissemination of a large quantity of information, which continues to grow day by day through the Internet and the creation of new added value via the use of internet-based activities. Increasing numbers of diverse connecting points with high computing capability have expanded cybersecurity concerns. Biswas et al. [57] have suggested a new deep learning strategy to distinguish malicious Botnet traffic from normal traffic.

The GRU is the most effective model for botnet detection; however, it is computationally costly, yet it can handle large amounts of data and identify sequences efficiently. Laghrissi et al. [58] developed deep learning techniques for identifying attacks based on the LSTM algorithm. They employed PCA and Mutual information to reduce the data's dimensionality and choose the best features. These techniques are also evaluated in terms of performance and processing time. The extensive availability of Internet services around the globe has presented a significant challenge to service providers in terms of protecting their systems, particularly against new breaches and threats.

The battery life of IoT devices is a huge issue since the devices use a lot of energy once they connect with each other. IoT devices would also include important network data, posing severe privacy and security concerns. Botnet attacks are significant threats to IoT-based smart devices. Using statistical learning-based botnet detection, Asharaf et al. [67] secure IoT-based smart networks against botnet cyberattacks. Louk et al. [68] address a gap in the literature by showing the importance of ensemble-based models for detecting potential attacks in a cyber-physical power system. They balanced the dataset by employing oversampling and undersampling techniques. Oversampling and undersampling were beneficial in a boosting-based ensemble but were ineffective in a bagging-based ensemble. Ensemble learners have outperformed single learners in a wide variety of applications, including the cybersecurity area. However, the majority of previously published works continue to produce unsatisfactory outcomes as a result of insufficient ensemble design. Nkonyereye et al. [69] demonstrate the efficacy of stacking ensemble-based models for anomaly detection, where a deep neural network is utilized as the basic learner model. The suggested model's efficacy and that of the underlying DNN model are experimentally compared using a variety of performance criteria.

A two-level hybrid anomalous activity detection model has been proposed to detect intrusions in IoT networks, which detects abnormal activity at level1 and analyses the anomalous activity identified at level2 [70]. The level-2 model selects relevant features using Recursive Feature Elimination, oversampling using Synthetic Minority Over-Sampling Technique (SMOTE), and cleaning the datasets using Edited Nearest Neighbors (ENN). A three-layer system was proposed to detect intrusions in a smart grid system [71]. The proposed structure includes an IDS in each Home Area Network (HAN) and Neighborhood Area Network (NAN) and many IDS sensors in the WAN. In modeling anomaly-based intrusion detection systems, feature selection is important. A filter-based feature selection methodology was proposed for anomaly-based intrusion detection systems that leverage information gain by considering each feature's consistency, dependency, content, and distance [72]. Additionally, using an industrial control system dataset, the suggested model for feature selection was evaluated for anomaly detection in SCADA networks [73].

It is challenging to extract valuable information from network traffic to identify possible anomalies. We investigate many types of network flow features to resolve this challenge [74][75]. A feed-forward neural network-based method for identifying anomalous activity in IoT networks based on flow and control flags features has been presented [76]. The model was assessed for binary and multiclass classification using a variety of IoT intrusion datasets. Using conditional GANs to create realistic distributions for a given feature set, a framework for identifying anomalies in IoT networks was proposed, which overcomes data imbalance by using conditional GANs to detect abnormalities in IoT networks [77].

Since IoT is increasingly being used in critical infrastructures and cyber-physical systems, there has been a significant increase in research efforts to develop efficient defenses against cyber-attack. IoT networks can be protected from a wide range of cyberattacks using a framework called Boost-Defense, developed by Al-Haija et al. [78]. A strong classifier for identifying and categorizing cyber-attacks in IoT networks is built using AdaBoost machine learning, Decision Trees, and substantial data engineering approaches. NIDS are lightweight, versatile, and anomaly-based construct profiles for normal and malicious behavior using a variety of ways. Al-Haija et al. [79] used machine-learning approaches to develop, construct, and evaluate an anomaly-based IoT NIDS. It was modeled as supervised multiclass learning, where a classification function was learned to map a collection of labels to ten classes.
III PROPOSED MODEL

A. RECURRENT NEURAL NETWORK

The usage of neural networks has the potential to enhance every part of our daily life. An artificial neural network with a sequential information structure is known as a recurrent neural network. They are referred to as recurrent because they execute the same function on each sequence element, with the outcome depending on prior computations. RNNs are networks that include loops, which allow for data persistence. Long Short-Term Memory (LSTM) networks are a kind of RNN that can learn long-term dependencies. Hochreiter et al. [80] introduce the LSTM networks. The LSTM networks perform very well across a broad range of issues and are now extensively utilized. LSTMs are designed to prevent long-term dependence [81]. Each recurrent neural network is composed of a chain of repeating neural network modules. A recurrent neural network includes loops, enabling information to be retained in the network. In Fig. 1, a simple recurrent neural network with loops is shown. The neural network in the illustration above, A, examines the input $x_t$ and then generates output $h_t$. A loop enables data to be transferred from one network phase to the next. LSTMs are expressly intended to prevent the issue of long-term dependence. Each recurrent neural network comprises a chain of repeating neural network modules. Table 2 shows a list of symbols used to help understand the various concepts presented in the following sections.

| Symbol | Meaning |
|--------|---------|
| $f_t$  | forget gate |
| $i_t$  | input gate |
| $o_t$  | output gate |
| $\sigma$ | sigmoid function |
| $\tanh$ | tanh function |
| $w_x$  | weight for respective gate($x$) neurons |
| $h_{t-1}$ | output of previous LSTM block |
| $x_t$  | input at current timestamp |
| $b_x$  | biases for the respective gate($x$) |
| $C_{t-1}$ | LSTM previous memory content |
| $C_t$  | LSTM current memory content |
| $\tilde{C}_t$ | LSTM new memory content |
| $r_t$  | GRU reset gate |
| $z_t$  | GRU updated gate |
| $h_t$  | GRU current memory content |
| $h_{t-1}$ | LSTM or GRU final memory content at current time step |

The first stage of LSTM is to determine which information from the cell state will be discarded. A sigmoid layer, known as the "forget gate layer," makes this determination. The sigmoid layer examines the values in $h_{t-1}$ and $x_t$ and returns a value between 0 and 1 for each value in the cell state $C_{t-1}$. A 1 indicates "totally retain this," whereas a 0 indicates "entirely discard this." The forget layer functioning of the LSTM is shown in Fig. 3, and the operation of forgets gate layer is represented by (1). It is necessary to determine what additional information will be stored in the cell state as a result of the input gate layer decision. The input gate layer, which is a sigmoid layer, determines the values to update first. Equation (2) represents the operation of the input gate layer, and the input gate layer functioning of the LSTM is shown in Fig. 4. A tanh layer generates a vector of potential nominee values, $\tilde{C}_t$ that may be included in the state. A tanh layer operation is represented by (3). Equation (4) updates the cell state $C_{t-1}$ into the new $C_t$, and (4) operation is represented in Fig. 5.

$$f_t = \sigma(W_f, [h_{t-1}, x_t] + b_f) \quad (1)$$
$$i_t = \sigma(W_i, [h_{t-1}, x_t] + b_i) \quad (2)$$
$$\tilde{C}_t = \tanh(W_{C_t}, [h_{t-1}, x_t] + b_C) \quad (3)$$
$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t \quad (4)$$

The last step is to determine the output. This result will be based on the current state of our cell, but it will be a simplified form of it. First, a sigmoid layer is implemented that sets the output of the cell state based on the values received. Equation (5) represents this operation. As shown in (6), the system implements a cell state to the tanh function and then multiplies it by the output of the sigmoid gate. This process ensures that only the values that have been selected to return are sent. The output $h_t$ operation are presented in (6) and Fig. 6.

$$h_t = \sigma(W_h, [h_{t-1}, x_t] + b_h) \quad (5)$$

FIGURE 1 Simple recurrent neural network with loops characteristic.

FIGURE 3 LSTM forget layer operation.
Cho et al. [82] developed a slight variant of LSTM called Gated Recurrent Unit (GRU). The GRU has only two gates, while the LSTM has three gates. The LSTM input and forget gates are combined into an “update gate” in GRU. The GRU eliminated the cell state and transferred information via the hidden state. Similar to an LSTM forget and input gates, the update gate operates in the same way. It determines which data should be discarded and which should be included. The reset gate is used to determine how much previous information should be erased from the memory. The Tensor operations that GRU use are fewer in number compared to LSTM, and as a result, they are faster to train than the LSTM model. A single GRU cell operation is presented in Fig. 7. Equation (7) represents the reset gate operation, (8) describes the update gate operation, (9) shows the current memory state of the GRU, and (10) represents the final memory state of the GRU.

\[
\begin{align*}
    r_t &= \sigma(W_r [h_{t-1}, x_t]) \\
    z_t &= \sigma(W_z [h_{t-1}, x_t]) \\
    h_t &= \text{tanh}(W_r [r_t \times h_{t-1}, x_t]) \\
    h_t &= (1-z_t) \times h_{t-1} + z_t \times \tilde{h}_t
\end{align*}
\]

BiLSTM is an extension of conventional LSTM that enhances model performance on sequence classification tasks by learning in both directions simultaneously. Because they train two LSTM on the input sequence instead of one, Bidirectional LSTM is useful when all timesteps in the input sequence are accessible. BiLSTM considers forward and backward activation to calculate the output.

B. PROPOSED MODEL

Deep learning techniques are gaining popularity due to their ability to detect computer network threats and abnormalities in various applications. A recurrent neural network has shown to be effective in a variety of areas. Due to its better capability, this article presents a model based on a recurrent neural network. This paper design and develop LSTM, BiLSTM, and GRU-based models for anomaly detection in IoT networks. The model consists of an input layer, output layer, and four recurrent, activation, normalization, activity regularization, and dropout layers. Overfitting is a major concern for deep learning models. We used kernel, bias, and activity regularizers at the LSTM, BiLSTM, and GRU layers. The kernel regularizer imposes a penalty on the kernel of the layer; the bias regularizer enforces a penalty on the bias of the layer while the output of the layer is penalized by the activity regularizer. The regularizer uses the value of l1-l2 to compute the value for kernel, bias, and activity regularization.

The Keras have different activation functions for activation layers. For the activation layer in this paper, we adopted the LeakyReLU activation function. Next, we used layer normalization, which usually improves the acceleration and stabilization of the learning process by reducing error rates. A neural network can be encouraged to learn weak features by using an activity regularization layer. The activity regularization support l1, l2, and l1-l2 regularization techniques. The activity regularization layer updates input activity that is dependent on a cost function. A recurrent neural network is prone to overfitting and will need
significant adjustments to the training dataset to avoid it. By disregarding certain neurons during the training period, a dropout layer mitigates the risk of overfitting. Neuron weights in a neural network settle into their environment as a network learns. Four recurrent, activation, normalization, activity regularization, and dropout layers were used across LSTM, BiLSTM, and GRU models. Before the output layer, we used a dense layer with 512 neurons and a ReLU activation function. The output layer is the last layer of the model, and the number of neurons in this layer is dependent on the number of classes in the dataset.

We design and build an anomaly detection model in IoT networks using a recurrent neural network. The same framework was used to construct LSTM, BiLSTM, and GRU models. Fig. 8 shows a layered perspective of the proposed recurrent neural network-based LSTM, BiLSTM, and GRU model. Table 1 shows the proposed LSTM model components, including one input layer, four LSTM layers, four activation layers, four normalization layers, four activity regularization layers, four dropout layers, one dense layer, and one output layer. The input layer receives network traffic with 64 features. First, a 64×1 input vector is created to suit the 64 best features selected by our feature selection method presented in [83]. The LSTM model uses 512 units at each LSTM layer. The LSTM model also uses kernel, bias, and activity regularizers at the LSTM layer. The regularizer uses the 11-2 function for penalties. The activation layer uses the LeakyReLU activation function, an alternative to Rectified Linear Unit (ReLU) leaky implementation. It does not contain zero-slope sections; LeakyReLU solves the "dying ReLU" issue. There is a significant increase in the pace of learning. It's been shown that having the "mean activation" near to 0 speeds up training. In contrast to ReLU, leaky ReLU is more "balanced," As a result, it can learn more quickly [84]. A model normalization layer may usually aid in accelerating and stabilizing the learning process by reducing error rates. The LSTM model training and validation details are presented in algorithm 1. The BiLSTM and GRU models were created using the same technique.

Recurrent network hidden state dynamics can very well be stabilized via layer normalization. To avoid introducing additional dependencies across training instances, unlike batch normalization, layer normalization calculates normalization statistics from the total of inputs to the neurons of the hidden layer [85]. Overfitting is a major issue in neural networks. To reduce the chance of overfitting, we used the regularization and dropout layers. The activity regularization layer makes changes to the input activity that is dependent on the cost function. 11-2 factors serve as a baseline for the activity regularization layer functionality. During training, the dropout layer changes some input units to 0 at a frequency equal to the dropout rate, thereby aiding overfitting prevention. This implies that their impact on downstream neuron activity is eliminated temporally on the forward pass, and any weight changes are not transferred to the cell on the backward trip.

BiLSTM is a sequence processing model with two LSTMs: one processing input in the forward direction and the other LSTM processing in the backward direction. In this way, BiLSTM-based modeling provides higher predictions than normal LSTM-based modeling due to the extra training of data it receives. Long training time is needed for the BiLSTM model. The proposed BiLSTM model consists of an input layer, four BiLSTM layers, four activation layers, four-layer normalization layers, four activity regularization layers, four dropout layers, one dense layer, and one output layer, as shown in Table 3.

The GRU was created to address the vanishing gradient issue inherent in the conventional recurrent neural network. GRU may also be seen as a variant of the LSTM since they are both constructed similarly and, in certain instances, provide equally good outcomes. GRUs use so-called update and reset gates to overcome the vanishing gradient issue of a conventional RNN. To put it another way, two vectors determine what information should be sent to the output device. The proposed GRU model also consists of an input layer, four GRU layers, four activation layers, four-layer normalization layers, four activity regularization layers, four

![FIGURE. 8 The proposed LSTM, BiLSTM, and GRU model layer's view, parameter, and hyperparameters for multiclass classification.](image)
dropout layers, one dense layer, and one output layer. Table 3 summarizes the proposed recurrent neural network model.

### Algorithm 1: LSTM Model

**Input:** X, y ← Network Flows  
**Output:** metrics  
Set LSTM, activation, normalization, regularization, and dropout layers parameters to define the LSTM model (LM)  
**Model** ← LM  
**Initialize** batch size, optimizer, learning rate, epochs  
for epochs= 1 to n do  
while early_stopping criteria= false  
train Model  
validate Model  
monitor='train loss'  
adjust loss function using sparse categorical cross-entropy  
end while  
end for  
**Evaluate** Model  
**Perform** prediction using Model  
**Calculate** metrics  
return metrics

### TABLE 3. Summary of parameter and hyperparameters recurrent neural network model for multiclass classification.

| Layer         | Layer Name | Layer | Configuration                                      |
|---------------|------------|-------|---------------------------------------------------|
| Input         | Input Layer| 1     | 64 input features                                 |
| Hidden layers | LSTM or BiLSTM or GRU | 4     | Units=512, Kernel regularizer, Bias regularizer, Activity regularizer |
|               | Activation | 4     | LeakyReLU(alpha=0.2)                              |
|               | Normalization | 4     | Axis=1, Center=True, Scale=True                   |
|               | Regularization | 4     | l1=0.0001, l2=0.0001                              |
|               | Dropout     | 4     | Dropout rate =0.2                                |
| Classification| Dense      | 1     | Neuron =512, Activation=ReLU                      |
| Output        | Output     | 1     | The number of neurons is equal to the number of classes in the dataset, Activation=SoftMax |
| Hyperparameters| Early Stopping | (monitor='loss', patience=5, verbose = 1), optimizers= Adam, Loss function= sparse categorical_cross_entropy, Learning rate=0.001, Batch size=120, epochs= 100 to 500 |

In deep learning, a Convolutional Neural Network (CNN) is an algorithm that takes in an input picture, assigns significance to various image elements, and can differentiate between them. Compared to other classification methods, the amount of preprocessing needed by a convolutional neural network is much less. Recent research has shown that a convolutional neural network can produce outstanding speech recognition and image identification results. The advantages of a convolutional neural network may also be more fully revealed if intrusion detection problems are transformed into image recognition problems. A convolutional neural network can effectively capture spatial and temporal correlations related to intrusion detection. The design provides a better match to the intrusion detection issue because of the reduced number of parameters and weights' reusability [86].

We design and build a model using convolutional and recurrent neural networks. Convolutional neural network learns input features without losing essential information, making them ideal for prediction. A layered view of the proposed convolutional neural network-based LSTM, BiLSTM and GRU models is presented in Fig. 9. Table 4 shows the configurations of LSTM, BiLSTM, and GRU models based on a convolutional neural network for multiclass classification. We used a convolutional 1D model for the convolutional layer. Convolution is computed in 1D using temporal access, with the kernel moving in just one direction. CNN1D requires two-dimensional input, and
output data is often used to process time-series data. First, an input vector 64x1 is created to correspond to the 64 best features selected by the feature selection method. The convolutional layer is combined with the activation, normalization, regularization, and dropout layers. The activation layer LeakyReLU activation function with an alpha value of 0.2. The layer normalization uses a feature axis for normalization. The activity regularization layer uses l1-l2 functions, and the dropout layer randomly drops neurons to reduce the chance of overfitting. A pooling layer is used to minimize the dimension of the convolved features to reduce the computing resources needed to analyze the data. We utilized an average pooling layer in the model.

The activation, normalization, regularization, and dropout layers are combined with two LSTM, BiLSTM, or GRU layers. The dense layer receives input from the final dropout layer, and the output of the dense is transferred to the output layer. The dense layer uses 512 neurons. The number of neurons in the output is determined by the number of classes in the dataset. The output layer of our proposed model has four, five, six, ten, and twenty neurons. The following benefits accrue from the initial model interface being a convolutional neural network: The spatial and temporal correlations associated with an anomaly detection problem may be captured effectively by a convolutional neural network when the optimal filters are used and with fewer parameters and reusable weights, the architecture is more suited to fit the anomaly detection data. The pooling layer lowers computing power by reducing the dimension of the features.

The proposed binary classification model utilizing a single recurrent neural network-based hidden layer. A layered view of the binary classification model is presented in Fig. 10. The model consists of an input layer, and the number of inputs to the model is equal to the number of features in the dataset. The hidden layer consists of LSTM, BiLSTM, or GRU layer combined with the activation, normalization, activity regularization, and dropout layer. The activation layer uses the LeakyReLU action function. The normalization layer uses batch normalization, and the regularization layer uses activity regularization along with the l1-l2 penalty function. The regularization and dropout layers lower the likelihood of the model overfitting; the dense layer use 512 neurons and the ReLU activation function. The dense layer works as a classification layer, and the output layer uses two neurons to classify the data as normal or anomalous. The configurations of the LSTM, BiLSTM, and GRU models for binary classification are shown in Table 5.

### Table 4. Summary of parameter and hyperparameters convolutional neural network-based LSTM, BiLSTM, and GRU models for multiclass classification.

| Layer | Layer Name | Configuration |
|-------|------------|---------------|
| Input | Input Layer| 1 64 input features |
| Hidden layers | CNN1D | 1 Filter =64, Kernel size =5, Padding= same, Kernel regularizer, Bias regularizer, Activity regularizer |
| LSTM or BiLSTM or GRU | 2 Units =512, Kernel regularizer, Bias regularizer, Activity regularizer |
| Activation | LeakyReLU(alpha=0.2) |
| Layer Normalization | Axis=1, Center= True, Scale= True |
| Regularization | 1 11 = 0.0001, 12 = 0.0001 |
| Dropout | Dropout rate = 0.2 |
| Classification | Dense | 1 Neuron =512, Activation= ReLU |
| Output | Output | The number of neurons is equal to the number of classes in the dataset, Activation= SoftMax |
| Hyperparameters | Early Stopping (monitor= loss, patience= 5, verbose = 1), optimizers= Adam, loss function= sparse categorical _ cross entropy, Learning rate= 0.001, Batch size= 120, epochs= 100 to 500 |

### Table 5. Summary of parameter and hyperparameters of LSTM, BiLSTM and GRU models for binary classification.

| Layer | Layer Name | Configuration |
|-------|------------|---------------|
| Input | Input Layer| 1 64 input features |
| Hidden layers | LSTM or BiLSTM or GRU | 1 Units =512, Kernel regularizer, Bias regularizer, Activity regularizer |
| Activation | LeakyReLU(alpha=0.2) |
| Layer Normalization | Axis=1, Center= True, Scale= True |
| Regularization | 1 11 = 0.0001, 12 = 0.0001 |
| Dropout | Dropout rate = 0.2 |
| Classification | Dense | 1 Neuron =512, Activation= ReLU |
| Output | Output | Two neurons, Activation= SoftMax |
| Hyperparameters | Early Stopping (monitor= loss, patience= 5, verbose = 1), optimizers= Adam, loss function= sparse categorical _ cross entropy, Learning rate= 0.001, Batch size= 120, epochs= 100 to 500 |
IV DATA COLLECTION

BoT-IoT [87], IoT-NI [88], IoT-23 [83], MQTT [83], MQTTset [83], and NSLKDD [89] datasets were used to evaluate the proposed models. The first step included processing pcap files of BoT-IoT [90], IoT Network Intrusion [91], MQTT-IoT-IDS2020 [92], MQTTset [93], and IoT-23 [94]. CICFlowmeter [95] extracts 79 network features from pcap files and stores them as CSV files. The adapted datasets can be accessed from [96]. Our previous work [83] describes the testbed setup and the attacks performed on each dataset in detail. It is determined how to label each dataset instance based on criteria pre-defined criteria for each dataset instance. Because the network characteristics flow ID, source IP, source port, destination IP, and timestamp characterize communication inside a specific IoT network; they were removed from all adapted datasets. Non-numeric features are transformed into numeric features. Duplicate instances were produced when the pcap data was converted to CSV files. All datasets have had these redundant instances deleted. The column characteristics were normalized within a given range to eliminate extreme values and significantly speed up calculations (-1, 1). The mean imputation method is used to fill in missing values in datasets. It replaces missing values with the average of a variable's remaining values that have missing information. Table 6 presents the BoT-IoT, IoT Network Intrusion, IoT-23, MQTT-IoT-IDS2020, and MQTTset datasets with and without redundancy.

### TABLE 6. BoT-IoT, IoT Network Intrusion, IoT-23, MQTT-IoT-IDS2020, and MQTTset dataset instances and classes.

| No | Category                  | With Redundancy | Without Redundancy |
|----|---------------------------|----------------|-------------------|
| (a) | BoT-IoT dataset instances and classes |                 |                   |
| 0  | Normal                    | 105202         | 7511              |
| 1  | DoS                       | 57027372       | 17420085          |
| 2  | DDoS                      | 37077674       | 18199716          |
| 3  | Scan                      | 4734836        | 4108211           |
| 4  | Data Theft                | 454715         | 445799            |
| (b) | IoT-NI dataset instances and classes |                 |                   |
| 0  | Normal                    | 40073          | 39851             |
| 1  | DoS                       | 59391          | 59391             |
| 2  | MITM                      | 35377          | 32909             |
| 3  | Mirai                     | 415677         | 366971            |
| 4  | Scan                      | 75265          | 72122             |
| (c) | IoT-23 dataset instances and classes |                 |                   |
| 0  | Normal                    | 4313776        | 4253672           |
| 1  | Attack                    | 1716778        | 1699608           |
| 2  | Mirai                     | 756            | 756               |
| 3  | File Download             | 8035           | 7707              |
| 4  | Heartbeat                 | 12895          | 12648             |
| 5  | C&C                       | 23981          | 20612             |
| 6  | Torii                     | 33858          | 24492             |
| 7  | Port Scan                 | 65944863       | 2999999           |
| 8  | DDoS                      | 20768988       | 4619869           |
| 9  | Okinu                     | 13718252       | 12908506          |
| (d) | MQTT dataset instances and classes |                 |                   |
| 0  | Normal                    | 334318         | 167159            |
| 1  | MQTT Bruteforce           | 2002780        | 2001972           |
| 2  | Scan-A                    | 31245          | 29276             |
| 3  | Scan-U                    | 33404          | 27843             |
| 5  | Sparta                    | 1252259        | 1217198           |
| (e) | MQTTset dataset instances and classes |               |                   |
| 0  | Normal                    | 4400699        | 420136            |
| 1  | Bruteforce                | 4547           | 4513              |
| 2  | MQTTFlood                 | 77793          | 77756             |
| 3  | MalariaDoS                | 11408          | 11265             |
| 5  | Malformed                 | 3580           | 3535              |
| 6  | SlowITe                   | 3044           | 3044              |
The BoT-IoT dataset was generated using pcap files by Koroniatis et al. [90]. The BoT-IoT dataset testbed comprises virtual machines (VMs) connected to the network through a LAN and the Internet. The PFsense system establishes a connection between the VMs and the Internet. A realistic smart home framework was developed using five IoT devices operated locally and connected to the cloud services using the node-red system for producing network traffic. Normal network traffic is generated with the help of the ostinato tool. The Ubuntu server delivers IoT resources for simulating a real-world IoT network, while Kali Linux serves as an attack system. Transmitting IoT messages into the Cloud is accomplished using the MQTT protocol. The BoT-IoT dataset category classes are Normal, DDoS, DoS, Scan, and Theft. The BoT-IoT dataset instances details are presented in Table 6(a). We combined DDoS and DoS categories to make it one attack class.

The IoT Network Intrusion detection dataset was generated by Kang et al. [91]. Two IoT devices, SKT NGU and EZVIZ Wi-Fi cameras, were used as victim devices. The wireless network adapter monitor mode collects the network traffic files. Except for the Mirai Botnet category, all cyberattacks are constructed of packets collected when modeling cyberattacks that use software such as Nmap. A laptop produced attack packets that were then altered to seem as though they came from an IoT device in the instance of Mirai Botnet. The IoT Network Intrusion detection dataset category classes are Normal, DoS, MITM, Mirai, and Scan. Table 6(b) shows the details of the IoT Network Intrusion detection dataset instances. Stratosphere Laboratory at CTU University in the Czech Republic generated the IoT-23 dataset using three real IoT devices [94]. The IoT-23 dataset comprises twenty-three instances of varied IoT network traffic generated by different IoT devices. The IoT-23 dataset provides researchers with a big, annotated dataset of real IoT network traffic to create machine learning models. The IoT-23 dataset category classes are Normal, Attack, Mirai, File Download, Heartbeat, C&C, Torii, Port Scan, DDoS, Okiru. Table 6(c) displays the details of the IoT-23 dataset instances.

The MQTT-IoT-IDS2020 dataset was created by Hindy et al. [92]. Five recorded scenarios are included in the dataset: one of normal operation and four of attack. The dataset represents a real MQTT IoT network in a typical operating situation. The dataset covers popular MQTT attacks and situations for real-world testing devices. The MQTT-IoT-IDS2020 dataset classes are Normal, MQTT-Bruteforce, Scan-A, Scan-U, and Sparta. Table 6(d) lists the instances of the MQTT-IoT-IDS2020 dataset that have been created. Security of IoT networks and devices is crucial due to the increasing number of linked devices and networks. The MQTTset dataset was generated by Vaccari et al. [93]. The testbed for the MQTTset dataset comprises ten IoT sensors in a typical smart home. MQTTset was created with the help of MQTT and CoAP protocols. The authors installed several IoT sensors linked to an MQTT broker to generate a dataset reflective of a real network. The MQTTset dataset category classes are Normal, Bruteforce, DoS, Flood, Malformed, and SlowITe. Table 6(e) displays the details of the MQTTset dataset instances.

Table 7 displays the details of the IoT-DS2 dataset instances. The BoT-IoT, IoT Network Intrusion, IoT-23, MQTT, and MQTTset datasets were combined to make a dataset with many attacks. The new dataset is named IoT-DS2. Table 7 displays the details of the IoT-DS2 dataset instances.

| No | Category                      | BoT-IoT | IoT-NI | IoT-23 | MQTT | MQTTset | IoT-DS2 |
|----|-------------------------------|---------|-------|--------|------|---------|---------|
| 0  | Normal                        | -       | -     | 4253672| -    | -       | -       |
| 1  | DDoS                          | 17420085| -     | -      | -    | -       | -       |
| 2  | DoS                           | -       | 59391 | -      | -    | -       | -       |
| 3  | MITM ARP Spoofing             | -       | 32909 | -      | -    | -       | -       |
| 4  | Mirai                         | -       | 366971| -      | -    | -       | -       |
| 5  | MQTT Bruteforce               | -       | -     | 2001972| -    | -       | -       |
| 6  | Sparta                        | -       | -     | 1217198| -    | -       | -       |
| 7  | Theft                         | 6257    | -     | -      | -    | -       | -       |
| 8  | Attack                        | -       | 1699608| -     | -    | -       | -       |
| 9  | C&C                           | -       | 20612 | -      | -    | -       | -       |
| 10 | File Download                 | -       | 7707  | -      | -    | -       | -       |
| 11 | Heartbeat                     | -       | 12648 | -      | -    | -       | -       |
| 12 | Okiru                         | -       | 12908506| -   | -    | -       | -       |
| 13 | OS Scan                       | 35675   | -     | -      | -    | -       | -       |
| 14 | Port Scan                     | -       | 2999999| -     | -    | -       | -       |
| 15 | Torii                         | -       | 24492 | -      | -    | -       | -       |
| 16 | MQTT Flood                    | -       | -     | 77793  | -    | -       | -       |
| 17 | Malformed                     | -       | -     | 3580   | -    | -       | -       |
| 18 | SlowITe                       | -       | -     | 3044   | -    | -       | -       |
| Total|                               | 5651079 |       |        |      |         |         |
IoT-DS2 dataset consists of eighteen attack classes and a normal class. The category column represents the name network traffic class which can be normal or any of the eighteen attack classes. The next five columns represent the dataset used in the development of the IoT-DS2 dataset. The final column represents the number of instances extracted from the BoT-IoT, IoT Network Intrusion, IoT-23, MQTT, and MQTTset datasets. These datasets, available at [96], may be used by researchers to develop and test IoT anomaly detection systems. The extracted datasets were split 80% for training and 20% for testing in the first phase, and then the training dataset was split into 80% for training and 20% for validation using a stratified way. Choosing features is a key stage in deep learning model building. Feature selection is a strategy for improving models that includes detecting and selecting just those features required for improved prediction. In addition to reducing overfitting, the feature selection technique speeds up model training and makes the model less susceptible to test inaccuracies. The recursive feature elimination technique selects 64 features using a random forest algorithm and IoT-DS2 dataset [83]. The feature selection technique was not used on the NSLKDD dataset. The same set of features was utilized in all models and across all datasets.

V. EVALUATION of RESULTS

The proposed LSTM or BiLSTM, or GRU models and convolutional neural network-based LSTM or BiLSTM, or GRU models are validated using accuracy, precision, recall, F1 score, TNR (True Negative Rate), FPR (False Positive Rate), FNR (False Negative Rate), PPV (Positive Predictive Value), NPV (Negative Predictive Value). The formula for these metrics is present in Eq. 11 to Eq. 19.

\[
\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{11}
\]

\[
\text{Precision} = \frac{TP}{(TP + FP)} \tag{12}
\]

\[
\text{Recall} = \text{Sensitivity} = \frac{TP}{(TP + FN)} \tag{13}
\]

\[
F1 \text{ score} = 2 \times \frac{(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \tag{14}
\]

\[
TNR = \text{Specificity} = \frac{TN}{(TN + FP)} \tag{15}
\]

\[
\text{FPR} = \frac{FP}{(FP + TN)} \tag{16}
\]

\[
\text{FNR} = \frac{FN}{(FN + TP)} \tag{17}
\]

\[
\text{PPV} = \frac{TP}{(TP + FP)} \tag{18}
\]

\[
\text{NPV} = \frac{TN}{(TN + FN)} \tag{19}
\]

Seven datasets were used to conduct a multiclass and binary classification experiment using the proposed anomaly detection models. This paper combines the Keras framework with the TensorFlow backend to conduct all the experiments. The experiments were conducted using Google Colab. Three processes comprise a neural network model assessment: training, validation, and testing. The classification method favors the majority class if unequally distributed data sets are employed. We used class weights and SMOTE techniques to handle imbalanced classes in the datasets. First, we used class weight for classes in the training phase. Class weights were calculated based on the number of class instances, so the class with a small number of instances will get a high weight.

The Adam optimizer was used to train each RNN model for 100 epochs on the same batch size. The loss functions of a neural network are certainly the most important factor. The sparse categorical cross-entropy loss function is used in this paper. We utilized three techniques to reduce the overfitting of the model. First, the kernel, bias, and activity regularizers are used at LSTM, BiLSTM, GRU, or CNN layers. The kernel, bias, and activity regularizers use l1-l2 penalty techniques for regularization. Second, we used the activity regularization layer, and third, we used the dropout layer. These three techniques entirely reduce the chance of overfitting the model. Finally, an early stopping approach was used to terminate the model if the validation loss did not decrease over the training phase. The early stopping strategy also minimizes the likelihood of overfitting, which occurs when a model is trained over a large number of epochs. A total of 100 epochs were used to train all RNN models, with a batch size of 128 and patience of 5 iterations for each model. The batch size and the number of epochs were increased and decreased to check for improvement in the model's accuracy. Still, the accuracy of the model was not improved. At each epoch value, the accuracy and loss of each model were computed for both the training and validation sets.

Fig. 11(a) illustrates the loss and Fig. 11(b) accuracy of LSTM, BiLSTM, and GRU models trained and validated on the BoT-IoT dataset. The loss and accuracy of the CNN-based LSTM, BiLSTM, and GRU models using the BoT-IoT dataset during training and validation are represented in Fig. 12(a) and 12(b). The loss function calculates the overall deviation across all tests in the training set. The early stopping strategy will stop the training process if the validation loss does not decrease after a specified number of iterations, hence reducing the over-fitting problem. As illustrated in Fig. 11 and 12, the loss function and the accuracy plot are inversely associated. With 200 and 500 epochs, as well as 10 iterations of patience, the accuracy did not increase. Overfitting of the training data occurs as a result of running a model over a significant number of iterations.

We used multiclass and binary class classification for seven datasets in this paper. Tables 8, 9, and 10 show the multiclass classification of LSTM, BiLSTM, and GRU models using NSLKDD, BoT-IoT, IoT-NI, MQTT, MQTTset, IoT-23, and IoT-DS2 datasets. The outcome of LSTM, BiLSTM, and GRU models using the NSLKDD dataset is presented in Table 8(a). The accuracy of the NSLKDD dataset was measured at 99.67% for LSTM, 99.82% for BiLSTM, and 99.78% for GRU models. Normal,
DoS, and Probe classes achieved a high detection rate while the R2L and U2R detection rates were low. The R2L and U2R attack categories are very rare in the dataset, which is the main reason for the low detection rate of these attack categories. The BiLSTM model achieved a high detection rate among the three models using the NSLKDD dataset. The accuracy of the BoT-IoT dataset is higher than the NSLKDD dataset. Table 8(b) shows the results of LSTM, BiLSTM, and GRU models using the BoT-IoT dataset. Normal, DoS, Scan, and Theft classes of the BoT-IoT dataset achieved at least a 99.50% detection rate. The weighted average of FPR was 0.04%, and FNR was 0.11% for the BiLSTM model. Scan and Theft categories have a high rate of misclassification.

The IoT-NI dataset multiclass classification outcomes are presented in Table 8(c). The accuracy of LSTM, BiLSTM, and GRU models using the IoT-NI dataset is 98.14%, 98.89%, and 98.42%, respectively. The detection rate for the Normal class was relatively high compared to the other classes in the dataset.

The IoT-23 dataset consists of ten classes, and LSTM, BiLSTM, and GRU models results for the IoT-23 dataset are presented in Table 8(d). The accuracy was from 99.71% to 99.83 for the proposed models. The FPR was 0.03%, while the FNR was 0.17% for the IoT-23 dataset for the BiLSTM model. The average detection for the IoT-23 dataset using the BiLSTM model measured 99.83%. The MQTT and MQTTset datasets had the highest detection rate among all datasets tested in this paper for LSTM, BiLSTM, and GRU models. The results of LSTM, BiLSTM, and GRU models for MQTT and MQTTset datasets are presented in Tables 8(e) and 8(d). The accuracy for MQTT and MQTTset datasets was 99.90% to 99.95%, and the detection rate was 99.90% to 99.95%. The average FPR was 0.0042%, and FNR was 0.06% for the MQTT dataset using the BiLSTM model. The average FPR was 0.062%, and the FNR was 0.06% using the BiLSTM model on the MQTTset dataset.

The IoT-DS2 dataset consists of 19 classes. The Accuracy of LSTM, BiLSTM, and GRU models using the IoT-DS2 dataset were 99.31%, 99.48%, and 99.32%, respectively. The BiLSTM model achieved a detection rate of 99.40%, which is higher than the LSTM and GRU models. The FPR for the IoT-DS2 dataset using the BiLSTM model was 0.06%, while the FNR was 0.51%. The detection rate for the Normal class was at least 99% in all three models. MITM, C&C, Heartbeat, and Malformed Data attack categories have less than a 98% detection rate. Sensitivity, specificity, PPV, and NPV results for multiclass classification of the NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, and MQTTset datasets using LSTM, BiLSTM, and GRU models are presented in Table 10.
A convolutional neural network requires significantly less preprocessing than other deep learning classification techniques, making it a more efficient classification approach. We provide detailed performance metrics for each kind of network anomaly and comparisons to other current deep learning-based systems to assess the CNN model’s ability to identify different network attack characteristics [83]. The advantages of a convolutional neural network for anomaly detection issues are fully realized when the networks are applied to image recognition tasks. Using appropriate filters, a convolutional neural network can effectively capture spatial and temporal connectivity.
anomaly detection problems. A convolutional neural network is good for prediction because they learn input characteristics without missing important information. We used a convolutional 1D neural network in the first hidden layer to learn the network features. The convolutional 1D neural network layer is followed by two LSTM or BiLSTM or GRU layers hidden layers. The proposed convolutional 1D neural network layer LSTM, BiLSTM, and GRU are evaluated using NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, MQTTset, and IoT-DS2 datasets. The evaluation results are presented in Tables 11 and 12.

### TABLE 9. Accuracy, Precision, Recall, F1 Score of IoT-DS2 dataset multiclass classification using LSTM, BiLSTM, and GRU models.

| Model       | LSTM            | BiLSTM           | GRU             |
|-------------|-----------------|------------------|-----------------|
| Class       | Accuracy Precision Recall F1 Score | Accuracy Precision Recall F1 Score | Accuracy Precision Recall F1 Score |
| Normal      | 99.31 99.03 99.51 99.27 | 99.48 99.50 99.58 99.54 | 99.32 99.50 99.30 99.40 |
| DDoS        | 99.31 99.84 99.86 99.85 | 99.48 99.65 99.90 99.77 | 99.32 99.60 99.89 99.75 |
| DoS         | 99.31 99.63 97.63 98.62 | 99.48 99.48 95.27 97.33 | 99.32 99.18 97.77 98.47 |
| MITM        | 99.31 96.25 87.48 91.66 | 99.48 94.84 94.06 94.45 | 99.32 96.77 94.82 95.78 |
| Mirai       | 99.31 98.24 99.01 98.62 | 99.48 99.10 97.75 98.42 | 99.32 99.08 98.64 98.86 |
| MQTT-BF     | 99.31 99.94 98.20 99.06 | 99.48 98.98 99.63 99.30 | 99.32 99.50 98.02 98.76 |
| Sparta      | 99.31 98.14 99.50 98.81 | 99.48 99.20 98.83 99.51 | 99.32 97.98 99.47 98.72 |
| Theft       | 99.31 98.92 96.68 97.79 | 99.48 98.44 97.24 97.84 | 99.32 90.93 95.06 92.95 |
| Attack      | 99.31 99.70 99.46 99.58 | 99.48 99.49 97.96 99.62 | 99.32 99.29 97.87 98.58 |
| C&C         | 99.31 97.32 89.26 93.12 | 99.48 97.99 96.49 97.24 | 99.32 98.32 92.42 95.28 |
| File Download | 99.31 98.98 97.88 98.43 | 99.48 98.73 97.13 97.92 | 99.32 98.72 91.20 94.81 |
| Heartbeat   | 99.31 90.39 98.51 94.27 | 99.48 97.02 98.43 97.72 | 99.32 91.48 97.11 94.21 |
| Okiru       | 99.31 99.99 99.99 99.99 | 99.48 99.99 99.99 99.99 | 99.32 99.99 99.99 99.99 |
| OS Scan     | 99.31 98.83 98.08 98.95 | 99.48 99.17 99.53 99.35 | 99.32 99.07 99.51 99.29 |
| Port Scan   | 99.31 99.99 99.99 99.99 | 99.48 99.99 99.98 99.99 | 99.32 99.99 99.98 99.99 |
| Torii       | 99.31 99.99 99.94 99.97 | 99.48 99.99 99.94 99.97 | 99.32 99.72 99.92 99.82 |
| MQTT Flood  | 99.31 99.99 99.99 99.99 | 99.48 99.99 99.99 99.99 | 99.32 99.99 99.99 99.99 |
| Malformed Data | 99.31 97.13 97.13 97.13 | 99.48 97.04 94.25 95.63 | 99.32 92.70 90.29 91.48 |
| SlowTe      | 98.76 99.83 99.99 99.92 | 99.48 98.17 99.25 98.71 | 99.32 96.67 98.47 98.05 |

The NSLKDD dataset evaluation results using CNN-based LSTM, BiLSTM, and GRU are presented in Table 11(a). We have observed an increase in the detection rate for all proposed models on the NSLKDD dataset. CNNBiLSTM model achieved a high accuracy and detection rate, while the LSTM model had the lowest accuracy and detection rate for the NSLKDD dataset. The Bot-IoT dataset evaluation outcome is presented in Table 11(b). All models developed for the Bot-IoT dataset demonstrate increased detection rates and decreased FPR and FNR. The IoT-NI dataset evaluation results are presented in Table 11(c). The average detection rate for IoT-NI was 98.14% when only LSTM was used, as seen in Table 8(c), but when LSTM was combined with CNN, the average detection rate was increased to 98.35% with a reduced number of hidden layers. The average detection rate for CNNBiLSTM and CNNGRU models was increased to 98.90% and 98.42%, respectively, as seen in Table 11(c).

The IoT-23 dataset evaluation result is presented in Table 11(d). The CNNBiLSTM model achieved a high detection rate for the IoT-23 dataset. The detection rate for all suggested models utilizing the IoT-23 dataset has improved. As shown in Table 11(d), the average detection rate for CNNBiLSTM and CNNGRU models has been enhanced to 99.83 and 99.86%, respectively. Multiclass classification of MQTT datasets is shown in 11(e), and MQTTset dataset multiclass classification is presented in Table 11(f). MQTT and MQTTset datasets achieved a high detection rate compared to other datasets used in this paper, as seen in Tables 11(e) and 11(f). The FPR for the MQTT dataset was 0.002%, while the FNR was 0.04% using the
CNNLSTM model. FPR was 0.0016% for the MQTT dataset, while FNR was 0.03% when the CNNBiLSTM model was used. FPR was 0.0026%, while FNR was 0.046% for the MQTT dataset when the CNNGRU model was used. Only 15, 12, and 12 instances of the MQTTset dataset were misclassified using the CNNLSTM, CNNBiLSTM, and CNNGRU models. The IoT-DS2 dataset has 18 attack categories, and the IoT-DS2 dataset assessment results are summarized in Table 12. The detection rates were 99.50%, 99.57%, and 99.48%, respectively, when CNNLSTM, CNNBiLSTM, and CNNGRU models were used. Sensitivity, specificity, PPV, and NPV results for multiclass classification of the NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, and MQTTset datasets using CNNLSTM, CNNBiLSTM, and CNNGRU models are presented in Table 13.

### Table 11. Accuracy, Precision, Recall, F1 Score of NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, and MQTTset datasets multiclass classification using CNNLSTM, CNNBiLSTM, and CNNGRU models.

| Class       | Accuracy | Precision | Recall | F1 Score | Accuracy | Precision | Recall | F1 Score | Accuracy | Precision | Recall | F1 Score |
|-------------|----------|-----------|--------|----------|----------|-----------|--------|----------|----------|-----------|--------|----------|
| Normal      | 99.81    | 99.91     | 99.91  | 99.91    | 99.88    | 99.90     | 99.96  | 99.93    | 99.85    | 99.87     | 99.96  | 99.92    |
| DoS         | 99.81    | 99.83     | 99.96  | 99.90    | 99.88    | 99.93     | 99.96  | 99.95    | 99.85    | 99.89     | 99.95  | 99.92    |
| Probe       | 99.81    | 98.47     | 96.76  | 97.61    | 99.88    | 99.24     | 97.75  | 99.49    | 99.85    | 99.75     | 98.51  | 99.13    |
| R2L         | 99.81    | 95.96     | 90.91  | 93.37    | 99.88    | 97.99     | 94.04  | 93.30    | 99.85    | 94.61     | 92.34  | 93.46    |
| U2R         | 99.81    | 97.18     | 97.18  | 97.18    | 99.88    | 98.27     | 97.42  | 97.84    | 99.85    | 99.11     | 93.31  | 96.12    |
| MQTT-BF     | 99.96    | 99.18     | 99.99  | 99.99    | 99.96    | 99.96     | 99.99  | 99.99    | 99.96    | 99.96     | 99.99  | 99.99    |
| Scan-A      | 99.96    | 99.96     | 99.96  | 99.96    | 99.96    | 99.96     | 99.96  | 99.96    | 99.96    | 99.96     | 99.96  | 99.96    |
| Scan-U      | 99.96    | 99.96     | 99.96  | 99.96    | 99.96    | 99.96     | 99.96  | 99.96    | 99.96    | 99.96     | 99.96  | 99.96    |
| Sparta      | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    |
| MQTT Flood  | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    |
| DoS         | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    |
| MQTT-BF     | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    |
| Malformed Data | 99.99   | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    |
| SlowITe     | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    | 99.99    | 99.99     | 99.99  | 99.99    |
A lightweight binary classification model is proposed and developed based on a single recurrent neural network-based hidden layer. Three binary classification-based models were created using a single hidden layer from the LSTM, BiLSTM, and GRU networks. They were tested using NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, and MQTTset datasets. First, we convert the whole dataset to anomalous occurrences in the NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, and MQTTset datasets. Next, we expand the binary classification model to classify normal and individual anomalous classes of the IoT-DS2 dataset. The IoT-DS2 dataset was divided into eighteen subsets, with each subset consisting of a normal data class and an anomalous data class. Each subset was used to evaluate LSTM, BiLSTM, and GRU models. The accuracy, precision, recall, and F1 score of the IoT-DS2 dataset normal and individual anomalous classes using LSTM, BiLSTM, and GRU models are presented in Fig. 13, and for the testing set are presented in Fig 14. Additionally, we compare the performance of binary classification using the receiver operating characteristic area under the curve (ROC AUC). We plot the ROC curves for the validation and testing sets for the NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, and MQTTset datasets. ROC curve for the validation set of NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, and MQTTset, datasets binary classification using BiLSTM model are presented in Fig. 13, and for the testing set are presented in Fig 14.

The findings of the binary classification evaluation using the BiLSTM model are summarized in Table 15. A detection rate of 99.42% was achieved with the MQTTset dataset using the BiLSTM model. IoT-DS2 has an average detection rate of 99.81%, much higher than the LSTM model but lower than the detection rate of the BiLSTM model. The binary classification model based on a single recurrent neural network-based hidden layer accurately detected normal and anomalous occurrences in the NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, MQTTset, and IoT-DS2 datasets.
IoT-23, MQTT, and MQTTset datasets. The Borderline SMOTE algorithm was used to generate new synthetic samples. The random state is used to control the algorithm's randomization. K is the number of neighbors used in the borderline SMOTE algorithm to calculate the average distance to minority samples. We tested a range of K values, but K=6 to 10 was determined as an optimal number of neighbors for calculating the average distance to minority samples. New synthetic samples were created using the borderline-1 algorithm. We used CNN-based LSTM, BiLSTM, and GRU models to evaluate the performance of borderline SMOTE. The borderline SMOTE is used exclusively to ensure that the training set is balanced. The multiclass classification using SMOTE for class balancing and CNNLSTM, CNNBiLSTM, and CNNGRU models for classification are presented in Table 15.

### TABLE 13. Sensitivity, Specificity, PPV, NPV of NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, and MQTTset datasets multiclass classification using CNNLSTM, CNNBiLSTM, and CNNGRU models.

| Dataset | Model | Model | Sensitivity | Specificity | PPV | NPV | Sensitivity | Specificity | PPV | NPV | Sensitivity | Specificity | PPV | NPV |
|---------|-------|-------|-------------|-------------|-----|-----|-------------|-------------|-----|-----|-------------|-------------|-----|-----|
| NSLKDD  | CNNLSTM | 99.90 | 99.95 | 99.90 | 99.94 | 99.92 | 99.96 | 99.92 | 99.97 | 99.88 | 99.93 | 99.88 | 99.96 |
| BoT-IoT | CNNLSTM | 99.93 | 99.98 | 99.93 | 99.98 | 99.96 | 99.99 | 99.96 | 99.99 | 99.95 | 99.98 | 99.95 | 99.99 |
| IoT-NI  | CNNLSTM | 98.35 | 97.88 | 98.35 | 99.22 | 99.18 | 99.34 | 99.18 | 99.46 | 98.76 | 98.97 | 98.77 | 98.88 |
| IoT-23  | CNNLSTM | 98.83 | 99.96 | 98.83 | 99.98 | 98.87 | 99.97 | 98.79 | 99.89 | 98.86 | 99.98 | 98.86 | 99.98 |
| MQTT    | CNNLSTM | 99.96 | 100.00 | 99.96 | 99.95 | 99.97 | 100.00 | 99.97 | 100.00 | 99.95 | 100.00 | 99.95 | 99.94 |
| MQTTset | CNNLSTM | 99.99 | 100.00 | 99.99 | 100.00 | 99.99 | 100.00 | 99.99 | 100.00 | 99.99 | 100.00 | 99.99 | 100.00 |
| IoT-DS2 | CNNLSTM | 99.45 | 99.94 | 99.45 | 99.96 | 99.57 | 99.96 | 99.57 | 99.97 | 99.52 | 99.96 | 99.52 | 99.97 |

### TABLE 14. NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, MQTTset, and IoT-DS2 datasets binary classification using LSTM model.

| Dataset | Class       | Accuracy | Precision | Recall | F1 score | TNR  | FPR  | FNR  |
|---------|-------------|----------|-----------|--------|----------|------|------|------|
| NSLKDD  | Normal      | 99.91    | 99.89     | 99.92  | 99.90    | 99.97| 0.03 | 0.11 |
|          | Anomaly     | 99.97    | 99.97     | 99.97  | 99.97    | 99.89| 0.11 | 0.03 |
| BoT-IoT | Normal      | 99.90    | 99.52     | 99.45  | 99.49    | 99.97| 0.03 | 0.55 |
|          | Anomaly     | 99.97    | 99.97     | 99.97  | 99.97    | 99.45| 0.55 | 0.03 |
| IoT-NI  | Normal      | 99.10    | 96.36     | 97.02  | 96.69    | 99.72| 0.28 | 2.98 |
|          | Anomaly     | 99.77    | 98.72     | 99.74  | 99.75    | 97.02| 0.28 | 2.98 |
| IoT-23  | Normal      | 99.80    | 99.74     | 99.93  | 99.84    | 99.89| 0.11 | 0.07 |
|          | Anomaly     | 99.97    | 99.89     | 99.93  | 99.93    | 99.93| 0.07 | 0.11 |
| MQTT    | Normal      | 99.91    | 99.21     | 99.81  | 99.51    | 99.96| 0.04 | 0.19 |
|          | Anomaly     | 99.99    | 99.96     | 99.98  | 99.81    | 99.81| 0.19 | 0.04 |
| MQTTset | Normal      | 99.96    | 99.99     | 99.99  | 99.99    | 99.95| 0.05 | 0.01 |
|          | Anomaly     | 99.94    | 99.95     | 99.94  | 99.94    | 99.99| 0.01 | 0.05 |
| IoT-DS2 | Normal      | 99.42    | 98.96     | 99.11  | 99.03    | 99.82| 0.18 | 0.89 |
|          | Anomaly     | 99.84    | 99.82     | 99.83  | 99.83    | 99.11| 0.89 | 0.18 |

### TABLE 15. NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, MQTTset, and IoT-DS2 datasets binary classification using BiLSTM model.

| Dataset | Class       | Accuracy | Precision | Recall | F1 score | TNR  | FPR  | FNR  |
|---------|-------------|----------|-----------|--------|----------|------|------|------|
| NSLKDD  | Normal      | 99.92    | 99.88     | 99.92  | 99.90    | 99.97| 0.03 | 0.08 |
|          | Anomaly     | 99.98    | 99.98     | 99.98  | 99.98    | 99.92| 0.08 | 0.03 |
| BoT-IoT | Normal      | 99.96    | 99.76     | 99.82  | 99.80    | 99.99| 0.01 | 0.17 |
|          | Anomaly     | 99.99    | 99.99     | 99.99  | 99.99    | 99.83| 0.17 | 0.01 |
| IoT-NI  | Normal      | 99.25    | 96.84     | 97.80  | 97.32    | 99.76| 0.24 | 2.20 |
|          | Anomaly     | 99.83    | 99.76     | 99.76  | 99.80    | 97.80| 0.20 | 0.24 |
| IoT-23  | Normal      | 99.70    | 99.63     | 99.75  | 99.66    | 99.84| 0.16 | 0.29 |
|          | Anomaly     | 99.88    | 99.85     | 99.85  | 99.87    | 99.71| 0.29 | 0.16 |
| MQTT    | Normal      | 99.88    | 98.74     | 99.84  | 99.29    | 99.94| 0.06 | 0.16 |
|          | Anomaly     | 99.99    | 99.99     | 99.99  | 99.99    | 99.94| 0.16 | 0.06 |
| MQTTset | Normal      | 99.96    | 99.99     | 99.99  | 99.99    | 99.95| 0.05 | 0.01 |
|          | Anomaly     | 99.98    | 99.95     | 99.95  | 99.95    | 99.99| 0.01 | 0.05 |
| IoT-DS2 | Normal      | 99.65    | 99.31     | 99.40  | 99.36    | 99.88| 0.12 | 0.60 |
|          | Anomaly     | 99.89    | 99.88     | 99.99  | 99.94    | 99.40| 0.60 | 0.12 |
TABLE 16. NSLKDD, Bot-IoT, IoT-NI, IoT-23, MQTT, MQTTset, and IoT-DS2 datasets binary classification using GRU model.

| Dataset   | Class  | Accuracy | Precision | Recall | F1 score | TNR  | FPR  | FNR  |
|-----------|--------|----------|-----------|--------|----------|------|------|------|
| NSLKDD    | Normal | 99.91    | 99.89     | 99.89  | 99.97    | 0.03 | 0.11 |      |
|           | Anomaly| 99.97    | 99.97     | 99.97  | 99.97    | 0.11 | 0.03 |      |
| BoT-IoT   | Normal | 99.94    | 99.71     | 99.66  | 99.98    | 0.02 | 0.39 |      |
|           | Anomaly| 99.98    | 99.98     | 99.98  | 99.98    | 0.39 | 0.02 |      |
| IoT-NI    | Normal | 99.25    | 96.84     | 97.80  | 99.76    | 0.24 | 2.20 |      |
|           | Anomaly| 99.83    | 97.76     | 99.80  | 97.80    | 2.20 | 0.24 |      |
| IoT-23    | Normal | 99.23    | 99.36     | 99.36  | 99.73    | 0.27 | 0.64 |      |
|           | Anomaly| 99.72    | 99.73     | 99.72  | 99.36    | 0.64 | 0.27 |      |
| MQTT      | Normal | 99.88    | 98.74     | 99.84  | 99.94    | 0.16 | 0.16 |      |
|           | Anomaly| 99.99    | 99.94     | 99.96  | 99.84    | 0.16 | 0.16 |      |
| MQTTset   | Normal | 99.96    | 99.99     | 99.99  | 99.95    | 0.05 | 0.01 |      |
|           | Anomaly| 99.96    | 99.95     | 99.95  | 99.99    | 0.01 | 0.05 |      |
| IoT-DS2   | Normal | 99.45    | 99.15     | 99.07  | 99.85    | 0.15 | 1.01 |      |
|           | Anomaly| 99.82    | 99.85     | 99.84  | 98.99    | 1.01 | 0.15 |      |

The NSLKDD dataset Prob, U2L, and R2L classes detection rates have been improved as shown in Table 15(a). We select one million instances randomly from the BoT-IoT dataset's DoS and Scan classes and then use borderline SMOTE to balance other classes of the BoT-IoT dataset. Table 15(b) shows the detection rate for the Normal and Theft classes has been enhanced. Four classes were balanced in the IoT-NI dataset. As presented in Table 15(c), the detection rates for the Normal, DoS, MITM, and Scan classes have been improved for the IoT-NI dataset. We select one million instances randomly from the IoT-23 dataset's Normal, Attack, Port Scan, DDoS, and Okiru classes. Mirai, File Download, Heartbeat, C&C, and Torii classes were balanced.

The IoT-23 dataset has improved the detection rate of minority classes, as seen in Table 15. (d). We randomly select one million MQTT-BF instances and one million Sparta instances from the MQTT dataset. The MQTT dataset achieved a high detection rate when class weight was used to handle imbalance classes in the MQTT dataset. Each model enhanced the overall detection rate in the MQTT dataset. All anomalous classes were in the minority in the MQTTset dataset. Class weight was employed to address imbalance classes in the MQTTset dataset, resulting in a high detection rate, but when border SMOTE was employed on the MQTT dataset, each model improved the overall detection rate. The border SMOTE technique performs better than class weight, but the border SMOTE technique requires more computing resources than class weight. Overfitting of the LSTM, BiLSTM and GRU models was evaluated using drop layer and early stopping approaches. Furthermore, we perform a 5-fold cross-validation on the LSTM, BiLSTM, and GRU models to evaluate further overfitting. Through the use of 5-fold cross-validation, identical results were obtained, demonstrating the consistency of the proposed model.
VI DISCUSSION AND COMPARISON OF RESULTS

The proposed model’s results are compared to previous research papers in this section. We propose and implement binary and multiclass classification models based on LSTM, BiLSTM, GRU, and multiclass classification models based on CNNLSTM, CNNBiLSTM, and CNNGRU. These models were evaluated using NSLKDD, BoT-IoT, IoT-NI, IoT-23, MQTT, MQTTset, and IoT-DS2 datasets. Initially, we implement LSTM, BiLSTM, GRU based models consisting of 4 hidden layers. Our proposed models performed much better at detecting anomalies in IoT networks in binary and multiclass classification. We investigated the prospect of solving anomaly detection in IoT networks by deploying a recurrent neural network and integrating it with a convolutional neural network. The proposed model also uses kernel, bias, and activity regularizers at the recurrent layers. These regularizers apply penalties on the kernel, bias, and layer output.

The activation layer used the LeakyReLU activation function. The pace of learning has been greatly enhanced when LeakyReLU is used. A model normalization layer may often aid in accelerating and stabilizing the learning process by decreasing the error rate in the model being learned. Layer normalization can stabilize the hidden state dynamics of the recurrent network. We employed the activity regularization layer and the dropout layer to decrease the possibility of overfitting. We propose and build a binary classification model based on a single recurrent neural network-based hidden layer. Table 18 compares binary classification techniques used in deep learning with proposed anomaly detection models. The detection rate was very high for IoT-23, MQTT, and MQTTset datasets. Given that these datasets are recently released datasets for the IoT network, there is not many research papers available in which authors constructed a deep learning model utilizing these datasets. Since the BoT-IoT dataset is the most referenced IoT dataset, we also used the BoT-IoT dataset to compare our proposed models to previously published IoT models. We also used the NSLKDD dataset to compare the evaluation results of our proposed binary classification model for anomaly detection in a generic network setting.

Anomaly detection models based on RNN were evaluated using the NSLKDD dataset by Yin et al. [38]. In their experiment, the model achieves a greater accuracy when there are 80 hidden nodes and a learning rate of 0.1. When assigned 100 epochs, RNN-IDS has a high detection rate of 83.28%. Liu et al. [37] suggested the features of long and short sessions and developed a neural network based on CNN and LSTM models to extract the variations between normal and abnormal models. The results demonstrate that not only can the suggested quantitative model and enhanced algorithm effectively prevent camouflaging identity information, but they can also improve computing efficiency and the accuracy of small subset anomaly detection. Their model achieved high detection rate.
Li et al. [39] use LSTM and GRU with a variable number of hidden layers, and they evaluate the recently suggested Broad Learning System (BLS) and its variants. Pseudoinverse weight adjustment replaces back-propagation in BLS models, with fewer hidden layers. In the event of incremental learning, weights can be dynamically updated. Due to the additional data points, they could better optimize weights for big databases. The accuracy and F1 score were very low for
LSTM and GRU models. A BiLSTM-based intrusion detection system is proposed by Imrana et al. [9]. BiLSTM and LSTM models performance was compared using the NSLKDD dataset. The LSTM model has high false alarms than the BiLSTM model. Biswas et al. [57] proposed a method for distinguishing malicious Botnet traffic from legitimate traffic by utilizing modern deep learning techniques such as ANN and GRU models. GRU model achieved a better detection rate compared to the ANN model. Liu et al. propose leveraging different aspects of each type of communication from three perspectives: anomaly discovery, clustering, and classification. Additionally, they offer a tailored loss function to adjust for traffic data that is distributed irregularly. DNN, CNN, and RNN models achieved a low detection rate compared to their proposed MTDL model.

| Article            | Year | Model   | Dataset | Accuracy | Precision | Recall | F1 Score | FPR  |
|--------------------|------|---------|---------|----------|-----------|--------|----------|------|
| Yin et al. [19]    | 2017 | RNN     | NSLKDD  | 71.52    | 86.83     | 83.39  | 83.39    | 13.2 |
| Arivud et al. [33] | 2019 | CNN     | NSLKDD  | 99.67    | 99.69     | -      | -        | 0.57 |
| Liu et al. [37]    | 2019 | CNN-LSTM| NSLKDD  | 98.90    | -         | -      | -        | -    |
| Li et al. [39]     | 2019 | LSTM    | NSLKDD  | 82.78    | -         | 83.34  | -        | -    |
| Li et al. [39]     | 2019 | GRU     | NSLKDD  | 82.87    | -         | 83.05  | -        | -    |
| Imrana et al. [9]  | 2021 | LSTM    | NSLKDD  | 89.81    | 97.75     | 84.03  | 90.38    | 2.55 |
| Imrana et al. [9]  | 2021 | BiLSTM  | NSLKDD  | 94.26    | 99.05     | 90.79  | 94.75    | 1.15 |
| Biswas et al. [57] | 2021 | ANN     | NSLKDD  | 98.39    | -         | -      | -        | -    |
| Biswas et al. [57] | 2021 | GRU     | NSLKDD  | 99.14    | -         | -      | -        | -    |
| Liu et al. [98]    | 2021 | DNN     | NSLKDD  | 93.20    | -         | -      | 91.10    | 0.09 |
| Liu et al. [98]    | 2021 | CNN     | NSLKDD  | 92.70    | -         | -      | 89.20    | 0.10 |
| Liu et al. [98]    | 2021 | RNN     | NSLKDD  | 91.90    | -         | -      | 88.10    | 0.12 |
| Liu et al. [98]    | 2021 | MTDL    | NSLKDD  | 95.50    | -         | -      | 94.10    | 0.079|
| Moizuddin et al. [99]| 2022| GMGWO   | NSLKDD  | 99.84    | 99.68     | 99.94  | 99.81    | -    |
| Ferrage et al. [32]| 2019 | RNN-BPTT| BoT-IoT | 98.20    | -         | -      | -        | 1.28 |
| Susilo et al. [46] | 2020 | MLP     | BoT-IoT | 79.01    | -         | -      | -        | -    |
| Susilo et al. [46] | 2020 | CNN     | BoT-IoT | 91.27    | -         | -      | -        | -    |
| Ferrag et al. [48] | 2020 | RNN     | BoT-IoT | 98.31    | 97.50     | -      | -        | 1.1  |
| Biswas et al. [57] | 2021 | ANN     | BoT-IoT | 97.23    | -         | -      | -        | -    |
| Biswas et al. [57] | 2021 | GRU     | BoT-IoT | 99.76    | -         | -      | -        | -    |
| Liu et al. [98]    | 2021 | DNN     | BoT-IoT | 99.80    | -         | -      | 97.80    | 0.33 |
| Liu et al. [98]    | 2021 | CNN     | BoT-IoT | 99.10    | -         | -      | 98.90    | 0.30 |
| Liu et al. [98]    | 2021 | RNN     | BoT-IoT | 99.90    | -         | -      | 99.70    | 0.30 |
| Liu et al. [98]    | 2021 | MTDL    | BoT-IoT | 99.90    | -         | -      | 99.80    | 0.21 |
| Ge et al. [47]     | 2020 | DNN     | BoT-IoT | 99.90    | -         | -      | -        | -    |
| Proposed Models    | 2021 | LSTM    | NSLKDD  | 99.91    | 99.96     | 99.96  | 99.96    | 0.09 |
|                   | 2021 | GRU     | NSLKDD  | 99.91    | 99.96     | 99.96  | 99.96    | 0.09 |
|                   | 2021 | BiLSTM  | NSLKDD  | 99.92    | 99.96     | 99.96  | 99.96    | 0.07 |
|                   | 2021 | LSTM    | BoT-IoT | 99.90    | 99.95     | 99.95  | 99.95    | 0.52 |
|                   | 2021 | GRU     | BoT-IoT | 99.93    | 99.97     | 99.97  | 99.97    | 0.37 |
|                   | 2021 | BiLSTM  | BoT-IoT | 99.96    | 99.98     | 99.98  | 99.98    | 0.16 |

Ferrag et al. [32] came up with a new way to protect Smart Grids from hackers. They used deep learning and blockchain technology to build a new framework for Smart Grids. They used a recurrent neural network to monitor the energy network for network threats and fraudulent transactions. Their model has a high accuracy of 98.20% when applied to the BoT-IoT dataset. Ferrag et al. [48] use a binary and multiclass classification approach to study seven deep learning models. RNN models with varying learning rates and hidden nodes were tested for accuracy and training time on the BoT-IoT dataset. The accuracy and training time of unsupervised models with various learning rates and hidden nodes were also compared using the BoT-IoT dataset. The RNN model achieved a high accuracy of 98.31%. Susilo et al. [46] created a deep-learning method for identifying denial-of-service (DoS) cyberattacks in an IoT network. They discovered that a deep-learning model might significantly improve accuracy, allowing for the most effective prevention of threats on an IoT network. The accuracy of their suggested model, which was based on the CNN model, was very poor, as seen in Table 18.
Advances in communication and information technology have made it possible to exchange an ever-increasing quantity of data over the internet, resulting in new uses for Internet services. Biswas et al. [57] suggested a method for distinguishing malicious Botnet traffic from legitimate traffic by using novel deep learning techniques such as ANN, GRU, and LSTM models. Testing reveals that the categorization accuracy is 99.76%, which is better than all prior research, as indicated by the author. The proposed LSTM, BiLSTM, and GRU models outperformed previously published anomaly detection models in terms of accuracy, precision, recall, and F1 score using the BoT-IoT dataset. We used a recurrent neural network to construct and implement two anomaly detection models for multiclass classification in IoT networks.

The LSTM, BiLSTM, and GRU models were built using the same structure. Fig. 8 represents a layered view of the proposed LSTM, BiLSTM, or GRU models. The first implementation is made up of four hidden layers. The hidden layers are either LSTM, BiLSTM, or GRU layers. Convolutional neural network advantages can become more apparent when anomaly detection difficulties are turned into image recognition problems. A convolutional neural network is very good at capturing spatial and temporal correlations that are important for intrusion detection. We then used a convolutional and recurrent neural network to construct and develop a model. In the second approach, three hidden layers are used to get the desired result. The first hidden layer uses a convolutional neural network to extract feature information and correlation from the input features. We utilized an average pooling layer to decrease the number of features by half. Next, there are two hidden layers of the recurrent neural network, and the hidden layers are composed of either LSTM, BiLSTM, or GRU layers.

The convolutional neural network layer replaces two recurrent neural network layers and reduces the number of features by half. The multiclass classification was performed in this paper to analyze data from the NSLKDD, BoT-IoT, IoT-NI, MQTT, MQTTset, IoT-23, and IoT-DS2 datasets. LSTM, BiLSTM, and GRU models using these datasets for multiclass classification are presented in Tables 8, 9, and 10, while CNN-based LSTM, BiLSTM, and GRU models using these datasets for multiclass classification are shown in Tables 11, 12, and 13. Table 19 compares the assessment results of our proposed multiclass classification models for anomaly detection in a generic network and IoT networks scenario using the NSLKDD and BoT-IoT datasets. Yin et al. [19], Wu et al. [25], Naseer et al. [26], Ding et al. [27], Chouhan et al. [34], Imrana et al. [9], Liu et al. [51], Sethi et al. [61], and Vinayakumar [35] used the NSLKDD dataset to evaluate their model, but these proposed techniques achieved very low accuracy, as shown in Table 19. The model proposed by Sahu et al. [100] and Moizuddin et al. [99] achieved reasonably high accuracy and detection rate compared to other model models.

| Article          | Year | Model | Dataset | Accuracy | Precision | Recall | F1 Score | FPR  |
|------------------|------|-------|---------|----------|-----------|--------|----------|------|
| Yin et al. [19]  | 2017 | RNN   | NSLKDD  | 81.29    | 83.06     | 81.29  | 79.25    | 13.00|
| Wu et al. [25]   | 2018 | CNN   | NSLKDD  | 79.48    | 80.91     | 79.48  | 77.24    | 14.11|
| Naseer et al. [26]| 2018 | LSTM  | NSLKDD  | 89.00    | -         | -      | -        | -    |
| Naseer et al. [26]| 2018 | CNN   | NSLKDD  | 85.00    | -         | -      | -        | -    |
| Ding et al. [27] | 2018 | LSTM  | NSLKDD  | 73.18    | -         | -      | -        | -    |
| Ding et al. [27] | 2018 | CNN   | NSLKDD  | 80.13    | -         | -      | -        | -    |
| Xu et al. [24]   | 2018 | BGRU  | NSLKDD  | 99.24    | 99.31     | -      | -        | 0.84 |
| Chouhan et al. [34]| 2019 | CNN   | NSLKDD  | 89.41    | -         | -      | -        | -    |
| Imrana et al. [9] | 2021 | BiLSTM| NSLKDD  | 91.36    | 92.81     | 91.36  | 91.67    | 4.03 |
| Liu et al. [51]  | 2021 | DSSTE+LSTM | NSLKDD | 81.78    | 82.71     | 81.77  | 80.98    | -    |
| Sethi et al. [61]| 2021 | Reinforcement | NSLKDD | -        | 96.50     | 99.10  | 98.80    | 0.0124|
| Vinayakumar et al. [35]| 2018 | DNN   | NSLKDD  | 78.50    | 81.00     | 78.50  | 76.50    | -    |
| Sahu et al. [100]| 2022 | LSTM  | NSLKDD  | 98.93    | 98.90     | 99.10  | 99.00    | -    |
| Moizuddin et al. [99]| 2022 | GMGWO | NSLKDD  | 99.90    | 99.06     | 99.79  | 99.41    | -    |
| Ge et al. [41]   | 2019 | DNN   | BoT-IoT | 99.03    | 99.06     | 99.03  | 99.04    | 0.73 |
| Ge et al. [47]   | 2020 | DNN   | BoT-IoT | 99.79    | 99.80     | 99.78  | 99.79    | 0.11 |
| Proposed Models  | 2021 | LSTM  | NSLKDD  | 99.91    | 99.94     | 99.92  | 99.93    | 0.06 |
|                 | 2021 | GRU   | NSLKDD  | 99.92    | 99.93     | 99.92  | 99.92    | 0.06 |
|                 | 2021 | BiLSTM| NSLKDD  | 99.94    | 99.96     | 99.94  | 99.95    | 0.05 |
|                 | 2021 | LSTM  | BoT-IoT | 99.94    | 99.97     | 99.94  | 99.95    | 0.03 |
|                 | 2021 | GRU   | BoT-IoT | 99.96    | 99.96     | 99.93  | 99.94    | 0.02 |
|                 | 2021 | BiLSTM| BoT-IoT | 99.97    | 99.97     | 99.95  | 99.96    | 0.02 |

Xu et al. [24] investigate a multilayer perception and gated recurrent units-based intrusion-detection model utilizing the KDD99 and NSLKDD datasets. Their model achieved a maximum accuracy of 99.24% for the NSLKDD dataset.
Numerous issues emerge as a result of the fact that malicious cyberattacks are constantly evolving and happening in extremely high numbers, demanding a scalable solution. Due to malware's dynamic nature and ever-changing attack methodologies, publicly accessible malware datasets must be regularly updated and benchmarked. DNNs and other conventional machine learning classifiers have been tested on various publicly accessible malware datasets by Vinayakumar et al. [35]. Their proposed model attained a maximum accuracy of 92.90% using four hidden layers for the KDD99 dataset and 78.50% using five hidden layers for the NSLKDD dataset. The security of an Internet of Things network is fundamentally linked to the security of the supporting computer and communication infrastructure.

Ge et al. [44] propose an intrusion detection technique for IoT networks, classifying traffic flow using a deep learning technique. They use the BoT-IoT dataset to obtain generic features from packet data. Their model has a high level of accuracy, with a score of 98.09%. Recently, Ge et al. [47] also proposed a model based on a feed-forward neural network with embedding layers for encoding high-dimensional categorical characteristics for multiclass classification. They also used transfer learning to encode high-dimensional category characteristics to construct a binary classifier based on another feed-forward neural network model. Their model reached a 99.79% accuracy rate. They used the source port in the feature set. All the attacks were generated from specific source ports. Due to this, there is a possibility that their model overfits the training data; resulting in high accuracy and detection rate. Anomaly detection methods such as the LSTM, BiLSTM, and GRU models were developed to outperform existing anomaly detection models in the usual criteria of accuracy, precision, recall, and F1-score.

The classification algorithm is biased towards the majority class when an imbalanced dataset is used. We used oversampling and undersampling techniques to balance the datasets to solve this issue. When using SMOTE, synthetic samples are generated for the minority class; however, the methodologies used in SMOTE are based on local knowledge rather than generalized information about the minority class. When it comes to categorization, if certain classes dominate, it can result in biased results. As a result, it is suggested to balance the dataset before fitting. These models effectively capture the spatial and temporal connectivity of anomaly detection problems. IoT networks are comprised of a diverse range of applications and data types. The same model can be applied to a wide variety of IoT applications and data types. Some IoT networks produce a lot of data because it keeps running. The proposed model can handle a lot of data. Additionally, the proposed model achieves better performance results when dealing with large amounts of data. One constraint is that it requires a large amount of data to perform better than other strategies.

VII CONCLUSION AND FUTURE WORK

The design, development, and evaluation of recurrent neural network-based anomaly detection models for IoT networks are proposed in this paper. Two models for multiclass classification were proposed. Each model was implemented using LSTM, BiLSTM, and GRU recurrent neural networks. Seven datasets were used to evaluate the proposed models. The adaptability of multiclass classification models was demonstrated by their high detection rates across LSTM, BiLSTM, and GRU implementation environments. The proposed multiclass classification models were compared to current deep learning-based techniques, and it was observed that the proposed models outperformed them on essentially every assessment criteria. Finally, a lightweight anomaly detection model for binary classification was proposed and implemented utilizing the single hidden layer of LSTM, BiLSTM, and GRU recurrent neural network. The reliability of the proposed architecture for anomaly detection in IoT networks is shown by the multiclass and binary classification models performing consistently across many datasets.

In future work, we plan to investigate more deep learning approaches for anomaly detection in IoT networks, adopting various optimization techniques to boost the detection capability of these models on small datasets. We also plan to develop and evaluate ensemble-based techniques for LSTM, BiLSTM, and GRU models.

REFERENCES

[1] D. Li, L. Deng, M. Lee, and H. Wang, “IoT data feature extraction and intrusion detection system for smart cities based on deep migration learning,” Int. J. Inf. Manage., vol. 49, pp. 533–545, 2019.
[2] Shekhar Saxena, “5 Layer Architecture of Internet of Things.”
[3] P. Sethi and S. R. Sarangi, “Internet of Things: Architectures, Protocols, and Applications,” J. Electr. Comput. Eng., vol. 2017, 2017, doi: 10.1155/2017/932405.
[4] A. Sarangam, “7 IOT Layers That You Should Know in 2021.”
[5] T. Vaiyapuri, Z. Sbai, H. Alasaker, and N. A. Alaseem, “Deep Learning Approaches for Intrusion Detection in IoT Networks–Opportunities and Future Directions.”
[6] H. Hou et al., “Hierarchical long short-term memory network for cyberattack detection,” IEEE Access, vol. 8, pp. 90907–90913, 2020.
[7] M. Al-Hawawreh, N. Moustafa, S. Garg, and M. S. Hossain, “Deep Learning-enabled Threat Intelligence Scheme in the Internet of Things Networks,” IEEE Trans. Netw. Sci. Eng., vol. 4697, no. c, pp. 1–1, 2020, doi: 10.1109/mtns.2020.3032415.
[8] T. Smitenidis, T. Lagkas, and K. Rantos, “Deep Learning in IoT Intrusion Detection,” J. Netw. Syst. Manag., vol. 30, no. 1, pp. 1–40, 2022.
[9] Y. Imrana, Y. Xiang, L. Ali, and Z. Abdul-Rauf, “A bidirectional LSTM deep learning approach for intrusion detection,” Expert Syst. Appl., vol. 185, p. 115524, 2021.
[10] M. M. Salim, S. K. Singh, and J. H. Park, “Securing Smart Cities using LSTM algorithm and lightweight containers against botnet attacks,” Appl. Soft Comput., vol. 113, p. 107859, 2021.
[11] H. Alkahtani and T. H. H. Aldhyani, “Botnet Attack Detection by Using CNN-LSTM Model for Internet of Things Applications,” Secur. Commun. Networks, vol. 2021, 2021.
[12] I. A. Khan, N. Moustafa, D. Pi, W. Haider, B. Li, and A. Jolfaei, “An Enhanced Multi-Stage Deep Learning Framework for Detecting Malicious Activities From Autonomous Vehicles,” IEEE Trans. Intell. Transp. Syst., 2021.
[13] S. I. Popoola, B. Adebisi, M. Hammoudeh, H. Gacanin, and G. Gui, “Stacked recurrent neural network for botnet detection in smart homes,” Comput. Electr. Eng., vol. 92, p. 107039, 2021.
[14] A. K. Desta, S. Ohira, I. Arai, and K. Fujikawa, “Id sequence analysis for intrusion detection in the can bus using long short term memory networks,” in 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2020, pp. 1–6.
[15] B. Wang, Y. Su, M. Zhang, and J. Nie, “A Deep Hierarchical Network for Packet-Level Malicious Traffic Detection,” IEEE Access, vol. 8,
[16] Y. Hao, Y. Sheng, and J. Wang, “Variant gated recurrent units with encoders to preprocess packets for payload-aware intrusion detection,” *IEEE Access*, vol. 7, pp. 49985–49998, 2019.

[17] Q. Wang, W. Zhao, and J. Ren, “Intrusion detection algorithm based on image enhanced convolutional neural network,” *J. Intell. Fuzzy Syst.*, vol. 41, no. 1, pp. 2183–2194, 2021, doi: 10.3233/JIFS-210683.

[18] W. Weng et al., “HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection,” *IEEE access*, vol. 6, pp. 1792–1806, 2017.

[19] C. Yin, Y. Zhu, J. Fei, and X. He, “A deep learning approach for intrusion detection using recurrent neural networks,” *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

[20] F. Meng, Y. Fu, F. Lou, and Z. Chen, “An effective network attack detection method based on kernel PCA and LSTM-RNN,” in *2017 International Conference on Computer Systems, Electronics and Control (ICSEC)*, 2017, pp. 568–572.

[21] A. Diro and N. Chilamkurthy, “Leveraging LSTM Networks for Attack Detection in Fog-to-Things Communications,” *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 124–130, 2018, doi: 10.1109/MCOM.2018.1701270.

[22] B. Roy and H. Cheung, “A Deep Learning Approach for Intrusion Detection in Internet of Things using Bi-Directional Long Short-Term Memory Recurrent Neural Network,” *2018 28th Int. Telecommun. Networks Appl. Conf. ITNAC*, 2018, doi: 10.1109/ATNAC.2018.8615294.

[23] Y. Yang, K. Zheng, C. Wu, X. Niu, and Y. Yang, “Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks,” *Appl. Sci.*, vol. 9, no. 2, p. 238, 2019.

[24] C. Xu, J. Shen, X. Du, and F. Zhang, “An intrusion detection system using a deep neural network with gated recurrent units,” *IEEE Access*, vol. 6, pp. 48697–48707, 2018.

[25] K. Wu, Z. Chen, and W. Li, “A novel intrusion detection model for a massive network using convolutional neural networks,” *IEEE Access*, vol. 6, pp. 50850–50859, 2018.

[26] S. Naseri et al., “Enhanced network anomaly detection based on deep neural networks,” *IEEE Access*, vol. 6, pp. 48231–48246, 2018.

[27] Y. Ding and Y. Zhai, “Intrusion detection system for NSL-KDD dataset using convolutional neural networks,” in *Proceedings of the 2018 International Conference on Computer Science and Artificial Intelligence*, 2018, pp. 81–85.

[28] R. H. Hwang, M. C. Peng, V. L. Nguyen, and Y. L. Chang, “An LSTM-based deep learning approach for classifying malicious traffic at the packet level,” *Appl. Sci.*, vol. 9, no. 16, 2019, doi: 10.3390/app9163414.

[29] H. Alaz-Almoutawakel, J. A. Crespo, M. Alrubaian, H. M. Hassan, and A. Ghareeb, “DeepDCA: Novel network-based detection of IoT attacks using artificial immune system,” *IEEE Access*, vol. 8, pp. 134695–134706, 2020.

[30] B. Susilo and R. S. Sari, “Intrusion detection in IoT networks using machine learning,” in *Information, vol. 11, no. 5*, p. 279, 2020.

[31] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, “Towards a deep learning-based intrusion detection for IoT networks,” in *Proceedings of the 2020 ACM International Conference on Information and Knowledge Management (CIKM)*, 2020, pp. 562–563.

[32] P. B. Borisenko, S. D. Erokhin, A. S. Fadeev, and I. D. Martishin, “Intrusion Detection Using Multilayer Perceptron and Neural Networks with Long Short-Term Memory,” in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, 2020, pp. 562–567.

[33] F. Jiang et al., “Deep learning based multi-channel intelligent attack detection for data security,” *IEEE Trans. Sustain. Comput.*, vol. 5, no. 4, pp. 204–212, 2018.

[34] J. Malik, A. Akhunzada, I. Bibi, M. Imran, A. Musaddiq, and S. W. Kim, “Hybrid deep learning: an efficient reconnaissance and surveillance detection mechanism in SDN,” *IEEE Access*, vol. 8, pp. 134695–134706, 2020.

[35] B. Susilo and R. S. Sari, “Intrusion detection in IoT networks using deep learning algorithm,” *Information, vol. 11, no. 5*, p. 279, 2020.

[36] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, “Towards a deep learning-based intrusion detection approach for Internet of Things,” *Comput. Networks*, vol. 186, no. August 2020, p. 107784, 2021, doi: 10.1016/j.comnet.2020.107784.

[37] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, “Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study,” *J. Inf. Secur. Appl.*, vol. 50, p. 102419, 2020.

[38] M. M. Hassan, A. Goumaei, A. Alsanad, A. Alrajhi, and G. Fortino, “A hybrid deep learning model for efficient intrusion detection in big data environment,” *Inf. Sci. (Ny)*, vol. 513, pp. 386–396, 2020.

[39] S. Aldhaheeri, D. Alghazzawi, L. Cheng, B. Alzahrani, and A. Al Barakati, “DeepDCA: Novel network-based detection of IoT attacks using artificial immune system,” *Appl. Sci.*, vol. 10, no. 6, p. 1909, 2020.

[40] L. Liu, P. Wang, J. Lin, and L. Liu, “Intrusion detection of imbalanced network traffic based on machine learning and deep learning,” *IEEE Access*, vol. 9, pp. 7550–7563, 2020.

[41] J. Ashraf, A. D. Bakhshi, N. Moustafa, H. Khurshid, A. Javed, and A. Beheshti, “Novel deep learning-enabled Istm autoencoder architecture for discovering anomalous events from intelligent transportation systems,” *IEEE Trans. Intell. Transp. Syst.*, 2020.

[42] M. B. Borisenko, S. D. Erokhin, A. S. Fadeev, and I. D. Martishin, “Intrusion Detection Using Multilayer Perceptron and Neural Networks with Long Short-Term Memory,” in *2021 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SYNCHROINFO)*, 2021, pp. 1–6.

[43] T. H. Hai and L. H. Nam, “A Practical Comparison of Deep Learning Methods for Network Intrusion Detection,” in *2021 International Conference on Electrical... Communication, and Computer Engineering (ICECECE)*, 2021, pp. 1–6.

[44] T. S. Pooja and S. Purohit, “Evaluating Neural Networks using Bi-
IMTIAZ ULLAH (Member, IEEE) received the M.Sc. degree in Internet, Computer, and System Security from the Department of Computer Science University of Bradford, U.K., and the M.Sc. degree in Computer Science from the Department of Computer Science University of Peshawar, Pakistan. He is currently pursuing the Ph.D. degree with the Department of Electrical, Computer and Software Engineering, Ontario Tech University, Oshawa, ON, Canada. His current research interests include deep learning and anomaly detection models for the Internet of Things.

QUSAY H. MAHMOUD (Senior Member, IEEE) was the Founding Chair of the Department of Electrical, Computer and Software Engineering at Ontario Tech University in Canada. He is currently a Professor of Software Engineering in the same department, and his research interests include intelligent software systems and cybersecurity.