IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Hybrid Machine Learning Model for Efficient Botnet Attack Detection in IoT Environment

MUDASIR ALI [1], MOBEEN SHAHROZ [2], MUHAMMAD FAHEEM MUSHTAQ [2] SULTAN ALFARHOOD [3,*], MEJDL SAFRAN [3] AND IMRAN ASHRAF [4,*]

[1]Department of Computer Science, The Islamia University of Bahawalpur, Bahawalpur, Punjab, 63100, Pakistan; (mudasiralics786@gmail.com)
[2]Department of Artificial Intelligence, The Islamia University of Bahawalpur, Bahawalpur, Punjab, 63100, Pakistan; (mobeen.shahroz@iub.edu.pk; faheem.mushtaq@iub.edu.pk)
[3] Department of Computer Science College of Computer and Information Sciences King Saud University P.O.Box 51178 Riyadh 11543 Saudi Arabia; (sultanf@ksu.edu.sa, mejdl@ksu.edu.sa)
[4]Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, South Korea; (email: ashrafimran@live.com)

Corresponding author: Correspondence: Imran Ashraf and Sultan Alfarhood;(email: ashrafimran@live.com; sultanf@ksu.edu.sa)

This research is funded by the Researchers Supporting Project Number (RSPD2024R890), King Saud University, Riyadh, Saudi Arabia.

ABSTRACT Cyber attacks are growing with the rapid development and wide use of internet technology. Botnet attack emerged as one of the most harmful attacks. Botnet identification is becoming challenging due to the numerous attack vectors and the ongoing evolution of viruses. As the Internet of Things (IoT) technology is developing rapidly, many network devices have been subject to botnet attacks leading to substantial losses in different sectors. Botnets pose serious risks to network security and deep learning models have shown potential for efficiently identifying botnet activity from network traffic data. In this research, a botnet identification system is proposed based on the stacking of artificial neural network (ANN), convolutional neural network (CNN), long short-term memory (LSTM), and recurrent neural network (RNN) (ACLR). The experiments are conducted by employing both the individual models, as well as, the proposed ACLR model for performance comparison. The UNSW-NB15 dataset is used for botnet attacks and contains nine different attack types including 'Normal', 'Generic', 'Exploits', 'Fuzzers', 'DoS', 'Reconnaissance', 'Analysis', 'Backdoor', 'Shell code' and 'Worms'. Experimental results indicate the proposed ACLR model gains 0.9698 testing accuracy showing that it is successful in capturing the intricate patterns and characteristics of botnet attacks. The proposed ACLR model's k values (3, 5, 7, and 10) for a K-fold cross-validation accuracy score is 0.9749 indicating that the model's robustness and generalizability are demonstrated by k = 5. In addition, the proposed model detects botnets with a high receiver operating characteristic area under the curve (ROC-AUC) of 0.9934 and a precision-recall area under the curve (PR-AUC) of 0.9950. Performance comparison with existing state-of-the-art models further corroborates the superior performance of the proposed approach. The results of this research can be helpful against evolving threats and enhance cyber security procedures.

INDEX TERMS Botnet attack detection; stacking; cyber-attacks; stacked ensemble; deep learning; IoT

I. INTRODUCTION

THE rise of Internet technology led to its rapid and wide adoption by the masses for daily, social, cultural, and institutional activities. Similar to other technologies, the internet also has negative uses where people are targeted to steal their money or personal information. Botnet attack detection is a crucial component of cyber security, largely because it aids in preventing and reducing a variety of online security risks. Security experts can protect networks and data from harmful activities, such as distributed denial of service (DDoS) assaults, data breaches, and malware distribution, by identifying and destroying botnets. Early identification not only minimizes possible damage but also preserves network effectiveness and user confidence in digital services. Additionally, it promotes cyber resilience, guarantees adherence to legal and regulatory standards, and supports innovation in the continuous struggle against globally advancing cyber threats. Deep learning-based botnet detection uses powerful machine

learning models to quickly find and categorize harmful botnet activity in the network data. Quickly identifying and counteracting the cyber dangers posed by botnets, aids organizations in protecting their systems and data. Such attacks have happened on both individuals and groups in a variety of ways to obtain financial advantage. One of the most well-known forms of assault is ransomware, which targets a person and locks their data until they pay the ransom demanded by the attacker. The attackers utilize botnets to assault huge organizations. Due to its success in fending off the growing menace of botnets, botnet detection employing deep learning algorithms has attracted a lot of interest recently.

Network traffic analyzers based on deep learning have become an effective tool for spotting and reducing botnet activity. These analyzers use deep learning models to automatically extract pertinent information from unprocessed packet data. The first few packets in a flow's headers are specifically extracted and examined to look for patterns and traits typical of botnet traffic. Using convolutional neural network (CNN) and autoencoder, it is possible to identify malicious botnet traffic independent of the architecture of the underlying botnet [1]. Autoencoders are used to teach the network how to rebuild its input to learn the fundamental form of network traffic data. This method aids in spotting peculiar patterns that point to botnet activity. By identifying spatial connections and hierarchical representations, CNNs on the other hand excel in the analysis of structured data, such as network traffic. Researchers and practitioners in the field of botnet detection have made tremendous progress in identifying and reducing botnet risks because of the strength of deep learning algorithms. These techniques have produced encouraging results in precisely classifying and identifying botnet traffic, allowing for proactive defenses against botnet attacks [2].

Botnet identification using deep learning algorithms has proven to be a promising method for lessening the threat of botnets. It has been suggested that deep learning-based network traffic analyzers can successfully detect and counteract botnet activity. Bidirectional long-short-term memory recurrent neural networks (BLSTM-RNN) are a famous example of how deep learning is being used in botnet identification. BLSTM-RNN models are well suited for analyzing network traffic and spotting trends connected to botnet activity because they are excellent at collecting both the past and future context of sequential data [3]. There are various benefits to using deep learning algorithms for botnet identification. First of all, these algorithms are capable of learning and adapting automatically to the changing characteristics of botnets, allowing them to recognize novel and previously unknown botnet behaviors. Second, these algorithms can find hidden patterns and anomalies that would not be noticeable using conventional detection techniques by extracting characteristics from raw data packets. Deep learning models can also handle enormous amounts of network traffic data quickly, making it possible to detect botnet activity in real-time or almost real-time. The security and integrity of networks and devices depend on the ability to detect botnet activity. Because they can be used for spam distribution, distributed denial of services (DDoS) attacks, and theft of data, botnets offer a serious threat. By utilizing deep learning algorithms in botnet detection, researchers and practitioners expect to increase the accuracy and efficacy of detection approaches and enable proactive protection strategies against botnet attacks [4].

The topic of botnet identification is facing additional difficulties as a result of the proliferation of Internet of Things (IoT) devices. The possible presence of a few devices becoming infected by botnet viruses can have disastrous effects because there are billions of connected devices worldwide. The size and diversity of IoT networks present challenges for traditional botnet detection methods, underscoring the necessity for cutting-edge solutions. Deep learning techniques for botnet identification have become more popular in this field. The challenge of botnet detection in IoT networks is to efficiently detect and reduce the presence of botnets. This problem is solved by deep learning algorithms, which automatically extract relevant features from unprocessed packets [5]. To properly detect new botnet behaviors, however, the detection models must be capable to adapt and update in the present as botnets continue to develop and adopt complex evasion strategies.

Real-time or nearly real-time detection of botnet activity is made possible by deep learning algorithms' capacity to handle massive volumes of network traffic data effectively. Scalable deep learning architectures that can manage the high-dimensional and dynamic nature of IoT traffic data are difficult to develop and put into practice. The generalization of detection models across various botnet topologies and variants is the focus of the problem statement. To ensure robustness and adaptability in the face of changing botnet threats, deep learning algorithms must be able to learn and detect botnet activities regardless of the underlying network structure [6].

Deep learning techniques for botnet identification have several benefits. They offer automatic and perceptive systems to deal with the diversifying and increasingly sophisticated dangers posed by botnets. These methods have the potential to accurately detect both well-known and newly-discovered botnet activity by utilizing deep neural networks. Additionally, the ability to extract features from packet headers enables effective analysis of network data, making it possible to detect botnet behaviors in real-time [7]. However, there are still issues with the creation and application of these suggested alternatives. Deep learning techniques are currently being modified to handle encrypted traffic and changing botnet structures. Further research is needed in the areas of generalizability of detection models across various network settings and handling the dynamic nature of botnet behaviors [8]. In this regard, this research proposes a stacked model and makes the following primary contributions:

- This research proposes a stacking model ACLR for botnet attack identification to improve security mea-

sures for IoT systems. The proposed model utilizes the strengths of artificial neural network (ANN), convolutional neural network (CNN), long short-term memory (LSTM), and recurrent neural network (RNN).

- Experiments involve the classification of several attack types, including common ones like normal and generic as well as worms, backdoors, shell code, fuzzers, DoS, reconnaissance, and analysis. For experiments, the dataset is preprocessed involving the removal of null values and label encoding for categorical values necessary for training machine learning models.
- The efficacy of the proposed approach is meticulously assessed through a comprehensive set of widely recognized performance evaluation metrics, encompassing accuracy, precision, recall, and the F1 score. In order to add further resilience to the results, the performance is thoroughly verified using k-fold cross-validation, with k values of 3,5, 7, and 10. Moreover, to evaluate the discriminative power of the model, the receiver operating characteristic area under the curve (ROC-AUC) metric is also utilized. In addition, performance comparison with state-of-the-art models is also carried out.

The preceding research is distributed into the following section: Section II presents the literature review of the current research works and technologies. Section III introduces the overall research approach, describing the data collection methods and discussing the data analysis techniques. Section IV shows the results and discussion of the proposed approach. Section V presents the conclusion and establishes a connection to the broader context.

II. RELATED WORK

Attacks on computer networks can read, damage, and steal data, which has a devastating impact on how well the system performs as a whole. Pre-intrusion actions like port scanning and IP spoofing come before attacks. Attacks are discovered by keeping track of data from source and destination IP addresses, ports, protocol specifics, header specifics, etc. [9]. Attacks can be divided into two categories: passive and active, depending on their nature. The passive attack may be network- or system-based, with the attacker covertly monitoring the network to obtain private data. Monitoring passive attacks might be tricky. Active attackers bypass all security precautions and gain access to networks by taking advantage of security flaws, posing as a reliable system, or stealing credentials.

A. CYBER SECURITY

To recognize and reduce one of the biggest hazards to systems connected to the internet, cyber security is a crucial duty. Botnets are collections of hacked computers that are coordinated by a master host and used for malicious purposes like spam distribution, DDoS, and data theft. Traditional botnet detection techniques, like anomaly-based identification and signature-based approaches, have trouble identifying unknown botnets, encrypted traffic, and complex evasion strategies used by attackers. Deep learning methods, however, present a more promising strategy for overcoming these difficulties.

The authors [10] were the first to employ machine learning for botnet traffic detection. They utilized a CNN model in that regard. Experimental results show that the accuracy of the training set is 98.62%, the loss is 4.74% and training takes an average of 32 seconds for each epoch. Accuracy is 99.57%, loss is 1.74% and test duration is 10 seconds per epoch for the test set.

The largest and most destructive internet cybercrimes have involved DDoS attacks. The Mirai botnet was one of the most well-known instances of a DDoS assault using the IoT. A DDoS attack is a kind of cyber-attack in which a hacker temporarily subjugates several compromised systems to attack a particular target and sends concurrent requests to a server for a specific service, overwhelming the server and convincing it to disregard real requests from end users. To create and disseminate a network of robots (botnets) made up of the afflicted IoT devices (bots), Mirai is a piece of malware that infects IoT devices. The attacker (the "botmaster") then instructs the bots to take part in DDoS attacks on Internet targets using a command and control (C&C) server. The research [11] presented a bidirectional LSTM (BLSTM-RNN) approach for botnet attack detection. To determine whether the BLSTM-RNN's incorporation of contextual data received from the past and future may lead to higher accuracy, the model was compared against a unidirectional LSTM-RNN. With 99%, 98%, and 98% validation accuracy and 0.000809, 0.125630, and 0.116453 validation loss metrics, respectively the findings for Mirai, UDP, and DNS were highly encouraging.

The research [12] used classifiers such as k-nearest neighbors (KNN), decision tree (DT), AdaBoost (AB), random forest (RF), linear SVM (LSVM), and radial basis function SVM (RSVM), all of which were applied to the three different sets of DS1 data. The performances obtained by the logistic regression (LR) and Naive Bayes (NB) classifiers were significantly worse.

The authors combined the CNN-LSTM model in [13] to identify DDoS attacks using the CICIDS 2017 dataset. Results show a 97.16% accuracy, 97.41% precision, and 99.1% reliability. A domain generation algorithm attack is studied in [14] and an accuracy of 94.9% is reported. Implementation of countermeasures for cyber attacks incurs large costs; so, cost reduction is an important factor in cyber security. Using only 25% of the implementation budget, the proposed model in [15] outperformed cutting-edge IoT botnet detection techniques in terms of accuracy. As a result, it cuts the implementation budget by almost 75%. The research [16] employs CNN and LSTM for botnet attack detection. In comparison to the average validation accuracy, the average training accuracy for DNNBoT1 and DNNBoT2 was 90.71% and 91.44% respectively, for each was 90.54% and 91.24%.

A growing trend in recent years is deep learning potential methods for botnet detection. Cyber security is seriously

threatened by botnets, which are networks of compromised hosts used by a master host to conduct malicious operations. In order to effectively address the variety of cyber security issues, popular deep learning methods can be used, including their ensembles and hybrid methods [17].

B. MACHINE LEARNING APPROACHES FOR BOTNET ATTACK DETECTION

An all-encompassing, useful technique for learning real-valued, discrete-valued, and vector-valued functions is the ANN. The research [18] proposed an ANN model and tested it using the CTU-13 dataset. In comparison to support vector machine (SVM), and NB, the model yields better accuracy of approximately 99%. Similarly, an ANN is deployed for automatically identifying DDoS attacks in [19]. Results showed that ANN in particular showed a very good accuracy of 99% and proved to be more effective against DDoS attacks.

Various CNN models are also utilized for botnet attack detection. A CNN-LSTM model is utilized in [20] for attack detection IoT settings, categorizing and halting network activity by severing wifi connections. CNN layers are utilized to extract features from the input data while LSTM is used for detection. The authors report good results with a specificity of 93% and an F1 score of 100%. The results demonstrate the innovative outcomes of utilizing the CNN-LSTM model in the analysis of regular packets, fuzzing assaults, and flood attacks. The weighted average findings of the author's suggested approach for identifying the botnet on the Provision PT-737E camera were as follows: 88% for camera precision, 87% for recall, and 83% for F1 score. On the Provision PT-838 camera, the system's classification results for botnet assaults and regular packets were 89% for recall, 85% for F1 score, and 94% for accuracy [21].

A botnet detection system that successfully locates P2P botnets using machine learning is employed in [22]. Using well-known P2P botnet datasets, the proposed approach demonstrates to be effective in identifying botnets with low false positive rates and great accuracy. The recommended technique creates a CNN-based model and gathers flow-based data from packet headers, a technique that is frequently employed for speech and image recognition. The test shows that this step increases detection precision to 98.6% while reducing false positive rates to 0.5%. [23].

The research [24] utilized a long short-term memory autoencoder (LAE) encoding phase to minimize the feature dimensionality of large-scale IoT network traffic data. Using the low-dimensional feature set's long-term interrelated changes to research the deep bidirectional long short-term memory (BLSTM) created by LAE to accurately categorize network traffic samples. The BoT-IoT data collection is used in numerous tests to verify the viability of the suggested hybrid deep learning approach. The LAE model showed the highest percentage of data size reduction (91.89%), according to the simulation's results. As the data size of key network traffic aspects lowers, the application of the deep learning technique in memory-restricted IoT devices appears to be more realistic for efficient botnet identification.

The authors used LSTM-based domain generation algorithms (DGAs) for botnet attack detection [25]. Different DGAs with benign and DGA domain names were employed for multiclass classification. On two different test data sets, the accuracy of the LSTM model for binary classification was 98.7%, whereas 68.3% and 67.0% respectively were the accuracy scores for multi-class classification.

The capability of recurrent neural networks (RNN) to properly identify samples of network traffic from extremely imbalanced classes has been investigated [26]. Stacked RNN (SRNN) surpassed RNN when the functionality of SRNN was assessed using the Bot-IoT dataset. RNN is used to train the feature representations of highly skewed network traffic data for discriminative categorization. Precision, recall, F1, AUC, GM, and MCC were all improved by the SRNN model.

Spam emails are becoming a serious issue for networks and user productivity. The research [27] analyzed and assessed how well deep RRN performs in detecting spam emails. The F score, average validation accuracy for the selected datasets, and estimates of the true positive rate (TPR), true negative rate (TNR), false positive rate (FPR), and true negative rate (TNR) were made. According to the findings, the deep RNN approach has a detection rate and accuracy of up to 98.65%. Additionally, the proposed method has the greatest F score, AUC, and TPR which are approximately and respectively 97.93%, 98.61%, and 98.65%.

Studies on DDoS attacks have identified various defense strategies, including traffic separation, attack identification and mitigation, and source tracking [28]. DDoS detection systems distinguish normal and anomalous activity streams, while traffic separation hinders significant movement. SDNs are vulnerable due to centralized control and security vulnerabilities. The performance of the proposed structure is assessed by the author using various traffic simulation situations, and the outcomes produced by the machine learning DDoS detection module are compared. For NB, SVM, and DT, the accuracy rates attained by the proposed framework were 97.4%, 96.1%, and 98.1%, respectively.

The Network Virtualization Feature replaces specialized hardware with virtual machines for network functions [29]. The System Data Network (SDN) manages all virtual computers and networks, reducing power consumption, efficiency, and security risks. The proposed design for smart cities uses black SDN-IoT with NFV integration, with a tiered approach comprising application, CP, DP, and perception layers. The distributed SDN controller routes data towards itself. The author emphasizes fault tolerance, load balancing, energy and security management, and scalability in SDN management solutions. AI-based methods are insufficient for intelligent decision-making in unpredictable situations. Blockchain, IoT, and AI can improve privacy, security, and transparency.

The Smart Data Network (SDN) approach offers a solution to the security challenges posed by the rapidly expanding In-

ternet of Things (IoT) networks [30]. It decouples control and data planes, simplifies network management, and prepares systems for IoT data attacks. The author presents a scalable method for automatic, verifiable, adaptive, and immutable access control policies for IoT devices.

The research [31] investigates cyber attacks on electrical power grids that occurred in Ukraine in 2015 and 2016. The approach is focused on early-stage attack detection by analyzing anomalies in the communication network. The focus is to locate the active attack in real time using a hybrid graph convolutional LSTM model. For implementation, SDN and anomaly detection are combined showing a detection accuracy of 96% thereby outperforming existing models. Similarly, the authors propose a novel intrusion detection model in [32] to detect attacks on smart grids. The authors combine a deep learning model with a feature selection approach for better detection accuracy. LSTM and extreme gradient boosting (XGBoost) models are combined for intrusion detection. For better parameter selection of XGBoost, a Bayesian method is utilized. Experimental results show superior performance of the proposed approach.

III. METHODOLOGY

The botnet detection is carried out using a hybrid deep learning model proposed in this research. Figure 1 shows the methodology of the proposed approach. The proposed approach is based on model stacking where the output of ANN, CNN, LSTM, and RNN is used for the final prediction. Additionally, it is estimated how well deep learning classification models perform when employed to analyze botnet attack detection using the UNSW-NB15 dataset. The preprocessing is carried out to remove null values and handle categorical data using label encoding. To expedite the process, a variety of deep learning algorithms including ANN, CNN, LSTM, and RNN are applied.

A. DATASET DESCRIPTION

The dataset for botnet detection is collected from the Kaggle dataset repository. The dataset was originally collected from the University of New South Wales (UNSW) for analyzing network behavior [33]. Despite not being created in an IoT environment, the dataset has been utilized in multiple studies on network security and IoT security. The UNSW-NB15 dataset has been used by researchers and cyber security professionals to test intrusion detection systems and create

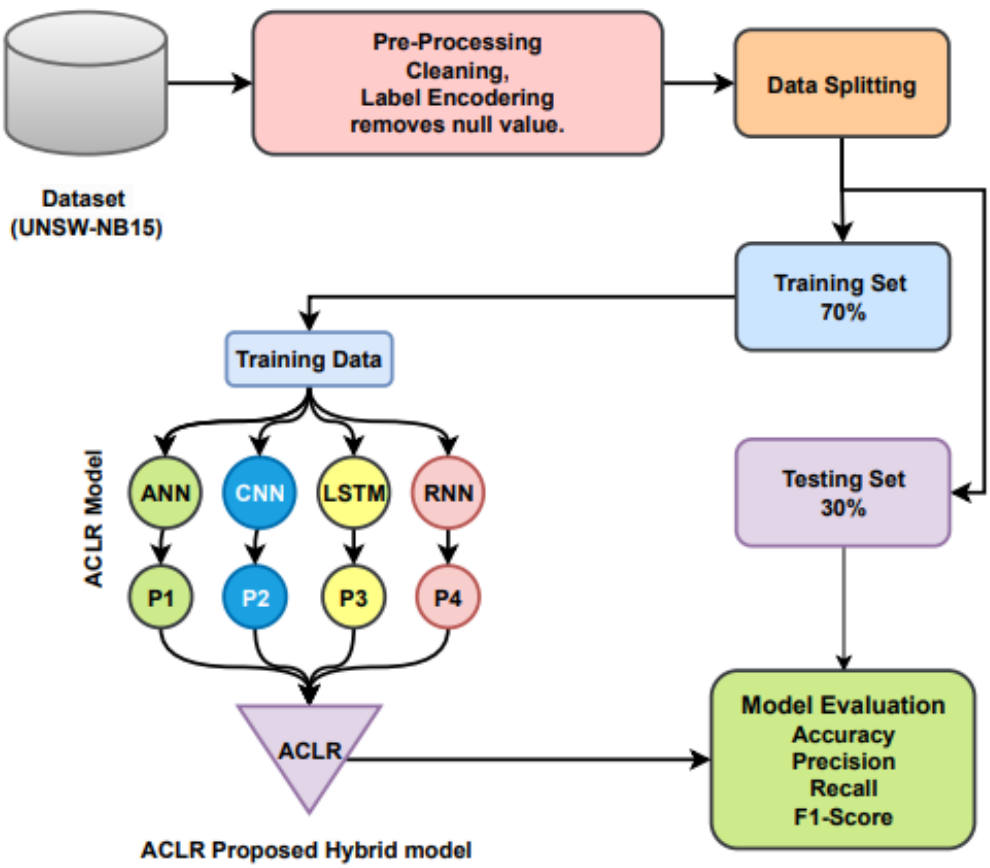

FIGURE 1: Architecture of the proposed approach.

algorithms to find different kinds of network assaults, including those that may harm IoT devices and networks [34]–[36].

TABLE1: Statistics of the UNSW-NB15 dataset.

Attack Types	Absolute Frequencies	Absolute Percentages
Normal	37000	0.449400
Generic	18871	0.229206
Exploits	11132	0.135209
Fuzzers	6062	0.073629
DoS	4089	0.049665
Reconnaissance	3496	0.042462
Analysis	677	0.008223
Backdoor	583	0.007081
Shellcode	378	0.004591
Worms	44	0.000534

The dataset is configured as a training set and testing set, namely UNSW_NB15_training-set.csv and UNSW_NB15_testing-set.csv respectively. Both a training set and a testing set are available for the UNSW-NB15 dataset. To assess the model's performance, the training file set is utilized as a main dataset which is further divided into training and testing datasets for further processing in the ratio of 0.7 to 0.3. The total number of records in the dataset is 82332 that contains nine attack types including 'Normal', 'Generic', 'Exploits', 'Fuzzers', 'DoS', 'Reconnaissance', 'Analysis', 'Backdoor', 'Shell code' and 'Worms'. The number of samples for each class and other details are given in Table 1 and Figure 2.

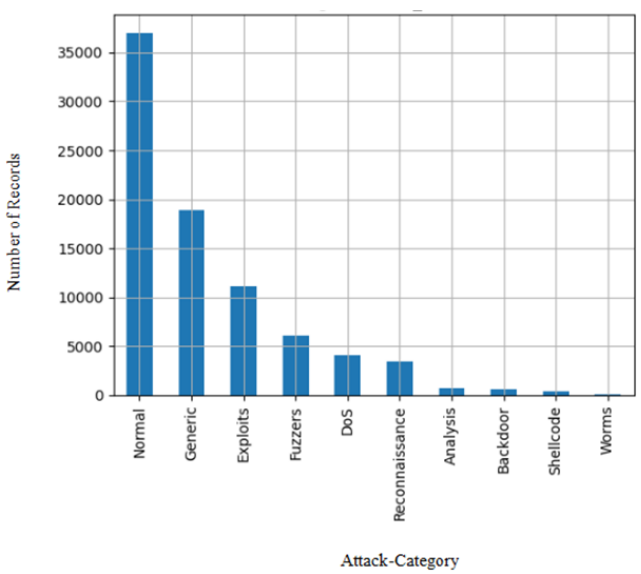

FIGURE2: Target attack category.

A variety of cyber-attacks can be carried out via botnets. Some of the most frequent attack types linked to botnets include

1) Generic

Block ciphers are vulnerable to generic attacks that do not take their internal structure into account. All block ciphers are vulnerable to general attacks because the length of the key and blocks is constrained. By selecting the proper external parameters, generic assaults can be found. Exhaustive key searches, dictionary attacks, rainbow table assaults, and other generic attacks on block ciphers are a few examples [37].

2) Exploits

An attacker seizes the ability to govern computer resources or network data, exploits a weakness in the programmer or operating system, and causes system failures or crashes. Zero-day exploits make use of software flaws that suppliers are unaware of [38].

3) Fuzzers

Fuzzers assault systems by flooding them with a lot of random data to break them and identify faults. It can locate security gaps in networks and operating systems as well as vulnerabilities in software and systems [39]. It has been demonstrated that deep neural networks (DNNs) are extremely sensitive to even small changes in their input data. This problem illustrates how Sensei may improve the robust accuracy of the DNN by up to 11.9% and 5.5% on average when compared to the state-of-the-art for each of the 15 models. This problem is comparable to the overfitting issue in test-based program synthesis and autonomous program repair. Additionally, Sensei-SA can improve strong accuracy while reducing the average DNN training time by 25% [40].

4) Denial of Service

Attacks with DoS suspend service, making network resources inaccessible to users. DoS assaults utilizing machine learning and deep learning models have dramatically increased in frequency and complexity, according to VeriSign [41]. The accuracy of the CNN-based intrusion detection system is higher for DoS attack detection [42].

5) Reconnaissance

Before starting the actual attack, reconnaissance assaults gather all available information regarding the intended system and act as a planning tool. Social, public, and software reconnaissance are the three basic categories of reconnaissance attacks. Information is obtained during this assault using packet sniffing, port monitoring, ping sweeps, and inquiries about internet data [43].

6) Analysis

It uses web scripts, spam emails, and port scanning to access the web application. By thwarting IP spoofing, modifying the frequency of port scans, and switching up the order in which ports are searched; machine learning models can detect port scanning. Because they propagate malicious code, carry out phishing scams, and generate revenue, spam emails are risky. The use of content-based email filtering using machine learning models discovers specific keywords that can result in a high variation between spam and valid emails. One of the many effects of malicious HTML code penetrations is

the exposure of cookies, which changes the content of the victim's page [44].

7) Backdoor

Attacks using backdoors to undermine security measures and gain access to computers and their data. This assault targets user's access to computing resources as well as their privacy [45].

8) Shellcode

A brief piece of code called shell code is utilized as the payload when software vulnerability is exploited. It launches a command interpreter that enables interactive command entry and reports back the results of commands executed on vulnerable systems. Run-time heuristics that depict machine-level operations can be used to identify shell code assaults [46].

9) Worms

By taking advantage of the security flaws, worms reproduce and propagate to other computational resources. Two expected characteristics of the worm detection system are early warning and a quicker response time for countermeasures. It takes into account the payload's structure and content, network traffic, packet headers, and host behavior monitoring for worm detection [47].

B. DATA PREPROCESSING

Preprocessing must be completed before any analysis to make the data ready for the model training and testing. The dataset needs to be loaded, cleaned, modified, and transformed into a form that is appropriate for machine learning models. Figure 2 shows the distribution of the samples for each class. It shows that there is an imbalance in the statistics since at least one of the category labels has lower samples compared to other category labels. To make the number of instances of each attack type equal, resulting in a total of 82332 cases. Then, category features are transformed into numerical values using label encoding. The number of columns are features. This dataset has 45 features. Encoding categorical values into numerical representations that deep learning algorithms can use is a common task when working with categorical data. Using a label encoder, which gives each category in the data a distinct number label, is a well-liked method for this purpose. There are several methods you may use when using a label encoder to handle missing values in categorical data. Before using the label encoder, one choice is to substitute the missing values with a particular placeholder, such as "NaN". This can be accomplished by using the label encoder after transforming the category data into strings. Numerous libraries, like sci-kit learn, use the Label Encoder class from the sklearn. The preprocessing module offers label encoding as a feature. This can be accomplished by utilizing the label encoder class from sci-kit learn, which concentrates on encoding target labels rather than input information.

C. DEEP LEARNING MODELS

ANNs are used in deep learning to conduct complex calculations on vast volumes of data. Healthcare, e-commerce, entertainment, and advertising are just a few applications that often use deep learning. For obtaining better performance from machine learning models, hyperparameter optimization is critical. The hyperparameters selection process involves careful consideration and experimentation to achieve optimal model performance. Cross-validation techniques assess generalization performance, mitigating overfitting risk. Prioritizing hyperparameters with significant influence allows for fine-tuning critical parameters for optimal results, ensuring robust evaluation of model performance. For this purpose, hyperparameter ranges are selected from the existing literature that carried out similar tasks. Once the range of various hyperparameters is defined, the gridsearchCV method is utilized to obtain the best parameters. This method executes the model with all the combinations of the parameters for the given range and provides the best-fit parameters for optimal performance.

1) Artificial Neural Networks

ANNs are crucial components of deep learning which are motivated by the design and operation of the human brain. ANNs are reported for their capacity to recognize intricate relationships and patterns in data, which makes them useful for a number of applications such as time series analysis, picture identification, and natural language processing. They have been successfully used in several industries including finance, healthcare, and autonomous cars. Despite their strength, ANNs can be computationally demanding and need a lot of labeled training data. Despite this, substantial strides in hardware and algorithms have made ANNs a pillar of contemporary machine learning. This research uses an ANN with the architecture shown in Figure 3. The discussion of hyperparameters of ANNs is shown in table 2.

TABLE2: Hyperparameter of ANN model.

Hyperparameter	Values
Number of Layers	3
Layer 1 Dense	64
Layer 2 Dense	32
Layer 3 Dense	1
Activation function	relu
Activation function	sigmoid
Optimizer	adam
loss	binary_crossentropy
metrics	accuracy
batch_size	32
Number of Epochs	30

2) Convolutional Neural Networks

An advanced form of an ANN is CNN, which was created to be particularly effective at processing and analyzing visual data, such as pictures and movies. CNNs are very good at extracting significant patterns and characteristics from images. The input data is subjected to filters in the convolutional

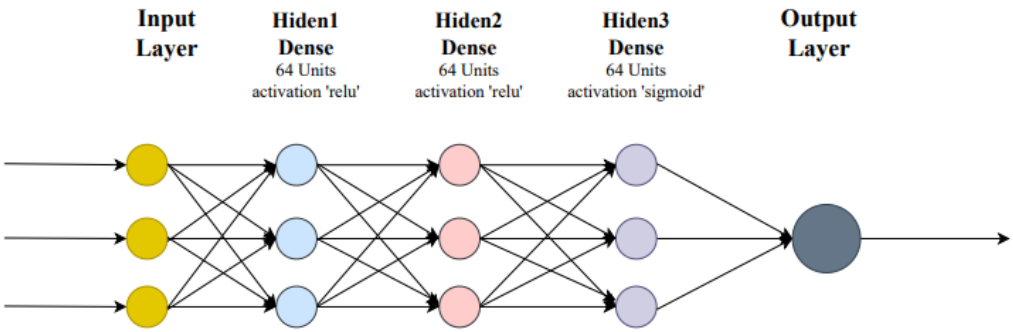

FIGURE3: Architecture of artificial neural network.

layers, which enables the network to learn regional patterns and spatial hierarchies. The feature maps are downscaled by the pooling layers, which reduces computational complexity while preserving crucial data. The completely connected layers at the network's end are where the high-level features are processed and the final output is produced. CNNs have demonstrated astonishing performance in a wide range of computer vision applications including object detection, image classification, and semantic segmentation.

TABLE3: Hyperparameter of CNN model.

Hyperparameter	Values
Number of Convolutional Layers	2
Number of Filters (Layer 1)	64
Number of Filters (Layer 2)	32
Activation function	relu
Activation function	sigmoid
Kernel Size	(3,3)
Pooling	MaxPooling (2x2)
Optimizer	adam
loss	binary_crossentropy
metrics	accuracy
batch_size	32
Number of Epochs	30

The effectiveness of CNNs can be due to their capacity to take advantage of shared weights and local spatial correlations. Compared to fully linked networks, this dramatically decreases the number of parameters, making CNNs more effective and simpler to train. This research uses a CNN for botnet detection using the architecture shown in Figure 4. The hyperparameters of the CNN model are shown in Table 3.

3) Long Short-Term Memory Network

The LSTM algorithm is an example of recurrent neural network (RNN) architecture created expressly to represent and analyze sequence data. LSTMs can capture long-term dependencies and maintain information over long time periods, unlike conventional RNNs. The pertinent information is stored and updated by the memory cells and the gates decide which facts are important to remember and which are not. The three gates that control the information flow via the LSTM units are gates for input, forgetting, and output. Because of their special construction LSTMs can handle sequences of various lengths and detect temporal relationships in the data.

LSTMs are widely used in speech recognition, natural language processing, and time series analysis. Significant improvements have been made across a variety of fields, including a result of the high efficiency with which LSTMs have been shown to capture intricate patterns and correlations within sequential data. Figure 5 shows the architecture of LSTM used in this research. The hyperparameters of LSTM are shown in Table 4.

4) Recurrent Neural Network

An artificial neural network that works well for processing sequential data is RNN. RNNs, as opposed to conventional

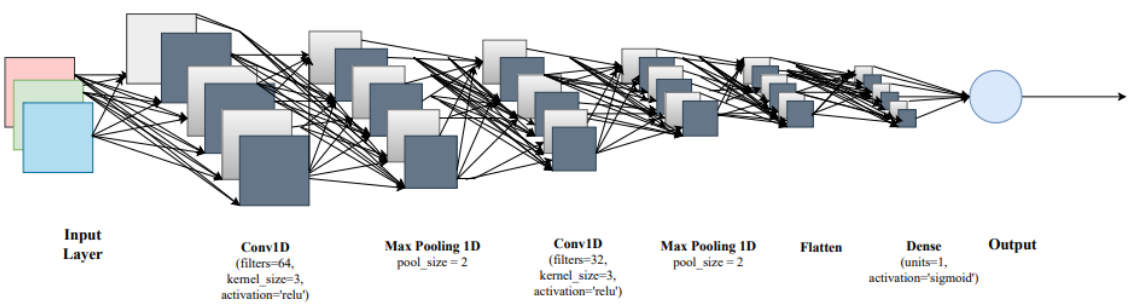

FIGURE4: Architecture of convolutional neural network.

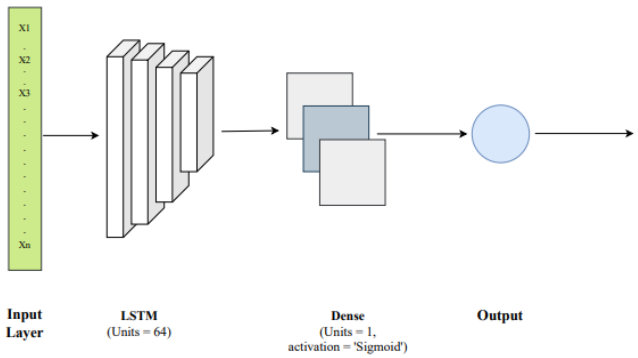

FIGURE5: Architecture of long short-term memory network.

TABLE4: Hyperparameter of LSTM model.

Hyperparameter	Values
Number of Layers	2
LSTM (Layer 1)	64
Dense (Layer 2)	32
Activation function	sigmoid
Optimizer	adam
loss	binary_crossentropy
metrics	accuracy
batch_size	32
Number of Epochs	30

feed-forward networks, feature connections between nodes that create a directed cycle, which enables them to remember information and gain knowledge from prior inputs. RNNs can represent sequences of any length and capture temporal dependencies thanks to their cyclic structure. Information travels through time because each node in an RNN receives input from both the input current and the prior hidden state.

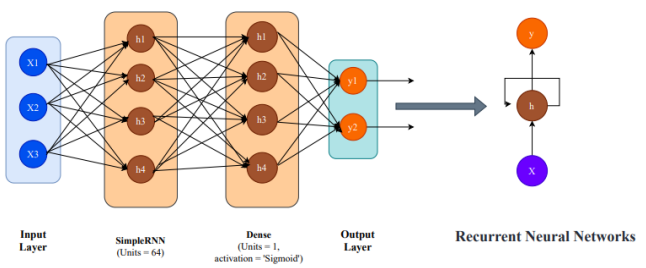

FIGURE6: Recurrent neural network used in this research.

RNNs are effective for tasks like processing natural language, speech recognition, and handwriting recognition due to their recursive nature. The vanishing gradient problem, in which gradients get smaller and smaller with time, can limit the ability of conventional RNNs to detect long-range dependencies. Variations like the LSTM and gated recurrent unit (GRU) are created to remedy this. The architecture of the sued RNN is shown in Figure 6. The hyperparameters of RNN are shown in Table 5.

Although LSTM is a type of RNN, they both bear several differences in their architecture and working functionality. A

TABLE5: Hyperparameter of RNN model.

Hyperparameter	Values
Number of Layers	2
SimpleRNN	64
Dense	1
Activation function	sigmoid
Optimizer	adam
loss	binary_crossentropy
metrics	accuracy
batch_size	32
Number of Epochs	30

comprehensive overview of their differences is presented in Table 6.

TABLE6: Major differences between LSTM and RNN.

Feature	RNN	LSTM
Architecture	Simple and straightforward recurrence.	More complex, including memory cells and gates.
Memory Handling	Short memory span due to vanishing and exploding gradient problems.	Long memory span, handles vanishing and exploding gradient problems effectively.
Memory Cells	No explicit memory cells.	Contains memory cells to store and access information over long sequences.
Gating Mechanisms	No gating mechanisms.	Gating mechanisms (e.g., input, output, forget) control information flow.
Gradient Flow	Suffers from vanishing or exploding gradients over long sequences.	Mitigates vanishing/exploding gradients through gating mechanisms, allowing better learning.
Training Complexity	Less complex	More complex due to additional parameters and gating mechanisms.
Performance	Struggles with long-term dependencies.	Well-suited for capturing long-term dependencies.

D. PROPOSED STACKING ACLR MODEL

A deep learning technique called stacking is used to integrate the results of various classification models to get a more potent and precise final prediction. To improve the overall predictive performance, stacking combines several models. By stacking models, we can benefit from the strengths and diversity of different algorithms. Each algorithm may have its unique way of capturing the data's patterns and linkages and by combining their predictions, we can make a prediction that is more reliable and precise. Stacking helps in reducing bias and variance, improving generalization, to the training data.

Figure 7 shows the architecture of the proposed ACLR model. Stacking is a powerful ensemble learning technique that leverages the strengths of multiple deep learning algorithms including ANN, CNN, LSTM, and RNN. While deep learning stacking methods solve compli-

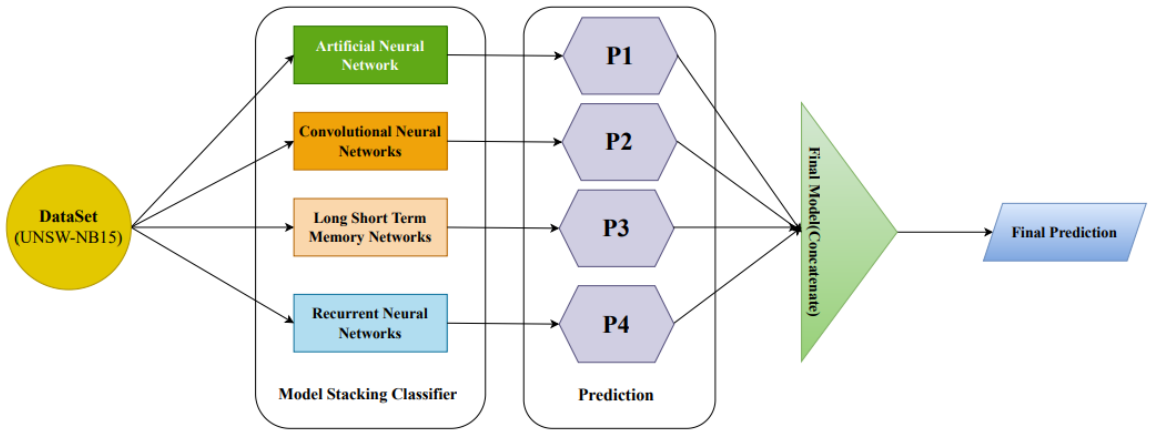

FIGURE7: Architecture of proposed Hybrid ACLR model.

cated problems but demand more data and computer resources and give less interpretability, ensemble techniques (ANN+CNN+LSTM+RNN) mix multiple models for enhanced generalization and interpretation ability. By combining their predictions in a layered fashion stacking can enhance the overall predictive performance leading to improved accuracy and robustness in deep learning models. Stacking discovers the most effective way to combine the predictions from various successful machine learning models. By utilizing each of their own capabilities, minimizing overfitting, and enhancing feature extraction, stacking RNNs, ANNs, CNNs, and LSTMs improves performance in deep learning. By using an ensemble technique, models become more resilient and flexible when dealing with different data distributions. It also increases computing complexity and presents difficulties in understanding the model, necessitating careful evaluation of the combination and training approach. Applying a single model will not produce excellent results but, when many models are combined through stacking, good accurate results will be produced. This is a novel approach in terms of these strategies. A complete list of the hyperparameters is given in Table 7.

E. EVALUATION PARAMETERS

Accuracy, recall, and precision are common measures for assessing the efficacy of machine learning models, particularly for classification tasks. The model's ability to forecast the various classes or categories of the input data is indicated by these measures.

Despite being a frequently used statistic, accuracy is not

TABLE7: Hyperparameters of models.

Models	Layers	Unit	Kernal_size	Activation Function	Loss	Optimizer	Metrix	Epoch	Accuracy
ANN	3	64,32,1	-	relu,sigmoid	binary_crossentropy	adam	accuracy	25	0.7568
ANN	3	64,32,1	-	tanh, sigmoid	binary_crossentropy	adamax	accuracy	25	0.7010
ANN	3	64,32,1	-	tanh,relu	categorical_crossentropy	rmsprop	accuracy	25	0.7268
ANN	3	64,32,1	-	softmax,sigmoid	binary_crossentropy	adagrad	accuracy	25	0.7465
ANN	3	64,32,1	-	tanh, sigmoid	mean_squared_error	sgd	accuracy	25	0.7567
CNN	6	64,32	3	relu,sigmoid	binary_crossentropy	adam	accuracy	30	0.944
CNN	6	64,32	3	elu,softmax	binary_crossentropy	rmsprop	accuracy	30	0.7582
CNN	6	64,32	3	softmax,sigmoid	categorical_crossentropy	adagrad	accuracy	30	0.8874
CNN	6	64,32	3	sigmoid,tanh	binary_crossentropy	sgd	accuracy	30	0.9273
CNN	6	64,32	3	elu,sigmoid	mean_squared_error	adamax	accuracy	30	0.6993
LSTM	2	64,1	-	sigmoid	binary_crossentropy	adam	accuracy	30	0.9651
LSTM	2	64,1	-	elu	binary_crossentropy	rmsprop	accuracy	30	0.9436
LSTM	2	64,1	-	softmax	categorical_crossentropy	adagrad	accuracy	30	0.9516
LSTM	2	64,1	-	tanh	binary_crossentropy	adamax	accuracy	30	0.9638
LSTM	2	64,1	-	relu	mean_squared_error	sgd	accuracy	30	0.9311
RNN	2	64,1	-	sigmoid	binary_crossentropy	adam	accuracy	10	0.9486
RNN	2	64,1	-	relu	binary_crossentropy	adamax	accuracy	10	0.9213
RNN	2	64,1	-	tanh	categorical_crossentropy	adagrad	accuracy	10	0.9471
RNN	2	64,1	-	softmax	binary_crossentropy	sgd	accuracy	10	0.9121
RNN	2	64,1	-	selu	mean_squared_error	rmsprop	accuracy	10	0.9491
ACLR	17	64,32,1	3	relu,sigmoid	binary_crossentropy	adam	accuracy	25	0.9698
ACLR	17	64,32,1	3	relu,softmax	binary_crossentropy	sgd	accuracy	25	0.9274
ACLR	17	64,32,1	3	elu,tanh	categorical_crossentropy	rmsprop	accuracy	25	0.9482
ACLR	17	64,32,1	3	selu,sigmoid	binary_crossentropy	adamax	accuracy	25	0.9598
ACLR	17	64,32,1	3	tanh,softmax	mean_squared_error	adagrad	accuracy	25	0.9058

always sufficient, especially when dealing with unbalanced datasets. The proportion of accurately anticipated positive instances recall, often referred to as sensitivity or the true positive rate, refers to responses to actual positive events. It demonstrates that the model can accurately identify all relevant positive cases. Higher recall values suggest that the model is good at minimizing false negatives. Precision, commonly referred to as positive predictive value, is the ratio of accurately predicted positive cases to all expected positive cases. It demonstrates the model's accuracy in identifying positive samples and minimizing false positives. High accuracy values imply that the model generates minimal false positives. The following equations are used to calculate the evaluation parameters

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{1}$$

where TP refers to true positive which is the number of correctly predicted attacks. FP refers to a false positive which indicates a normal packet predicted as an attack. Similarly, TN indicates true negative which shows a normal packet predicted as a normal packet. Conversely, FN indicates an attack predicted as a normal packet.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

An often-used statistic in research articles to assess the efficacy of classification systems is the F1 score. By including precision and recall into a single score, it provides an accurate evaluation of the model's performance. The following formula used to determine the F1 score is

$$F1\ score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \tag{4}$$

The ROC-AUC measures the area under the Receiver Operating Characteristic curve, which is a graphical representation of a binary classification model's performance. It plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various thresholds. The ROC-AUC represents the area under the ROC curve and ranges from 0 to 1, where a higher value indicates better model performance. A value of 0.5 suggests random classification, while a value of 1 indicates perfect classification.

$$\text{ROC-AUC} = \int_0^1 \text{TPR}(\text{FPR})\, d\text{FPR} \tag{5}$$

The PR-AUC measures the area under the Precision-Recall curve, which assesses a model's performance in binary classification, with a focus on positive class prediction accuracy. The PR-AUC represents the area under the Precision-Recall curve and ranges from 0 to 1, with higher values indicating better model performance. A value of 1 means perfect precision and recall, while a value of 0 suggests poor performance.

$$\text{PR-AUC} = \int_0^1 \text{Precision}(\text{Recall})\, d\text{Recall} \tag{6}$$

In this formula, the integral represents the area under the curve, Precision and Recall are functions of the threshold applied to the model's predicted probabilities.

IV. RESULTS AND DISCUSSION

The experiments have been conducted with multiple deep learning models but due to their complexity and requires large number of datasets, show low prediction accuracy. That's why the selection criteria for the deep learning algorithms (ANN, CNN, LSTM, and RNN) have been adopted based on less complexity and works better with small datasets instead of large datasets. By using 70% training data and 30% testing data from the dataset, the performance of the deep learning model is assessed. The predictability of the model is investigated using a random look at set sizes from the dataset. Experiments are conducted to evaluate the multi-class deep learning models' precision, recall, accuracy, and F1 score.

A. PERFORMANCE OF ANN

The ANN is trained on a labeled dataset to calculate the evaluation metrics, and it is then assessed on a test dataset. The ANN's performance is illustrated in Table 8. It successfully predicted different types of attacks with an accuracy of 0.7568Ṡimilarly, other performance evaluation metrics are good including a precision of 0.7817, a recall of 0.7568, an F1 score of 0.7559, and a training duration of 25.13 seconds.

TABLE8: Performance of ANN model.

Epochs	Accuracy	Precision	Recall	F1-Score	Training Time
5	0.7275	0.7631	0.7274	0.7088	33.16s
10	0.7479	0.7483	0.7479	0.7456	26.14s
15	0.6868	0.7134	0.6868	0.6644	28.16s
20	0.6761	0.7442	0.6761	0.6647	28.15s
25	0.7568	0.7817	0.7568	0.7559	25.13s
30	0.6332	0.7750	0.6332	0.5536	27.14s

B. RESULTS OF CONVOLUTIONAL NEURAL NETWORKS

Many text classification tasks are performed using CNNs, which may also be used to determine accuracy, precision, and F1 score. To calculate these metrics using a test dataset a CNN's performance can be evaluated after having been trained on a labeled dataset. An indicator of how well CNN made general predictions is the percentage of cases in the test set that were correctly categorized as a whole.

TABLE9: Performance of CNN.

Epochs	Accuracy	Precision	Recall	F1-Score	Training Time
5	0.7663	0.7658	0.7663	0.7652	107.24s
10	0.8851	0.8879	0.8851	0.8843	215.11s
15	0.8957	0.8967	0.8957	0.8959	321.11s
20	0.9268	0.9279	0.9268	0.9264	427.14s
25	0.9392	0.9398	0.9392	0.9390	533.11s
30	0.9440	0.9444	0.9440	0.9439	639.15s

Results given in Table 9 show the performance of the CNN model for different epochs ranging from 5 to 30 epochs. Results indicate that the CNN model achieved an accuracy score of 0.9440, precision of 0.9444, recall of 0.9440, F1 score of 0.9439, and a 639.15-second training period. These results are significantly better compared to results obtained by the ANN model.

C. RESULTS OF LONG SHORT TERM MEMORY NETWORKS

The ability of LSTMs to recognize long-term dependencies in sequential data makes them particularly useful for tasks like natural language processing voice recognition and time series analysis. LSTMs can achieve high accuracy, precision, and F1 scores by utilizing their capacity to model context and sequential patterns resulting in trustworthy and accurate classification results for applications requiring sequence-based data. Experimental results of the LSTM model are given in Table 10.

TABLE10: Performance of LSTM model.

Epochs	Accuracy	Precision	Recall	F1-Score	Training Time
5	0.9494	0.9507	0.9494	0.9495	670.80s
10	0.9608	0.9608	0.9608	0.9608	1325.36s
15	0.9559	0.9565	0.9559	0.9560	1983.58s
20	0.9641	0.9641	0.9641	0.9641	2641.36s
25	0.9631	0.9631	0.9631	0.9631	3299.32s
30	0.9651	0.9651	0.9651	0.9651	3957.32s

The results of the LSTM model indicate its superior performance compared to ANN and CNN models. Compared to the highest accuracy score of 0.9440 which CNN obtained with 30 epochs. the LSTM model obtained an accuracy score of 0.9651 for the same number of epochs. Similarly, other performance parameters are also better than ANN and CNN. LSTM obtains a precision score of 0.9651, a recall score of 0.9651, an F1 score of 0.9651, and 3957.32 seconds to complete the training session.

D. RESULTS OF RECURRENT NEURAL NETWORKS

RNNs are excellent at modeling sequential data and can detect temporal dependencies. They are useful for applications like time series analysis natural language processing and speech recognition because of their recurrent connections which let them retain knowledge from earlier time steps. RNNs can achieve high accuracy, precision, and F1 scores by utilizing their capacity to record temporal patterns and contextual data, enabling precise and reliable sequence classification.

TABLE11: Performance of RNN model.

Epochs	Accuracy	Precision	Recall	F1-Score	Training Time
5	0.9391	0.9396	0.9391	0.9389	308.8s
10	0.9486	0.9486	0.9486	0.9486	616.16s
15	0.9522	0.9523	0.9522	0.9521	925.24s
20	0.9471	0.9472	0.9471	0.9471	1234.32s
25	0.9477	0.9479	0.9477	0.9478	1542.40s
30	0.9448	0.9448	0.9448	0.9448	1851.48s

Experimental results of the RNN model are given in Table 11. Results indicate that the RNN model can achieve the highest accuracy score of 0.9522 with 15 epochs. Similarly, the highest precision score is 0.9523, the recall score is 0.9522, the F1 score is 0.9521, and 925.24 seconds ended the training phase. Although these results are not better than what we achieved using the LSTM model, the performance of RNN is substantially better than the ANN model. Similarly, LSTM has a slightly better performance than the CNN model.

E. RESULTS USING STACKING ACLR MODEL

Comparing deep learning models is a vital part of this research. In this research, we contrast the most widely used deep learning classification techniques. Similarly, the results of the proposed ANN+CNN+LSTM+RNN(ACLR) model are analyzed; Table 12 shows experimental results using the proposed model. Experimental results demonstrate that the proposed model shows better performance than other deep learning models used in this research with the highest accuracy score of 0.9698. This accuracy is better than the LSTM model which has the best accuracy score of 0.9651. The precision, recall, and F1 scores of the proposed model are also the highest with 0.9691, 0.9693, and 0.9692 for the period of the training session, respectively.

TABLE12: Performance of the proposed ACLR model.

Epochs	Accuracy	Precision	Recall	F1-Score	Training Time
5	0.9500	0.9407	0.9703	0.9553	915.8s
10	0.9554	0.9517	0.9681	0.9598	1819.21s
15	0.9583	0.9634	0.9608	0.9621	2726.53s
20	0.9634	0.9739	0.9593	0.9665	3634.42s
25	0.9698	0.9691	0.9693	0.9692	4543.50s
30	0.9597	0.9610	0.9659	0.9635	5451.57s

F. COMPARATIVE ANALYSIS OF ALL MODELS

Table 13 shows the results of all models employed in this research. Results are given regarding the accuracy, precision, recall, and F1 scores. It can be observed that the proposed model performs better than all other models including ANN, CNN, LSTM, and RNN which are fine-tuned to obtain optimal performance using the current dataset. Of the employed models, ANN shows the poorest performance with a 0.7568 accuracy score. The CNN model shows slightly better performance than the ANN model with an accuracy score of 0.9440. The performance of both LSTM and RNN is substantially better than ANN with 0.9651 and 0.9522 accuracy scores, respectively. However, the performance of the proposed model is superior among all employed models. Figure 8 illustrates a visual presentation of the performance of all the deep learning models used in this research. Evaluation of the stacking strategy for comparability.To Compare the models and their performance. The proposed strategy stacking provides good results.

TABLE13: Analysis of all employed models.

Models	Accuracy	Precision	Recall	F1-Score	Training Time
ANN	0.7568	0.7817	0.7568	0.7559	25.13s
CNN	0.9440	0.9444	0.9440	0.9439	533.11s
LSTM	0.9651	0.9651	0.9651	0.9651	3299.32s
RNN	0.9522	0.9523	0.9522	0.9521	1542.40s
ACLR	0.9698	0.9691	0.9693	0.9692	4543.50s

G. RESULTS OF K-FOLD CROSS-VALIDATION FOR ACLR MODEL

For performance validation, k-fold cross-validation is carried out for the proposed approach. Table 14, 15, 16 and 17 shows the results with the training time and for k=3,5,7, and 10, respectively. Results indicate that the performance of the proposed models varies slightly across different folds, however, its average performance is better than other models.

TABLE14: Model performance with k =3.

Fold	Accuracy	Precision	Recall	F1-Score	Training Time
1	0.9403	0.9285	0.9661	0.947	10.38s
2	0.9437	0.934	0.9662	0.9498	10.38s
3	0.943	0.9301	0.9692	0.9492	10.36s

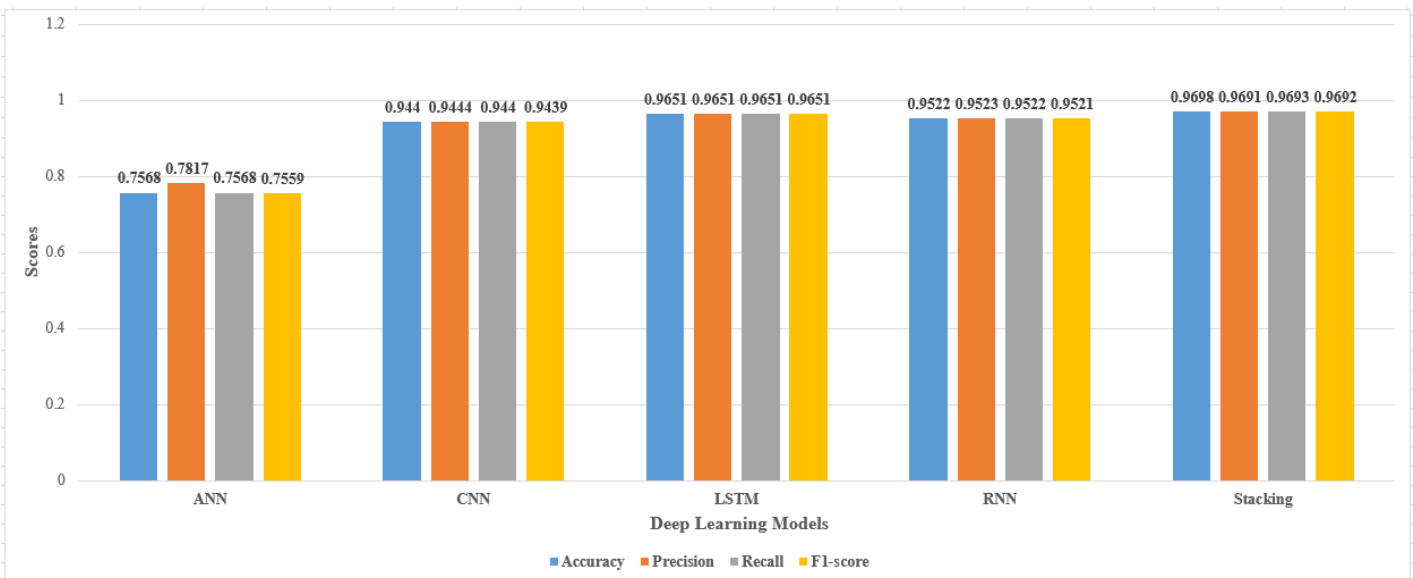

FIGURE8: Performance analysis of deep learning models.

TABLE15: Model performance with k =5.

Fold	Accuracy	Precision	Recall	F1-Score	Training Time
1	0.9749	0.9770	0.9717	0.9723	10.38s
2	0.9708	0.9711	0.9709	0.9731	10.38s
3	0.9659	0.9665	0.9644	0.9654	10.36s
4	0.9714	0.9713	0.9701	0.9732	11.38s
5	0.9713	0.9708	0.9711	0.9710	11.39s

H. ROC-AUC AND PR-AUC OF THE MODELS PERFORMANCE

To extract insights and make wise judgments, proposed data-driven solutions use a variety of neural network architectures, including ANN, CNN, LSTM, and RNN. ROC-AUC and PR-AUC metrics are used to measure model efficacy, guaranteeing strong classification performance and a thorough comprehension of model behavior. The performance of ROC-AUC and PR-AUC is shown in Figures 9, and 10. The accuracy shows the correctly predicted values over the total number of predictions which shows how many numbers of attacks have been detected in real-life systems as shown in the tables of the result section.

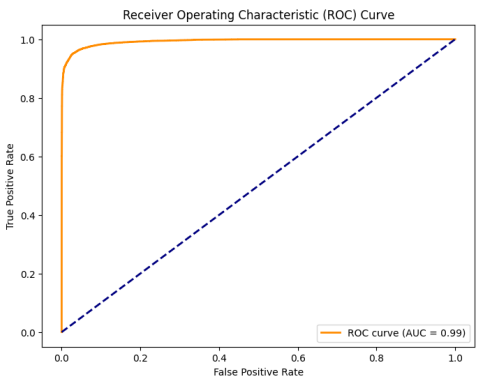

FIGURE9: Performance regarding ROC-AUC curve.

TABLE16: Model performance with k =7.

Fold	Accuracy	Precision	Recall	F1-Score	Training Time
1	0.9556	0.9582	0.9609	0.9596	10.38s
2	0.9475	0.97	0.9341	0.9517	10.38s
3	0.9446	0.9454	0.9549	0.9501	10.36s
4	0.9272	0.906	0.9662	0.9351	11.38s
5	0.9404	0.9454	0.9454	0.9454	11.39s
6	0.9477	0.9559	0.9502	0.9531	11.39s
7	0.9452	0.9466	0.955	0.9506	11.39s

TABLE17: Model performance with k =10.

Fold	Accuracy	Precision	Recall	F1-Score	Training Time
1	0.9387	0.9363	0.9539	0.945	10.38s
2	0.9518	0.9533	0.959	0.9562	10.38s
3	0.92	0.9458	0.9039	0.9244	10.36s
4	0.9503	0.9577	0.9502	0.9539	11.38s
5	0.9444	0.9448	0.9561	0.9504	11.39s
6	0.9589	0.9647	0.9604	0.9625	11.39s
7	0.9463	0.9689	0.9328	0.9505	11.39s
8	0.9512	0.9585	0.9536	0.956	9.33s
9	0.9455	0.9655	0.9337	0.9494	9.32s
10	0.9533	0.9548	0.9469	0.9507	8.29s

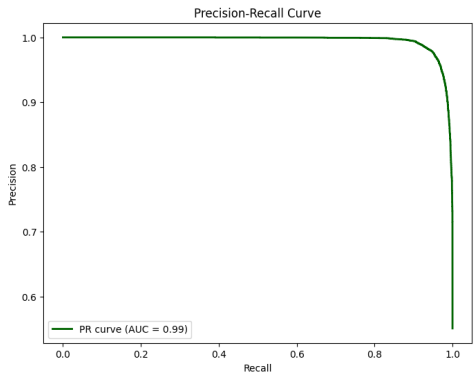

FIGURE10: Performance regarding PR-AUC curve.

As an example in Table 18 the proposed ACLR model shows an accuracy of 97.49% which means approximately 97.50% predictions are accurately performed by the proposed model and with the training period. Where precision 97.70% and recall 97.17% present the class-wise results and prediction accuracy per class. And f1 score shows the harmonic mean of the precision and recall which is 97.23% shows the balance between prediction accuracies.

I. PERFORMANCE COMPARISON WITH EXISTING APPROACHES

Performance comparison with existing state-of-the-art models is necessary to determine the efficacy of the proposed model. For this purpose, several existing studies have been selected that utilized deep learning models for botnet detection. For example, studies [49], [50], [52], [54], [57] and [53] deployed either CNN or an ensemble of CNN with a deep learning model and obtained very good results. Similarly, [48] proposed an ANN model and utilized the same dataset that is used in this research. Both [51] and [55] made use

TABLE18: Comparative analysis of the stacking algorithm.

Models	Accuracy	Precision	Recall	F1-Score	ROC-AUC	PR-AUC	Training Time
ANN+CNN	0.8211	0.8280	0.8520	0.8398	0.9184	0.9333	81.44s
RNN+ANN	0.9223	0.9156	0.9461	0.9306	0.9808	0.9850	292.161s
LSTM+RNN	0.7949	0.9689	0.6483	0.7768	0.7887	0.8489	740.407s
CNN+LSTM	0.9106	0.8779	0.9730	0.9230	0.9838	0.9872	316.173s
Proposed ACLR	0.9749	0.9770	0.9717	0.9723	0.9934	0.9950	1604.104s

TABLE19: Comparative analysis with existing literature.

Reference	Algorithms	Dataset	Classes	Accuracy	Precision	Recall	F1-score
[48]	ANN	UNSW-NB15	Multi-class	0.6397	-	-	-
[49]	CNN+BLSTM	UNSW-NB15	Multi-class	0.7556	0.7933	0.7564	0.7744
[50]	CNN1D+BLSTM	UNSW-NB15	Multi-class	0.7632	0.8100	0.7600	0.7700
[51]	LSTM	UNSW-NB15	Binary-class	0.7000	-	-	-
[52]	SimpleRNN	UNSW-NB15	Binary-class	0.8070	0.7750	0.9840	0.8670
[53]	CNN-LSTM	UNSW-NB15	Binary-class	0.9368	-	-	-
[54]	CNN 1D	UNSW-NB15	Binary-class	0.8980	-	-	-
[55]	LSTM	UNSW-NB15	Binary-class	0.8899	-	-	-
[56]	CNN	UNSW-NB15	Binary-class	0.8112	0.8237	0.7836	0.8006
[57]	CNN-LSTM	UNSW-NB15	Binary-class	0.8993	0.8615	-	0.9043
Proposed	ACLR	UNSW-NB15	Binary-class	0.9698	0.9691	0.9693	0.9692
K-fold	ACLR	UNSW-NB15	Binary-class	0.9749	0.9770	0.9717	0.9723

of LSTM models for the same task while [56] leveraged an RNN model. Results comparison given in Table 19 reveals that the proposed model outperforms these studies and obtains a better accuracy score using the same dataset.

J. PERFORMANCE COMPARISON OF MACC, FLOPS AND INFERENCE TIME.

Within the domain of computational efficiency, assessing the Performance Comparison of Multiply-Accumulate Operations (MACC), Floating Point Operations Per Second (FLOPS), and Inference Time is crucial for evaluating the effectiveness of machine learning models. The study obtained the computational capabilities of a single model in comparison to the combination of two models, specifically considering MACC, FLOPS, and inference time metrics. The analysis uncovered that employing a singular model substantially decreased both MACC and FLOPS requirements, indicating a more efficient computational process. Additionally, the inference time for the individual model was significantly reduced compared to the merged model, highlighting the efficiency of a consolidated approach. The proposed model not only demonstrated superior time efficiency but also yielded improved results compared to the cumulative processing time of the two models. This underscores the importance of optimizing model architecture for enhanced computational efficiency, contributing to more effective and swift machine learning applications. Table 20 presents a comparison of the MACC, FLOPS, and inference time models' respective performances.

TABLE20: Performance comparison of MACC, FLOPS, and inference time.

Models	MACC	FLOPS	Inference Time
ANN	1556640320	10772022.54	0.1053 seconds
CNN	15554	3912.37	3.9755 seconds
LSTM	33922	818.03	41.468 seconds
RNN	8578	1017.0066	8.4346 seconds
ANN+CNN	251865900	503731800	10.5890 seconds
RNN+ANN	483057900	966115800	24.600 seconds
LSTM+RNN	1561163500	3122327000	83.094 seconds
CNN+LSTM	615449900	1230899800	41.606 seconds
ACLR	88946	2240.3580	39.701 seconds

K. COMPUTATIONAL COMPLEXITY VS ACCURACY

Computational complexity is very important for botnet attack detection approaches as they have to work in real time. Often a trade-off is made between the computational complexity of the model and its accuracy. A better accuracy can be obtained for the deep learning model by increasing its number of layers and neurons, however, it would increase the training time of the model. Table 21 shows the relationship between the training time and attack detection accuracy for various models used in this study. Results suggest increased training time when the number of epochs is increased for better training of the models. With increased training time, a better accuracy is observed for CNN, LSTM, RNN, and ACLR time, except for a few cases for LSTM and RNN where a marginal decrease is observed in the accuracy.

TABLE21: Model accuracy vs training time, layers, and epochs.

Model	Layers	Epoch	Accuracy	Training time
CNN	**6**	5	0.7663	107.24s
		10	0.8851	215.11s
		15	0.8957	321.11s
		20	0.9268	427.14s
		25	0.9392	533.11s
		30	0.9440	639.15s
LSTM	**2**	5	0.9494	670.80s
		10	0.9608	1325.36s
		15	0.9559	1983.58s
		20	0.9641	2641.36s
		25	0.9631	3299.32s
		30	0.9651	3957.32s
RNN	**2**	5	0.9391	308.8s
		10	0.9486	616.16s
		15	0.9522	925.24s
		20	0.9471	1234.32s
		25	0.9477	1542.40s
		30	0.9448	1851.48s
ACLR	**3**	5	0.9500	915.8s
		10	0.9554	1819.21s
		15	0.9583	2726.53s
		20	0.9634	3634.42s
		25	0.9698	4543.50s
		30	0.9597	5451.57s

In addition, increasing the number of layers of the employed models also increases the training time of models, as shown in Table 22. Traditionally, increasing the number of layers and neurons is aimed at learning intricate relationships between the input and output of the model, thereby improv-

ing the prediction performance of the model. However, this increase in the model's accuracy comes at the cost of higher computational complexity.

TABLE22: Comparison of model complexity vs accuracy.

Model	Layers	Epoch	Accuracy	Training time
CNN	6	30	0.9440	639.15s
	7		0.9511	728.11s
	8		0.9578	815.31s
LSTM	2	30	0.9651	3957.32s
	3		0.9701	4521.42s
	4		0.9753	4832.01s
RNN	2	30	0.9448	1851.48s
	3		0.9468	2031.847
	4		0.9586	2558.037
ACLR	3	30	0.9597	5451.57s
	4		0.9617	5641.21s
	5		0.9683	6043.32s

L. DISCUSSION

The research introduces ACLR, a hybrid deep learning model designed for the effective detection of botnet attacks in the IoT network. The innovative approach integrates four distinct models ANN, CNN, LSTM, and RNN through a stacking technique. These models synergize to form the powerful hybrid model, ACLR. Validation of ACLR's performance is conducted using k-fold cross-validation, resulting in a remarkable accuracy of 0.9749. Through a systematic evaluation using epochs ranging from 5 to 30, to proposed ACLR model demonstrated remarkable performance, achieving an impressive accuracy of 0.9698 without K-fold cross-validation. Evaluation metrics, including ROC-AUC and PR-AUC, have been used to evaluate the performance of the proposed research based on ACLR. The highest ROC-AUC is 0.9934 and PR-AUC is 0.9950 has been achieved. Comprehensive assessment parameters, encompassing accuracy, precision, recall, and F1 score, collectively emphasize the robustness of ACLR in the detection and mitigation of botnet threats within IoT environments. This research signifies a notable advancement in IoT security, offering an innovative and efficient hybrid machine-learning solution.

V. CONCLUSION

The frequency and intensity of cyber attacks have witnessed a growth lately and botnet attacks have emerged with the potential to cause serious damage. Deep learning-based models have shown potential for automated botnet detection; ensemble models come out as better predictors than individual models. This research proposes a hybrid stacking model, ANN+CNN+LSTM+RNN (ACLR) for botnet detection. The experimental setup involves ACLR implementation in the Google COLAB environment using the UNSW-NB15 dataset. In addition, this research employed ANN, CNN, LSTM, and RNN models for performance comparison with the proposed ACLR model. Experimental results suggest a superior performance of ACLR with a 0.9698 accuracy score while the k-fold cross-validation accuracy score is 0.9749 where the value of k is 3,5,7 and 10, respectively. In addition, increasing the number of layers for the deployed models is observed to produce better performance, however, it comes at the cost of increased computational complexity and higher training time. Among the employed models, the ANN model shows poor performance while LSTM, RNN, and CNN show better results. The comparative findings demonstrate that the proposed approach outperforms ANN, CNN, LSTM, and RNN models in terms of performance and accuracy for the detection of botnets. In comparison to previous models, the suggested ACLR model has the greatest ROC AUC (0.9934) and PR AUC (0.9950) values. Performance analysis with existing models indicates that ACLR can perform better than state-of-the-art models. It is important to note that deep learning algorithms for botnet identification still have limitations such as a lack of labeled statistics on training and the possibility of hostile attacks. To increase the precision, scalability, and robustness of deep learning-based botnet detection systems more research and development in this area are required. The stacking model in the proposed research has been based on four deep learning models which consume more time than the single model in predictions but show more efficient results than the single model. It also necessitates data interchange and synchronization. This demonstrates the significance of carefully balancing model complexity and effectiveness across a range of applications. As it will be totally automated in future research, there should be more training using reinforcement learning, which can be more effective.

ACKNOWLEDGMENT

The authors extend their appreciation to King Saud University for funding this research through Researchers Supporting Project Number (RSPD2024R890), King Saud University, Riyadh, Saudi Arabia.

REFERENCES

[1] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," Future Generation Computer Systems, vol. 100, pp. 779–796, 2019.

[2] O. Ibitoye, O. Shafiq, and A. Matrawy, "Analyzing adversarial attacks against deep learning for intrusion detection in iot networks," in 2019 IEEE global communications conference (GLOBECOM), pp. 1–6, IEEE, 2019.

[3] M. Shahhosseini, H. Mashayekhi, and M. Rezvani, "A deep learning approach for botnet detection using raw network traffic data," Journal of Network and Systems Management, vol. 30, no. 3, p. 44, 2022.

[4] S. Homayoun, M. Ahmadzadeh, S. Hashemi, A. Dehghantanha, and R. Khayami, "Botshark: A deep learning approach for botnet traffic detection," Cyber Threat Intelligence, pp. 137–153, 2018.

[5] M. Ge, X. Fu, N. Syed, Z. Baig, G. Teo, and A. Robles-Kelly, "Deep learning-based intrusion detection for iot networks," in 2019 IEEE 24th pacific rim international symposium on dependable computing (PRDC), pp. 256–25609, IEEE, 2019.

[6] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," Journal of Information Security and Applications, vol. 50, p. 102419, 2020.

[7] T. Hasan, J. Malik, I. Bibi, W. U. Khan, F. N. Al-Wesabi, K. Dev, and G. Huang, "Securing industrial internet of things against botnet attacks using hybrid deep learning approach," IEEE Transactions on Network Science and Engineering, 2022.

[8] D. T. Son, N. T. K. Tram, and P. M. Hieu, "Deep learning techniques to detect botnet," Journal of Science and Technology on Information security, vol. 1, no. 15, pp. 85–91, 2022.

[9] M. Gandhi and S. Srivatsa, "Detecting and preventing attacks using network intrusion detection systems," International Journal of Computer Science and Security, vol. 2, no. 1, pp. 49–60, 2008.

[10] J. Liu, S. Liu, and S. Zhang, "Detection of iot botnet based on deep learning," in 2019 Chinese control conference (CCC), pp. 8381–8385, IEEE, 2019.

[11] C. D. McDermott, F. Majdani, and A. V. Petrovski, "Botnet detection in the internet of things using deep learning approaches," in 2018 international joint conference on neural networks (IJCNN), pp. 1–8, IEEE, 2018.

[12] S. Sriram, R. Vinayakumar, M. Alazab, and K. Soman, "Network flow based iot botnet attack detection using deep learning," in IEEE INFOCOM 2020-IEEE conference on computer communications workshops (INFOCOM WKSHPS), pp. 189–194, IEEE, 2020.

[13] B. Nugraha, A. Nambiar, and T. Bauschert, "Performance evaluation of botnet detection using deep learning techniques," in 2020 11th International Conference on Network of the Future (NoF), pp. 141–149, IEEE, 2020.

[14] P. Karunakaran, "Deep learning approach to dga classification for effective cyber security," Journal of Ubiquitous Computing and Communication Technologies (UCCT), vol. 2, no. 04, pp. 203–213, 2020.

[15] N. Elsayed, Z. ElSayed, and M. Bayoumi, "Iot botnet detection using an economic deep learning model," arXiv preprint arXiv:2302.02013, 2023.

[16] M. A. Haq and M. A. Rahim Khan, "Dnnbot: Deep neural network-based botnet detection and classification.," Computers, Materials & Continua, vol. 71, no. 1, 2022.

[17] I. H. Sarker, "Deep cybersecurity: a comprehensive overview from neural network and deep learning perspective," SN Computer Science, vol. 2, no. 3, p. 154, 2021.

[18] A. A. Ahmed, W. A. Jabbar, A. S. Sadiq, and H. Patel, "Deep learning-based classification model for botnet attack detection," Journal of Ambient Intelligence and Humanized Computing, pp. 1–10, 2020.

[19] I. Letteri, M. Del Rosso, P. Caianiello, and D. Cassioli, "Performance of botnet detection by neural networks in software-defined networks.," in ITASEC, 2018.

[20] T. H. Aldhyani and H. Alkahtani, "Attacks to automatous vehicles: A deep learning algorithm for cybersecurity," Sensors, vol. 22, no. 1, p. 360, 2022.

[21] M. Y. Alzahrani and A. M. Bamhdi, "Hybrid deep-learning model to detect botnet attacks over internet of things environments," Soft Computing, vol. 26, no. 16, pp. 7721–7735, 2022.

[22] Y. N. Soe, P. I. Santosa, and R. Hartanto, "Ddos attack detection based on simple ann with smote for iot environment," in 2019 fourth international conference on informatics and computing (ICIC), pp. 1–5, IEEE, 2019.

[23] S.-C. Chen, Y.-R. Chen, and W.-G. Tzeng, "Effective botnet detection through neural networks on convolutional features," in 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), pp. 372–378, IEEE, 2018.

[24] S. I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, and H. Gacanin, "Hybrid deep learning for botnet attack detection in the internet-of-things networks," IEEE Internet of Things Journal, vol. 8, no. 6, pp. 4944–4956, 2020.

[25] S. Akarsh, S. Sriram, P. Poornachandran, V. K. Menon, and K. Soman, "Deep learning framework for domain generation algorithms prediction using long short-term memory," in 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS), pp. 666–671, IEEE, 2019.

[26] S. I. Popoola, B. Adebisi, M. Hammoudeh, H. Gacanin, and G. Gui, "Stacked recurrent neural network for botnet detection in smart homes," Computers & Electrical Engineering, vol. 92, p. 107039, 2021.

[27] M. Alauthman, "Botnet spam e-mail detection using deep recurrent neural network," Int. J, vol. 8, no. 5, pp. 1979–1986, 2020.

[28] J. Bhayo, S. A. Shah, S. Hameed, A. Ahmed, J. Nasir, and D. Draheim, "Towards a machine learning-based framework for ddos attack detection in software-defined iot (sd-iot) networks," Engineering Applications of Artificial Intelligence, vol. 123, p. 106432, 2023.

[29] S. Siddiqui, S. Hameed, S. A. Shah, I. Ahmad, A. Aneiba, D. Draheim, and S. Dustdar, "Towards software-defined networking-based iot frameworks: A systematic literature review, taxonomy, open challenges and prospects," IEEE Access, 2022.

[30] M. Khalid, S. Hameed, A. Qadir, S. A. Shah, and D. Draheim, "Towards sdn-based smart contract solution for iot access control," Computer Communications, vol. 198, pp. 1–31, 2023.

[31] A. Presekal, A. Ştefanov, V. S. Rajkumar, and P. Palensky, "Attack graph model for cyber-physical power systems using hybrid deep learning," IEEE Transactions on Smart Grid, 2023.

[32] C. Song, Y. Sun, G. Han, and J. J. Rodrigues, "Intrusion detection based on hybrid classifiers for smart grid," Computers & Electrical Engineering, vol. 93, p. 107212, 2021.

[33] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in 2015 military communications and information systems conference (MilCIS), pp. 1–6, IEEE, 2015.

[34] V. Timčenko and S. Gajin, "Machine learning based network anomaly detection for iot environments," in ICIST-2018 conference, 2018.

[35] M. Zeeshan, Q. Riaz, M. A. Bilal, M. K. Shahzad, H. Jabeen, S. A. Haider, and A. Rahim, "Protocol-based deep intrusion detection for dos and ddos attacks using unsw-nb15 and bot-iot data-sets," IEEE Access, vol. 10, pp. 2269–2283, 2021.

[36] M. Ahmad, Q. Riaz, M. Zeeshan, H. Tahir, S. A. Haider, and M. S. Khan, "Intrusion detection in internet of things using supervised machine learning based on application and transport layer features using unsw-nb15 data-set," EURASIP Journal on Wireless Communications and Networking, vol. 2021, no. 1, pp. 1–23, 2021.

[37] M. Thapliyal, A. Bijalwan, N. Garg, and E. S. Pilli, "A generic process model for botnet forensic analysis," in Conference on Advances in Communication and Control Systems (CAC2S 2013), pp. 98–102, Atlantis Press, 2013.

[38] F. Hussain, S. G. Abbas, U. U. Fayyaz, G. A. Shah, A. Toqeer, and A. Ali, "Towards a universal features set for iot botnet attacks detection," in 2020 IEEE 23rd International Multitopic Conference (INMIC), pp. 1–6, IEEE, 2020.

[39] H. N. Thanh and T. Van Lang, "Evaluating effectiveness of ensemble classifiers when detecting fuzzers attacks on the unsw-nb15 dataset," Journal of Computer Science and Cybernetics, vol. 36, no. 2, pp. 173–185, 2020.

[40] X. Gao, R. K. Saha, M. R. Prasad, and A. Roychoudhury, "Fuzz testing based data augmentation to improve robustness of deep neural networks," in Proceedings of the acm/ieee 42nd international conference on software engineering, pp. 1147–1158, 2020.

[41] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in 2015 military communications and information systems conference (MilCIS), pp. 1–6, IEEE, 2015.

[42] M. A. Ferrag, L. Shu, H. Djallel, and K.-K. R. Choo, "Deep learning-based intrusion detection for distributed denial of service attack in agriculture 4.0," Electronics, vol. 10, no. 11, p. 1257, 2021.

[43] Z. Al-Othman, M. Alkasassbeh, and S. A.-H. Baddar, "A state-of-the-art review on iot botnet attack detection," arXiv preprint arXiv:2010.13852, 2020.

[44] E. G. Dada, J. S. Bassi, H. Chiroma, A. O. Adetunmbi, O. E. Ajibuwa, et al., "Machine learning for email spam filtering: review, approaches and open research problems," Heliyon, vol. 5, no. 6, 2019.

[45] Y. Zhai, L. Yang, J. Yang, L. He, and Z. Li, "Baddga: Backdoor attack on lstm-based domain generation algorithm detector," Electronics, vol. 12, no. 3, p. 736, 2023.

[46] S. Soltani, S. A. H. Seno, M. Nezhadkamali, and R. Budiarto, "A survey on real world botnets and detection mechanisms," International Journal of Information and Network Security, vol. 3, no. 2, p. 116, 2014.

[47] R. U. Khan, X. Zhang, R. Kumar, A. Sharif, N. A. Golilarz, and M. Alazab, "An adaptive multi-layer botnet detection technique using machine learning classifiers," Applied Sciences, vol. 9, no. 11, p. 2375, 2019.

[48] T. A. Tuan, H. V. Long, L. H. Son, R. Kumar, I. Priyadarshini, and N. T. K. Son, "Performance evaluation of botnet ddos attack detection using machine learning," Evolutionary Intelligence, vol. 13, pp. 283–294, 2020.

[49] K. Jiang, W. Wang, A. Wang, and H. Wu, "Network intrusion detection combined hybrid sampling with deep hierarchical network," IEEE access, vol. 8, pp. 32464–32476, 2020.

[50] B. Bowen, A. Chennamaneni, A. Goulart, and D. Lin, "Blocnet: a hybrid, dataset-independent intrusion detection system using deep learning," International Journal of Information Security, pp. 1–25, 2023.

[51] N. Guizani and A. Ghafoor, "A network function virtualization system for detecting malware in large iot based networks," IEEE Journal on Selected Areas in Communications, vol. 38, no. 6, pp. 1218–1228, 2020.
[52] H. Zhu, X. Peng, and X. Gao, "Research on anomalous behavior detection of federated deep learning network intrusion based on fate-cnn," in Second International Symposium on Computer Applications and Information Systems (ISCAIS 2023), vol. 12721, pp. 136–142, SPIE, 2023.
[53] M. Azizjon, A. Jumabek, and W. Kim, "1d cnn based network intrusion detection with normalization on imbalanced data," in 2020 international conference on artificial intelligence in information and communication (ICAIIC), pp. 218–224, IEEE, 2020.
[54] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Shallow neural network with kernel approximation for prediction problems in highly demanding data networks," Expert Systems with Applications, vol. 124, pp. 196–208, 2019.
[55] R. A. Disha and S. Waheed, "Performance analysis of machine learning models for intrusion detection system using gini impurity-based weighted random forest (giwrf) feature selection technique," Cybersecurity, vol. 5, no. 1, p. 1, 2022.
[56] N. Elmrabit, F. Zhou, F. Li, and H. Zhou, "Evaluation of machine learning algorithms for anomaly detection," in 2020 international conference on cyber security and protection of digital services (cyber security), pp. 1–8, IEEE, 2020.
[57] A. Halbouni, T. S. Gunawan, M. H. Habaebi, M. Halbouni, M. Kartiwi, and R. Ahmad, "Cnn-lstm: hybrid deep neural network for network intrusion detection system," IEEE Access, vol. 10, pp. 99837–99849, 2022.

MUDASIR ALI pursuing his MS Computer Science degree from Department of Computer Science, The Islamia University of Bahawalpur, Bahawalpur, Pakistan. He has received a BSCS degree in 2021 from the Department of Computer Science, Institute of Southern Punjab, Multan. His current research interest includes Data mining, Artificial Intelligence, Cyber Security, Machine Learning and Deep Learning.

MOBEEN SHAHROZ pursuing his Ph.D. in Computer Science from The Islamia University of Bahawalpur. He received an MS Computer Science degree in 2020 and an MCS degree in 2018 from the Department of Computer Science, Khawaja Fareed University of Engineering and Information Technology(KFUEIT), Rahim Yar Khan, Pakistan. His current research interest includes the Internet of Things(IoT), Artificial Intelligence, Data mining, Natural Language Processing, Machine learning, Deep Learning, and Image classification.

MUHAMMAD FAHEEM MUSHTAQ received his Ph.D. degree from the Department of Information Security, Faculty of Computer Science and Information Technology, University Tun Hussein Onn Malaysia (UTHM), Malaysia, in 2018. He has made several contributions through research publications and book chapters toward Information Security. He earned his BS(IT) and MS(CS) degrees from The Islamia University of Bahawalpur, Punjab, Pakistan, in 2011 and 2013, respectively. He received Microsoft certifications of Internet Security and Acceleration (ISA) Server, Microsoft Certified Professional (MCP), and Microsoft Certified Technology Professional (MCTS) in 2010. He is currently working as Head, of the Department of Artificial Intelligence, The Islamia University Bahawalpur, Bahawalpur Pakistan. Previously, he is working as Head/Assistant Professor, at the Department of Information Technology, Khwaja Fareed University of Engineering and Information Technology, Rahim Yar Khan, Pakistan. He has been working as a Research Assistant during his Ph.D. degree from March 2016 to August 2018. His main research interest includes Information Security, Artificial Intelligence, Cognitive system, and applications.

SULTAN ALFARHOOD is an Assistant Professor in the Department of Computer Science at King Saud University (KSU). Since joining KSU in 2007, he has made several contributions to the field of computer science through his research and publications. Dr. Alfarhood holds a PhD in Computer Science from the University of Arkansas and has published several research papers on cutting-edge topics such as Machine Learning, Recommender Systems, Linked Open Data, and Text Mining. His work includes proposing innovative approaches and techniques to enhance the accuracy and effectiveness of recommender systems and sentiment analysis.

MEJDL SAFRAN is an Assistant Professor of Computer Science at King Saud University. He has been a faculty member since 2007 and holds a MSc. (2013) and a PhD (2018) in Computer Science from Southern Illinois University Carbondale. His research interests include computational intelligence, artificial intelligence, deep learning, pattern recognition, and predictive analytics. He has published articles in refereed journals and conference proceedings such as ACM Transactions on Information Systems, Applied Computing and Informatics, MDPI Biomedicine, MDPI Sensors, IEEE International Conference on Cluster, IEEE International Conference on Computer and Information Science, International Conference on Database Systems for Advanced Applications, and International Conference on Computational Science and Computational Intelligence. His current research focus includes developing efficient recommendation algorithms for large-scale systems, predictive models for online human activities, machine learning algorithms for performance management, and modeling and analyzing user behavior. Since 2018, he has been providing part-time consulting services in the field of artificial intelligence to private and public organizations and firms.

IMRAN ASHRAF received his Ph.D. in Information and Communication Engineering from Yeungnam University, South Korea in 2018, and the M.S. degree in computer science from the Blekinge Institute of Technology, Karlskrona, Sweden, in 2010 with distinction. He has worked as a postdoctoral fellow at Yeungnam University, as well. He is currently working as an Assistant Professor at the Information and Communication Engineering Department, at Yeungnam University, Gyeongsan, South Korea. His research areas include positioning using next-generation networks, communication in 5G and beyond, location-based services in wireless communication, smart sensors (LIDAR) for smart cars, and data analytics.