[1] **Deebalakshmi Ramalingam**

[2] **Balaji Ganesh Rajagopal**

[3] **Sushanth Chandra Addimulam**

[4] **Jyoti Kanjalkar**

[5] **Ajeet Kumar Vishwakarma**

A Supervised Hybrid DNN for Real-time Intrusion Detection in Firewalls

Journal of Electrical Systems

Abstract: - With networking infrastructure and communication technology developments, Internetworking has become inalienable to everyone's day-to-day lives. The volume of utilization of computer networking has increased due to various factors that have enabled the modernization of cyber-physical systems, which have a remarkable effect on the networks' security. Securing the cyber-physical systems has become imperative, as internet utilization has reached record levels in these pandemic times. Various experimental studies have been performed in the Intrusion Detection System using ensemble machine learning algorithms on benchmark datasets. This article proposes a hybrid approach for a Network Intrusion Detection system using an ensemble Deep Neural Network (DNN). The proposed architecture classifies the normal and intruder packets from the real-time network packet traces. The architecture is optimized for analyzing the packets in an online network that can instantaneously produce the packet data classification result. The proposed DLNN is evaluated with real-time network traces and benchmark datasets to prove concept reliability, and scalability measures. The F1-score for the various ensemble Machine Learning Techniques such as Decision Tree Classifiers, Random Forest Classifier, and XG Boost are in the range of 87%. Our proposed method produces an F1-score of 94.8% for the real-time packet traces.

Keywords: Ensemble DNN, High-speed network, Intrusion Detection System, ML Techniques, Network security, Real-time packet classification.

1. INTRODUCTION

Technological advancements in communication have caused a steep rise in network utilization, putting network security at the center of our concern. With the progress in cyber-physical systems, including the extensive use of cloud and grid computing technologies, there has been an upswing in the intruders' threat levels. While there is an evolution in technology and a transition from traditional software or systems to intelligent software systems, there is also an evolution in cyberattacks, with better hacking techniques and their propagation. As large companies shift toward remote working, cybercriminals have already started launching target attacks. Covid-19 related fraud reports have increased during this pandemic, leading to the proposal of this work. Several phishing e-mails related to lockdown were sent to the targeted end-users, leading to financial crime, money looting, etc. Many organizations worldwide are forced to deploy work from home in this pandemic situation, which hardens their network's Intrusion detection system against malicious activities and threats. Hence, there is a strong need for security tools such as a firewall, antivirus software, or an intrusion detection system to protect critical data/services from hackers or intruders. All organizations rely upon a single or multilevel firewall system that is insufficient to prevent a corporate network from all types of network attacks.

A firewall cannot defend the network against intrusion attempts on open ports required for network services.

[1] *Corresponding author: Assistant Professor, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Tiruchirappalli. Email: deebalakshmi.r@ist.srmtrichy.edu.in

[2] Assistant Professor, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Tiruchirappalli. Email: balajiganesh.r@ist.srmtrichy.edu.in

[3] Sr. Infrastructure and Security Engineer, Applied Computer Techniques (contract to University of Pittsburgh Medical Center), 28345 Beck Road STE 308, Wixom, MI- 48393. Email: sushanth93@gmail.com

[4] Assistant Professor, Vishwakarma Institute of Technology, Pune. Email: jyoti.kanjalkar@vit.edu

[5] Associate Professor, Department of Computer Science and Engineering, Dev Bhoomi Uttarakhand University, Dehradun, Uttarakhand, India. Email: ajeet7488@gmail.com

Hence, an Intrusion Detection System (IDS) is usually installed to complement the firewall. The complexity and frequent occurrence of imbalanced packet datasets indicate the need for extra research efforts. Computer Network Attacks (CNAs) [2] are security breaches that will be carried out to access confidential data or modify the secret data. CNA will take down the system under the intruder's control or use it as a botnet for performing Distributed Denial Of Service (DDoS) attacks. The active attackers use the information collected from a passive attacker or from any insider to attack the system to exploit the system or network by collecting, altering, or destroying the data available in the system or network. Suspicious network activity is a good source of information for detecting the intruder packet and its unintended action in the network. Suspicious network activity can refer to several different activities that involve abnormal access patterns, illegal network access, unauthorized user access, account abuse, file changes, and other unusual actions that either cause an attack or a data breach [1][2][3]. While the information security threat vector is highly complex and evolving rapidly, specific patterns and types of activity can be considered early warning signs for unauthorized access to the network [4][5]. Some of the notable signs are:

- Strange Patterns in User Access
- Abnormal Database Activities
- Mismatches between users and devices
- Change in the file configuration
- Changes During Scheduled Updates
- Misuse of Privileged account
- User Reports
- Unauthorized Port Access
- The inability of users to access

Detecting the intruder in complex, high-speed networking takes months or longer to detect a data breach from many organizations' system database. Hence, early detection is crucial to ensure that such an incident does not become a full-scale breach, leading to a massive loss for the organization. Most data breaches are complete in a few minutes; therefore, real-time detection is essential for the organization's safety [13]. An Intrusion detection system collects information in the form of packet traces from the network, analyses the packet traces for any threats or vulnerabilities, and logs off any untoward incidents. Such incidents can be classified as the most severe threat to Low threat. This classification helps the network system and users to mitigate the risk associated with the intruding packet. System breaches can be identified well in advance with modern deep learning techniques, which adhere to the dynamic attack types and gigabit volume of packet data in an extensive structured network system.

These system breaches lead to a hike in demand for a better and more accurate real-time intrusion or malware detection system driven by Deep Learning or more advanced techniques with less False Positive and False Negative. There should be a publicly available dataset for developing such an advanced system, closely resembling the real-time working computer network. Deep Learning has been widely used in the detection of network attacks. However, prominent study cases were not performed on a real-time intrusion detection system in online mode with a custom dataset. Different research articles like [6] present the real-time intrusion detection system with experimental results on two aspects of applying the online network intrusion detection system, which captures the packet data on its arrival and instantly analyses the packet any malicious information. The second aspect is the application of various machine learning algorithms. The algorithms used for Intrusion detection are hybrid or straightforward. The algorithms [5] were initially trained on the standard benchmark datasets to evolve the learning parameters. The learned model has been used for inference in real time. This workflow must address the attack types' dynamic nature and varying pace. Hence the work proposed in this paper embodies the application of a hybrid deep neural network for online network intrusion detection systems by incorporating;

- Real-time port mirrored data (Gigabit volume)
- Evolution of Hyperparameters for custom data
- Deep learning models to uncover the patterns in large and dynamic packet datasets.

The remaining parts of this paper are organized as follows. Section 2 presents the analysis of research experiments carried out in the design of Intrusion Detection Systems. Section 3 elaborates the hybrid deep neural network design by varying hyperparameters and structure with an insight into the analysis of different machine learning algorithms for IDS. Section 4 presents the experimental results with performance evaluation metrics for the proposed method on the real-time dataset. Section 5 delivers the future directions in the design of real-time IDS.

II. RELATED WORKS

Much of the experimental research [8][10] has been carried out in Network security for earmarking a better Intrusion detection system. However, most research works have been directed towards developing an intelligent model for the classification of the Intruder packet from the benchmarked datasets. The benchmarked datasets were recorded under a constrained environment with fixed parameters on the networking systems. The scenario in which the packet classification has been done still unknown, and the datasets don't correspond to the latest protocols and their datagram structures. In addition to human-made attacks, system-driven attacks and hacker programs' stability have increased significantly [12][18], which gives a vital sign of designing an AI-based Intrusion detection system to mitigate the problems caused by robotic hackers and intruders. The shortcomings of machine learning-based intrusion detection methods have made significant importance to appraise the existing internet security framework and its auxiliaries. In recent years, there has been growing attention in evolving insights into intrusion detection technologies powered with high-dimensional gigabit data processing, automatic feature extraction, and sustainable learning with dynamic patterns [9][11]. With all these put into effect, deep learning has become a widespread application in internet security, especially in Intrusion Detection System design.

Different algorithms are widely used in the development of Intrusion Detection Systems to improve accuracy. Various machine learning algorithms such as K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and K-means clustering were investigated [23][24] for the development of network intrusion detection, respectively. The application of particle swarm optimization with different experiments to improve the KNN algorithm's accuracy was presented in [23]. An IDS based on wavelets and AI Techniques for Knowledge discovery in packet datasets was proposed [26]. In other related works [25][28] hybrid machine learning approach allows sharing the best characteristics of each machine learning method [15][21]. Combining different machine learning techniques reduces false positives compared to the isolated use of single machine learning algorithms.

8A survey presented in [27] shows different intrusion detection system models, including RFC, the model was efficient with a low false alarm rate and high detection rate. In [29], a novel two-stage deep learning (TSDL) based approach to detect intrusion, and the authors investigate the impacts on the proposed model's performance. In cloud-based cyber-physical systems [22], features suitable for classification were extracted after preprocessing the collected data. And then, the classification process with a chunk of packets at a time has been done. The classification results are sent back while the classification for another chuck of packet data happens in the network port dynamically, which will help achieve real-time Intrusion detection during the adaptive inferencing process. This principle of a packet-based real-time intrusion detection system has been incorporated in our proposed work with the addition of an ensemble DLNN for better classification accuracy. The deep-learning approach's accuracy was compared against standard statistical machine learning algorithms, which are considered safe generalist choices for the design of intrusion detection systems [27].

A broad study [30], which tests 179 different machine learning classifiers across 121 datasets from different application areas, showed that random forest was the overall best performing technique. The random forest classifier with an average accuracy of 94% and reporting over 90% accuracy across 84% of the datasets, followed closely by SVM with an average of 92% accuracy. Random forest and SVM are commonly used in intrusion detection [30]. Hence, this paper compares the proposed ensemble-based deep learning approach with random forest, SVM classifiers, logistic regression, and decision trees to design cyber-physical intrusion detection systems. But with the continuous change in the volume of network traffic data, a non-linearity pattern is introduced in the input dataset. This non-linearity compels the researchers to move towards hybrid deep learning neural networks for feature engineering and classification. It is also inferred that the deep neural network's performance depends on the feature vector. There are chances that the feature vector might be redundant in most cases, which leads to the problem of introducing an optimization network. In [14][25][26], the curse of dimensionality has caused a negative effect on performance. The volume of the network data is non-linear and highly distorted in high dimensional space. The application of linear dimensionality reduction techniques again a cumbersome task and leads to a reduction in the helpful feature.

At the outset, many machine learning models have been developed to detect zero-day attacks. However, most approaches are suffering to identify and update the information about new attacks and result in low accuracy or high false-positive rate. In addition to this, the standard benchmark datasets are becoming very old to include recent malware activities. These benchmark datasets are no longer presented as candidates for evaluating real-time intrusion detection systems since they are publicly available [13]. Most of the available learning models are complicated in applying the Intrusion detection system in a real-time environment since the models use metadata in the packet. The models are reluctant to pay significant importance to the raw packet data. When the Metadata changes, continuous re-training is required, which is cumbersome in real-time situations. One of the significant assignments of Machine Learning is to develop a reasonable model from a dataset. Creating models from information is called learning or preparing, and the learned model can be called a hypothesis. The learning algorithms develop a set of classifiers and afterwards classify the recent data by deciding on their predictions are known as Ensemble Method. It has been found that ensembles are considered more accurate than the individual classifiers, which make them up. The ensemble method, otherwise called committee-based learning or learning different classifier frameworks, train various speculations to tackle a similar issue.

An ensemble contains various theory learners typically produced from a training algorithm by utilizing a base learning algorithm. The learning algorithms can be homogeneous ensembles in these processes, whereas the production will be heterogeneous ensembles. Ensemble methods aid in boosting the weak learners. The learning algorithms suffer from certain issues like statistical, computational and representational issues. These issues are measurable; however, the ensemble can overcome them as it reduces bias and variance. We focus our study on applying an ensemble model in a real-time environment with inputs from firewall agencies. The summary of the review is depicted in Table 1. Based on the analysis from the existing works, we inferred that a significant re-evaluation is needed to develop an intrusion detection system in two aspects;

- The necessity for a real-time network packet dataset for the re-evaluation of existing machine learning techniques.
- Design of hybrid deep layered network to address the non-linearity in the voluminous network packet data

Motivated by the above aspects, we have presented a novel deep learning framework for the design of hybrid deep layered networks and evaluate the model's performance concerning the real-time packet data.

Table 1. Summary of various research kinds of literature related to Intrusion Detection System

Ref. No.	Technique used	Parameters	Performance Evaluation
8	Intrusion Detection System (IDS), machine learning models	Losses, Accuracy, F1-score metrics.	The 36-features dataset gave slightly better accuracy, more accurate and less loss than the 23-features dataset.

10	data mining tools, Machine Learning	Attacks, detection capability, Attack features	Over a network dataset, low-frequency assaults employing machine learning approaches have been presented. As a result, by classifying numerous machine learning algorithms for each type of assault, the importance of a methodology for specific attacks has been demonstrated.
11	Software-Defined Networking, Machine Learning	Machine learning-based SDN security models	The threat models while paying attention to designing the ML solutions. Make the ML model auditable and follow a secure development process to Produce an initial operational cost model.
13	Intrusion Detection Systems (IDSs)	Payload, accuracy	Seven machine learning techniques were used to evaluate the performance and accuracy of the selected features. Finally, we compare the quality of the generated dataset to other publicly available datasets by looking for frequent errors and complaints in other synthetically constructed datasets using the 11 criteria of the dataset evaluation framework.
14	online class imbalance learning, re-sampling	Data streams, Concept drift	The mutual effect of applying class imbalance techniques and concept drift detection methods.
15	SVM method, 10-fold cross-validation	Accuracy and detection rate, False Alarm Rate.	K-Medoids, GFR, and the Nave Bayes classifier produced confusion matrix. This approach, it can be shown, performs better in terms of identifying Probe U2R and R2L attacks. The proposed technique also performed well in terms of accuracy, detection rate, and false alarm rate, according to the nave Bayes classifier.
18	Machine-learning (ML) algorithms, privacy-aware decision tree training algorithm	Random, baseline, ideal, Whitebox, Blackbox.	With no false positives, it can be used to infer sensitive responses from survey respondents. It can also extract images from facial recognition models that can be reliably re-identified by several expert individuals.
23	Random forest, voting, variational auto-encoder, stacking machine learning classifiers.	Accuracy, attack detection	The highest achieved accuracy was 99.99% with the original distribution, down-sampling approach, in RF. Compared to DNN, the achieved accuracy 99.30% for the up-sampling method.
24	The real-time intrusion detection system,	Accuracy, Information gain, Detection Rate	An IDS model that is simple enough to use with existing machine learning techniques. The 12 most important characteristics of DoS and Probe attacks. It could be creating a post-processing procedure to reduce the number of false alarms.
25	Intrusion	Datastream,	The DBN-SVM algorithm outperforms the

	Detection, Deep Belief Network and Support Vector Machine (DBN-SVM), integrated algorithm boosting and bagging methods.	classification accuracy	DBN, SVM, and other machine learning algorithms in terms of classification and detection accuracy, effectively enhancing classification accuracy and lowering false alarm rates.
27	Anomaly-based Intrusion Detection Systems, Signature-based Intrusion Detection Systems	Detection avoidance, more security	The contemporary models on the performance improvement of AIDS as a solution to overcome IDS issues. An effective IDS should detect different kinds of attacks accurately, including intrusions that incorporate evasion techniques.
28	Long short-term memory algorithm, simple recurrent unit-based (SRU)-based model	Accuracy, positive and negative learning samples	The SRU-DCGAN model, however, shows good test results in both KDD'99 and NSL-KDD datasets with high detection rates. It achieves 99.73% accuracy on the KDD'99 dataset and 99.62% on the NSL-KDD dataset.
30	Neural networks and boosting ensembles, random forest, support vector machine	Accuracy	Over the entire UCI machine learning classification database, the 179 classifiers are divided into 17 families. The 121 data sets represent the whole UCI data source as well as additional real-world challenges in order to draw significant conclusions regarding the behaviour of the classifiers that are independent of the data set collection.

III. METHODOLOGY

3.1 *Real-Time Intrusion Detection System*

An intrusion Detection System is one of the best methods for improving network security in Software Defined Networks (SDN). IDS is the logical complement of network firewalls and security management [3]. An IDS is a proactive intrusion detection tool that collects data information from various sources and use it to detect and classify intrusions, attacks, or violations of the security policies automatically at network-level and host-level infrastructure promptly. The primary function of IDS is to identify intrusive activities in a host or a network. Intrusive activity is any unauthorized access or modification to the system's information, thus compromising the confidentiality or integrity of the system [4]. Based on the behaviours, Intrusion Detection Systems are classified as perimeter-based intrusion detection system (PIDS), Virtual Machine based Intrusion detection system (VMIDS), Host-Based Intrusion Detection System (HIDS), and Network-Based Intrusion Detection System (NIDS). An intrusion detection system can use various methods for detection, which could be either anomaly-based or signature-based. An anomaly-based IDS establishes a baseline of all the regular activities occurring in the system and stores it in a database. Intrusion will be reported when it detects any movement not present in the database. Thus, an anomaly-based IDS is highly susceptible to false positives; IDS identifies regular packets as attack packets and generates an alert for vulnerability. On the other hand, the signature-based database stores the signatures of attacks in a database and uses signatures to detect an attack. Thus, a signature-based IDS is highly susceptible to false negatives, which will not identify specific attacks if these attacks' signature is not in its database.

We have collected the real-time packet data using port mirroring in a node attached to the sonic firewall in this work. The sonic firewall uses the real-time Intrusion Detection System for network packet classification using pre-defined libraries. The sonic firewall enforces real-time packet intrusion detection using packet reassembly, Deep packet inspection, and finally, signature matching. This sonic firewall enables dynamic signature matching pushed by Distributed Enforcement Architecture, which is updated globally. The packet data in varying time slices in the peak hour network traffic has been used to evaluate the deep neural models. Since the developed models are adaptive to the dynamic attack types, the model can learn quickly about the intruding packet's Metadata. We have highlighted that the proposed hybrid deep layer network follows feature extraction, pattern matching, and classification that automatically generates the feature of the intruder pattern on supervised learning. The proposed ensemble-based deep learning approach will enhance the reliability of the real-time intrusion detection system.

3.2 Public Dataset

The objective of this article is to develop a deep learning-based model for an Intrusion detection system. For proof of concept and systematic analysis, experiments are carried out with Machine Learning algorithms and Deep Learning algorithms over Standard benchmark datasets and real-time datasets. First, the KDD '99 cup, NSL-KDD, CICIDS-2017 datasets are considered for the experimentation of ML algorithms and Deep Learning models. Then the tuned algorithms and models are used to evaluate the effectiveness with a real-time dataset.

3.2.1 KDD 99 Cup Dataset

KDD CUP 99 dataset [4] is a widely used benchmark dataset for the intrusion detection system for software-defined networks. In 1999, Advanced Research Project Agency (MIT Lincoln Laboratory) (DARPA) [8] created the KDD Cup Dataset by processing the TCP dump data obtained from the Defense. This dataset was used in the Third International Knowledge Discovery and Data Mining Tools Competition. This dataset contains 41 features [6], the four primary attacks' categories mentioned in the KDD are Denial of Service (DoS), Probe, Remote to Local Attack (R2L), and User to Root Attack (U2R). This data set consists of three components: "10% KDD", "Corrected KDD," and "Whole KDD" [8][20]. This dataset is publically available and contains around 5 Million data points. The drawback of this dataset is the duplicity in the packets being used. Table 2 and Table 3 show the redundancy in the dataset. The redundancy has caused the learning algorithms, mainly Machine Learning Algorithm, to be biased towards the frequent attacks and prevent it from learning infrequent attacks, which can be harmful to the networks [17]. This redundancy is in and testing data, which results in biased and skewed evaluation results.

Table 2. Statistics of Redundant Records in the KDD Train Dataset

	Training dataset	Distinct Record	Reduction ratio
Normal	972,781	812,814	16.44%
Attack	3,925,650	262,178	93.32%
Total	4,898,431	1,074,992	78.05 %

Table 3. Statistics of Redundant Records in the KDD Test Dataset

	Testing dataset	Distinct Record	Reduction ratio
Normal	60,591	47,911	20.92%

Attack	250,436	29,378	88.26%
Total	311,027	77,289	75.15%

Table 4. Training and Testing Dataset for Model Evaluation of KDD'99

	Training	Testing
Normal	97278	60593
Attack	396743	250436
Total	494021	311029

3.2.2 NSL-KDD Dataset

NSL KDD is also a publicly available dataset created using the KDD'99 dataset [5][8] in 2009. It contains 150K data points as the dataset's size is compared to 5 Million points in the case of the NSL-KDD Dataset. The main objective in creating this dataset was the large number of redundant records found in the KDD'99 dataset. Redundant records generally led to biased evaluation results when Machine Learning Algorithms are applied to them, leading to a false conclusion. The work [10] improved KDD Dataset and published a new dataset called NSL-KDD, which is considered the updated version of the KDD'99 Dataset and is given in Table 4. The Duplicate records were removed in both training and testing datasets and made it more practical. The whole dataset could be used for training and testing instead of using the dataset partly in KDD'99 Dataset. Like in KDD'99, it also provides two sets of training sets and testing sets, which are complete datasets, and the other includes only a 20% subsample of the entire training set. Table 5 and Table 6 show the detailed analysis of the training and testing dataset for NSL-KDD.

But the main drawback for this dataset is that this dataset also does not represent real-time existing network traffic profile characteristics and is not suitable for network-based Intrusion detection. Furthermore, like KDD'99 data, this dataset is outdated and cannot be used to identify new malware attacks.

Table 5. Training and Testing Records for NSL-KDD Dataset

	Training	**Training (20%)**	**Testing**	**Testing (20%)**
Normal	67,343	13449	9,710	2152
Attack	58630	11743	12834	9697
Total	125973	25192	22544	11849

Table 6. Training and Testing Dataset for Model Evaluation of NSL-KDD

	Training	**Testing**
Normal	67343	9,710
Attack	58630	12834
Total	125973	22544

3.2.3 CICDS Dataset

This dataset is developed by the Canadian Institute of Cybersecurity (CIC) [7], which is considered the most up to date dataset that depicts real-time network traffic as claimed by the author and is regarded as one of the best benchmark IDS Dataset. This dataset contained 3.1 Million flows of packets and emulated in a small network [9]. This dataset includes the seven most common types of attack families: Brute Force Attack, Botnet, Heatbleed Attack, DoS Attack, DDoS Attack, Web Attack, and Infiltration Attack. Network traffic was collected for five days from Monday to Friday, with benign activity on the first day and attack being injected in the rest of the days. Hence, this dataset comprises eight small datasets with different types of attacks occurring each day. Table 6 shows the detailed analysis of the total records containing all eight datasets for the CICIDS Dataset.

A large number of missing data is the drawback of this CICDS dataset. According to a survey conducted [7][9], around 288602 instances of the missing class label and 203 missing information. Also, the dataset is divided into eight small datasets, and each dataset is of ample size. However, the main drawback is that the dataset is prone to high-class imbalance [9][14]; this often results in a bias towards the majority class by the detector [16][19]. The majority class percentage is about 83.34%, but one minority class is only 0.00039% [17]. Due to the vast difference in the majority and minority prevalence rate, the detector will be biased toward the larger class. If a random sample of the dataset is used for training and testing the data, there is a considerable probability that it will not detect this minority class. Hence the model will fail to see such an attack when an instance of such attack arrives [18].

3.3 NITT-IDS Dataset (Custom Dataset)

The National Institute of Technology Tiruchirappalli (NITT) campus is powered by Black Diamond X8 9OCore Switch (Extreme Networks), which can support 40 Gbps network bandwidth connectivity and 100 numbers of layer three switches. The Three tire network architecture has 3 Gbps internet connectivity via three Government Internet Service Providers. Two (active-active) numbers of Sonic Wall NSSP 12400 High-End Unified Threat Management (UTM) and two (active-active) numbers of Sonic Wall ESA 9000 E-mail appliances with high availability with automatic failover provides Network security. Active Directory Service linked with User authentication. All the Wi-Fi devices are 3 x 3 MIMO Wireless Access points and connected through high-end Wi-Fi controllers. All network resources of the department are independent of the geographic location of the devices. The devices are bound by Virtual Local Area Network (VLAN) to improve security, scalability, and manageability. Three edge switches in the manageable layer, managed from Epicenter (NMS) software and the Campus bandwidth monitored using PRTG and MRTG. The most important and valuable data Servers are in DMZ. More than 10000 users of the institute can access the Servers in DMZ via UTM.

Network Packet data has been collected from the Institute network and contains 1791 unique IP source addresses and 3028 unique IP destination addresses. The dataset was created in the year 2019 during the covid-19 pandemic period. The network architecture in Figure 1 shows how packet data has been collected in the Institute network using high-end workstations attached to the UTM appliances.

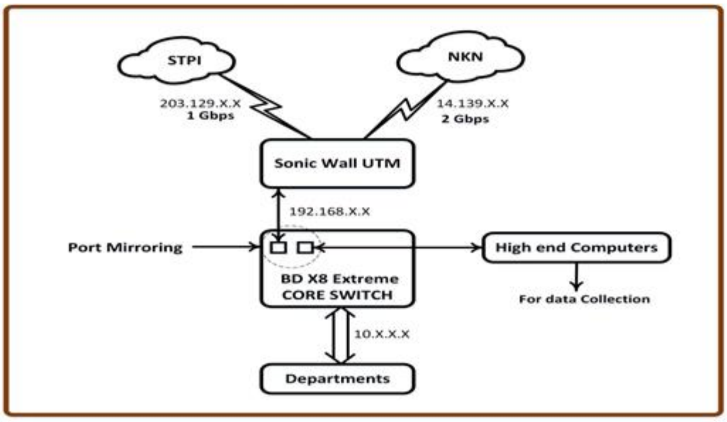


Figure 1. The network structure in which packets captured from UTM appliances for the design of IDS

The network traffic data is usually collected in two formats: Packet-based data and Flow-based data. Packet-based data are in pcap format and most collected from network monitoring tools like Wireshark. These data contain both payloads and Metadata, which includes information on protocols and used network. Flow-based data is a more condensed and aggregated form of packet-based data. Packet-based data has Metadata but no payloads about the network used. Hence packet-based data are considered more useful than flow-based data. Interpretation of content related to flow-based and packet-based traffic is quite difficult for third parties to interpret. The Metadata also provides information about network structure, transport protocols, and IP address, which contains essential sources for detecting intruder packets. Our custom dataset also provides additional anonymity information like attributes of the captured packets, payload information. The dataset was collected in a real-time production environment, which helps the design of Intrusion detection systems. The comparative analysis of the different benchmark datasets and the custom NITT dataset is detailed in Table 7.

Table 7. Review of the Various Benchmark Datasets and the Custom Dataset

Data set	Year	Format	Metadata	Network type
KDD'99	1998	Flow-based	No	Emulated
NSL-KDD	1998	Flow-based	No	Emulated
CICIDS	2017	Packet-based, flow-based	Yes	Emulated
NITT-IDS (Custom dataset)	2019	Packet-based	Yes	Real

Our dataset contains 83 different attributes, and some of the attributes and the average packet size of protocols are given in Table 8 and Figure 2, respectively. Our dataset has a total of 14.8 million records distributed into eight segments. Each segment was captured in our Institute network for 20 seconds.

Table 8. Some of the Attributes of the captured network dataset

Attribute	Type
Duration (in milliseconds)	Double
Content-Type	String
Connection	String
Frame length on the wire	Integer
Frame length – capture file	Integer
Header length	Integer
Fragment	Integer
Time to live	Integer
TCP payload	String
Protocol	String

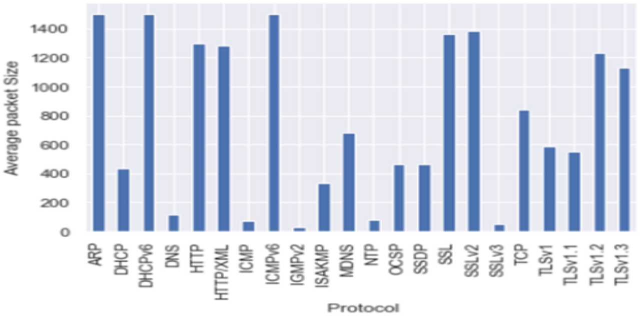


Figure 2. The average packet size of various protocols provides information to the Intrusion Detection System

3.4 Machine Learning Algorithm for IDS

IDS are considered a classification problem in the development of the ML-based system. The classification problem focuses on developing an optimal model for classifying the packets into either normal or abnormal. Research works exploring ML algorithms' development for IDS used the benchmark datasets to evaluate the Machine Learning models as a holistic approach [10]. However, in the real-time scenario, the Intrusion Detection System in SDN plays a c1rucial role in evaluating the IDS. In this article, three machine learning algorithms, such as the Decision Tree classifier, Random Forest Classifier, XG Boost classifier, are included for completeness and systematic analysis of the model with the benchmark datasets and real-time dataset. This analysis also helped in the experimental design of a DLNN for the datasets.

3.4.1. Decision Tree Classifier (DTC)

A decision tree is a tree-based supervised classifier algorithm that builds the classification model by creating a decision tree. The Decision Tree is similar to a flowchart that acquires knowledge based on information gain or entropy. DTC is based on the CART Algorithm in which each node splits the current set of training examples into a smaller subset based on the best-fitted features or attributes. It works very accurately for lesser features, but with more features, when the tree becomes very large, its performance decreases due to over-fitting. Decision tree pruning is done to avoid overfitting. In this work, while considering the dataset's size, entropy was selected as the criterion to measure the quality of split instead of Gini impurity ('Gini') to improvise the accuracy. The maximum depth of the tree ('max_depth') was tuned from 4 to 10 in the steps of 1. The 'max_depth' parameter was set to eight, which resulted in better accuracy with less processing time.

3.4.2. Random Forest Classifier (RFC)

Random Forest Classifier (RFC) is an ensemble estimator that uses multiple decision tree classifiers and applies it to each dataset subset. Then the classifier takes the mean of all the decision tree classifier predictions to provide the final classification results with improved accuracy. RFC is an efficient technique to decrease the variance and control the over-fitting caused in the decision tree classifier. In this work, the value of 'n_estimators,' i.e., the number of trees in the forest, has been made on a trial basis from 100 to 1500 in the steps of 100. We have inferred that the accuracy starts to improve on increasing the value of 'n_estimators' from 100 to 1200. Once the number of estimators increased above 1200, there was a very slight change in precision with increased proc11essing time. Hence the optimum value for 'n_estimators' has been set as 1200. The information gain ('entropy') was taken as the criterion to measure the split quality due to its better accuracy. For the maximum depth of the tree ('max_depth'), its value was tuned from 4 to 10 in the steps of 1. The 'max_depth' was set to eight due to better accuracy and less processing time. The minimum number of samples required to split the internal node ('min_sample_split') was tuned from 2 to 5 in the steps of 1. The value for the 'min_sample_split' parameter was set as three due to better accuracy and less processing time.

3.4.3. XG Boost Classifier

XGBoost stands for Extreme Gradient Boosting, which works on a Gradient Boosting algorithm based on the decision tree. The XGBoost Classifier shows much better performance in structured data. Random Forest builds fully-fledged decision trees on the subset of the dataset in parallel. Each tree is not generalized; that is, they have high variance and are highly specialized in predicting its subset by decreasing bias. The predictions made by each separate decision tree will be combined by reducing variance and bias, resulting in an excellent performance by Random Forest Classifier.

On the other hand, the XGBoost classifier builds short and simple decision trees repetitively. In this, each tree is being considered a weak learner due to its high bias. It first creates a simple tree that shows poor performance. Then, another tree is built, which is also a weak learner and learned to predict what the first tree could not predict. Thus, this algorithm sequentially creates weaker learners where each one tries to predict what the previous tree could not optimally. The algorithm runs until the specified stopping condition, based on the number of estimators or trees built. It also has an advantage over Random Forest Classifier due to its speed and flexibility. Furthermore, XGBoost Algorithm parallelizes each decision tree's training instead of building it one by one sequentially, as in Gradient Boosting, making the training faster. Due to its optimal approach, it is considered as the gradient boosting algorithm. Results of different Machine Learning Algorithms concerning various benchmarked datasets and NITT-IDS datasets are given in Tables 9, 10, 11, and 12, respectively. This analysis helps to experiment with the different Deep Learning models to develop an Intrusion Detection System.

Table 9. Results of Machine Learning Algorithms for KDD'99 Dataset

Algorithm	Accuracy	Precision	Recall	F1-score	Training time (s)
DTC	91.9	99.6	91	95.1	28.12
RFC	92.8	99.5	91.7	95.4	997.23
XGB	92.1	99.5	91.2	95.1	953.46

Table 10. Results of Machine Learning Algorithms for NSL-KDD Dataset

Algorithm	Accuracy	Precision	Recall	F1-score	Training time (s)
DTC	92.9	92.7	94.4	93.4	15.18 sec
RFC	93	94.8	92.1	93.4	294.56 sec
XGB	93.8	93.7	94.5	94.1	344.54 sec

Table 11. Results of Machine Learning Algorithms for CICDS Dataset

Algorithm	Accuracy	Precision	Recall	F1-score	Training time (s)
DTC	90.5	85.4	95.7	90.2	18.12
RFC	91.1	86.9	95.3	90.9	410.23
XGB	92.1	88.4	95.6	91.8	431.46

Table 12. Results of Machine Learning Algorithms for NITT-IDS Dataset

Algorithm	Accuracy	Precision	Recall	F1-score	Training time (s)
DTC	87	86.6	87	86.8	3.54
RFC	86.9	87.7	86.9	87.3	586.38
XGB	86.9	87.2	86.9	87	597.58

From these results, we infer that XGBoost and RF Classifiers outperform well in both the Standard Benchmark datasets and Real-time datasets. Therefore, our NITT-IDS dataset can also be considered a real-time benchmark dataset for the design and evaluation of Intrusion Detection Systems. Further, our custom dataset can also be used to design and evaluate the performance of Intrusion Detection Systems based on Deep Learning Algorithms.

3.5 Deep Learning Architectures for IDS

Deep learning architectures provide better classification and prediction models for different data types ranging from two-dimensional numerical data to multi-dimensional image and video data. Deep learning architectures' power depends on the varying hyper-parameters that produce non-linearity in the data and its features. The non-linearity provides increased flexibility and permit the deep neural network to scale in proportion to training data. However, the downside of the DNN is that it gives different weights in different iterations of training and produces different predictions, which are considered high variance. Therefore, the performance of those deep learning architectures can be further enhanced by training multiple deep neural networks and combining the prediction results to get the final predicted class label. Combining the performance of multiple deep neural networks is referred to as hybrid deep-layer neural networks. Ensemble models' results are less sensitive to the specifics of the dataset and are best suited for training and prediction of custom datasets like the NITT-IDS dataset.

3.5.1. Deep Neural Networks

For hybrid deep neural networks, hyperparameter tuning is an important task that decides the classification results' performance metrics. The different hyper-parameters like learning rate, number of hidden layers, activation functions, epoch, dropout, kernel_initializer were optimally tuned for the NITT-IDS custom dataset. Five different deep neural networks, namely DNN1 to DNN5, as shown in Figure 3, were designed with 70 units in the input layer and one neuron in the output layer for identifying whether the packet is normal or anomalous with a fully connected connection between all layers. These five DNNs were customized with one to five hidden layers, each with 1024 to 128 hidden units. The hidden layers in the hidden layers were calculated by experimenting with 2048 units in the hidden layer, which resulted in increased processing time and reduced performance. By analyzing the experimental results, it was identified that 1024 could be taken as ideal hidden units. The optimal value for the number of epochs was arrived at by conducting experimental trials. The five hidden layers were taken into consideration, and 300 epochs were run. It was found out that the model showed an early stopping at 150 epochs. The early stopping signifies that DNN could learn the patterns of normal packets with 150 epochs than those with anomalous packets. For capturing the significant features or attributes that can differentiate between normal packet and anomalous packet, 150 epochs were required. After 150 epochs, the performance of our model started diminishing due to over-fitting. However, the number of epochs was found to be varying with the number of units in DNN.

For finding the optimal value of the learning rate, two trials were conducted with 300 epochs using a learning rate between 0.05 and 0.2 with 1024 hidden units. From experimental results, it was concluded that DNN works better with the learning rate of 0.1. Also, the reduction in the learning rate to 0.05 consumed more training time and did not improve the accuracy; increasing the learning rate to 0.2 reduces the performance of the model. Hence learning rate has been selected as 0.1 for the deep neural network. Two trials of the experiment were conducted with 300 epochs, a learning rate of 0.1, and a batch size of 128 for optimizing the kernel_initializer using 'zero' 'random' 'normal' 'uniform' 'he_normal' initialization with a medium-size DNN consisting of 1024 hidden units. Our experimental results infer that the 'he_normal' initialization technique works better for weight initialization. Two trials were conducted with 300 epochs, learning rate as 0.1, batch size of 128, and kernel_initializer as he_normal initialization for optimizing the activation function with tanh, sigmoid or ReLU in the hidden layers. It was found out that the tanh activation function and sigmoid activation function showed a weak performance due to the vanishing gradient problem. Hence ReLU was considered as the optimal activation function for the hidden layer. Since binary classification is required, the sigmoid activation function for the output layer was adopted.

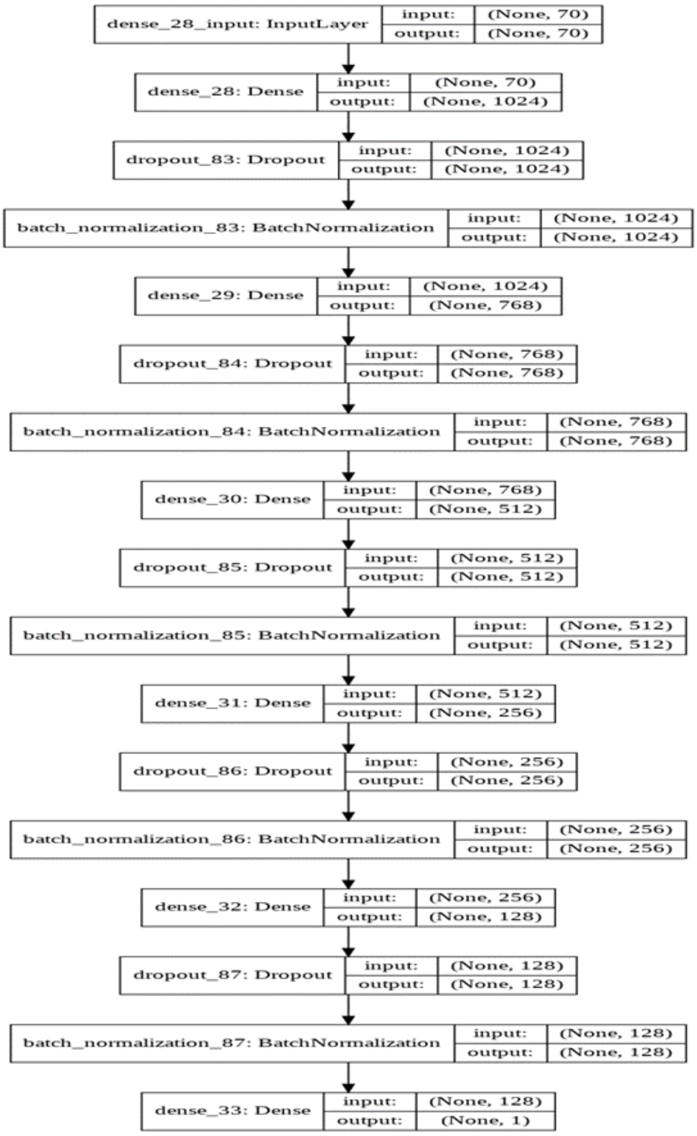


Figure 3. The architecture of the Deep Neural Network DNN1 to DNN5

Similarly, the dropout value was trialed between 0.05 and 0.2. The disharmony between Dropout and Batch Normalization was avoided by choosing the least value of dropout 0.1 for the model training. The specificity of the model has been reduced. Five DNN networks were designed for experiments with varying numbers of hidden layers from 1 to 5. In DNN-1, only one layer with 1024 hidden units was used, but in DNN-5, five hidden layers, each with 1024 units, were used.

3.6 Ensemble Deep Neural Network

3.6.1. Modified Residual LSTM+RNN

A Recurrent Neural Network is the extension of a Neural Network which makes use of sequential information. The Algorithm behind RNN is that every next step's input uses the information from the previous step, which is the output of the last step and the current step input. RNN uses the Back Propagation Through Time (BPTT) algorithm for training the data. However, due to the vanishing gradient problem [16], a new concept called LSTM replaces the traditional RNN. LSTM is a modified version of RNN that has three gates to update the cell state, which carries the relevant information throughout the sequence. It has three gates, such as the forget gate, the input gate, and the output gate. These three gates decide what relevant information from the previous step is to be kept, what new information to be added, and the next state. The hyperparameter for the modified residual

LSTM+RNN network is shown in Figure 4. The network was optimized for the NITT-IDS dataset. A residual network is added along with the LSTM+RNN model to improve the performance by further avoiding the vanishing gradient problem.

Our NITT-IDS dataset was fed into the modified LSTM+RNN network with 70 units in the input and output layers. The non-linearity is introduced in the network with three hidden layers with 256 units in each. The addition of a fourth hidden layer, either with 256 or 1024 units resulting in performance loss. The model was trained for 300 epochs, but the model showed an early stopping at 100 epochs, after which there was no significant improvement in the accuracy. Two trials were conducted for 100 epochs, with learning rates in the range of 0.01 and 0.1. From the experimental results, we found that the learning rate to be 0.05. Similarly kernel_initializer was set to he_normal, after experimenting with 'zero', 'random', 'normal', 'uniform' kernel_initializers. For the batch size of 128, the ReLU activation function was used in the hidden layers, and the sigmoid activation function was used in the output layer. The dropout value was set to 0.05 to balance the disharmony between Drop out and Batch normalization.

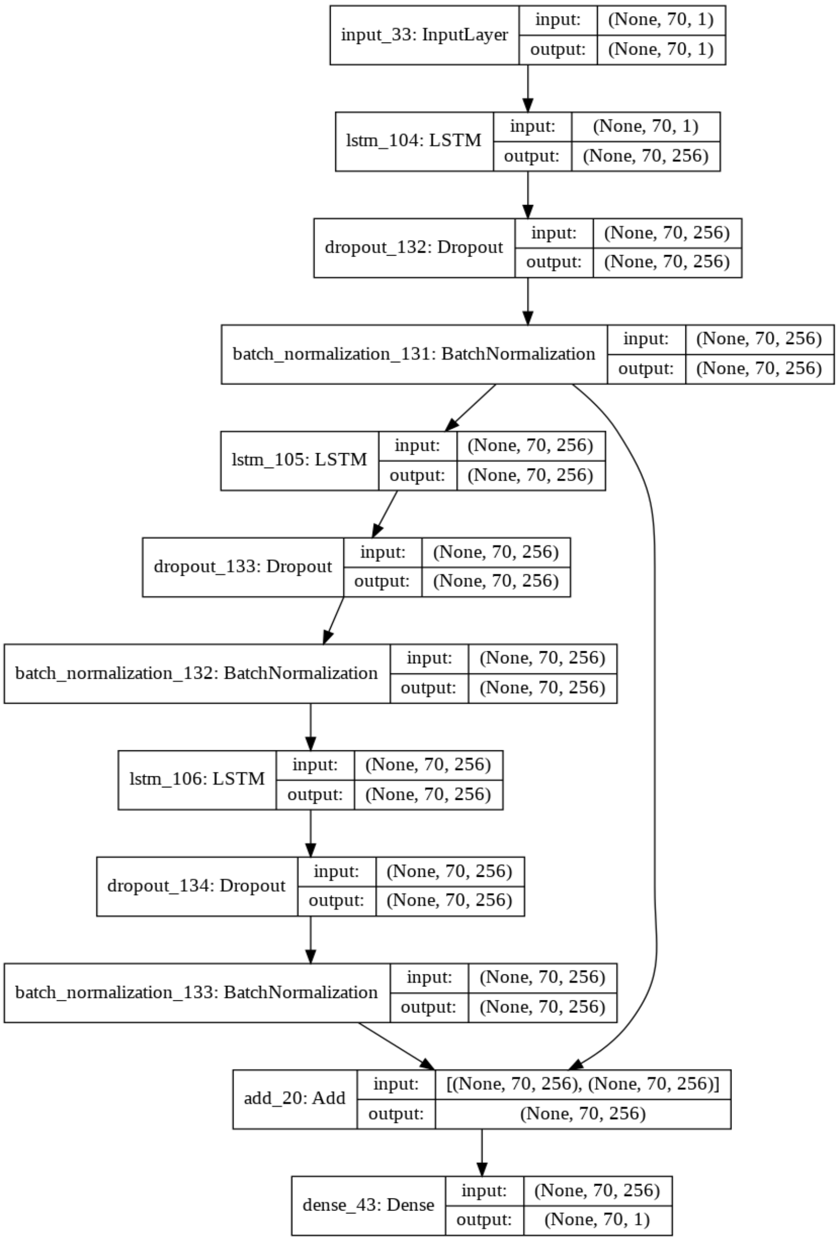


Figure 4. The architecture of the Modified Residual Lstm+Rnn Model

3.6.2. CNN+LSTM model without Residual

A CNN+LSTM+RNN model can be defined by the repeatable addition of a CNN layer followed by an LSTM and layer on the output. This architecture uses two sub-models, both CNN and LSTM Model for feature extraction and RNN model, to interpret each step's features. An optimal hyperparameter for the model has been obtained for the NITT-IDS dataset. In the model shown in Figure 5, the residual connection has been included to increase the accuracy and reduce training and testing time. In this model, the LSTM (dim_number, return_sequence = False) (input). Since the return_sequences are False, it will return the output in the output sequence. From model figure 5, we could also see that when the input of 2D tensor of shape (17,256) is given to LSTM, it returns a 1D tensor of shape (128).

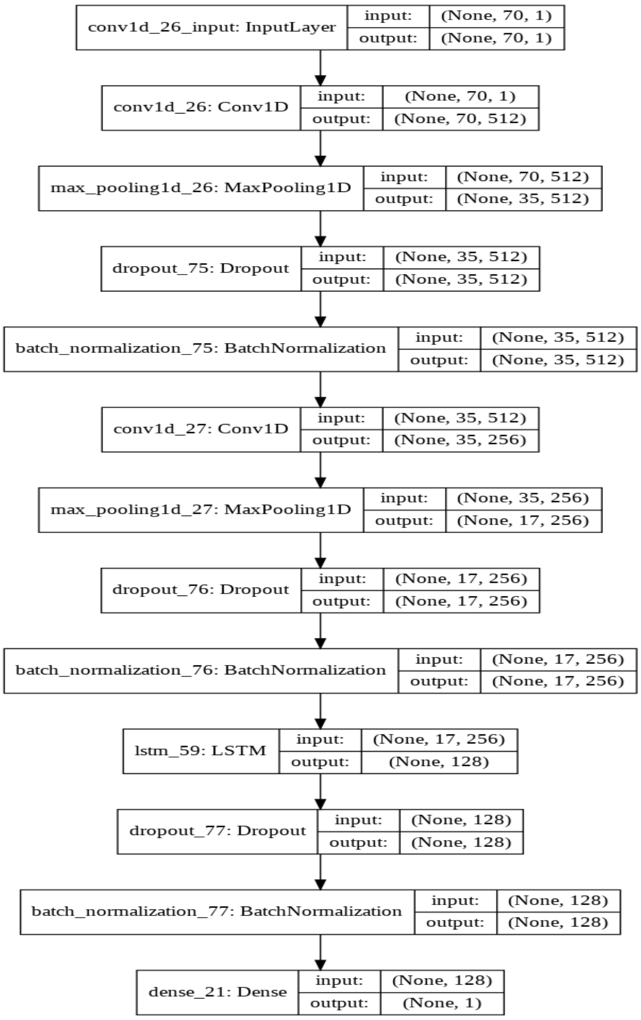

conv1d_26_input: InputLayer | input: | (None, 70, 1) | output: | (None, 70, 1)
conv1d_26: Conv1D | input: | (None, 70, 1) | output: | (None, 70, 512)
max_pooling1d_26: MaxPooling1D | input: | (None, 70, 512) | output: | (None, 35, 512)
dropout_75: Dropout | input: | (None, 35, 512) | output: | (None, 35, 512)
batch_normalization_75: BatchNormalization | input: | (None, 35, 512) | output: | (None, 35, 512)
conv1d_27: Conv1D | input: | (None, 35, 512) | output: | (None, 35, 256)
max_pooling1d_27: MaxPooling1D | input: | (None, 35, 256) | output: | (None, 17, 256)
dropout_76: Dropout | input: | (None, 17, 256) | output: | (None, 17, 256)
batch_normalization_76: BatchNormalization | input: | (None, 17, 256) | output: | (None, 17, 256)
lstm_59: LSTM | input: | (None, 17, 256) | output: | (None, 128)
dropout_77: Dropout | input: | (None, 128) | output: | (None, 128)
batch_normalization_77: BatchNormalization | input: | (None, 128) | output: | (None, 128)
dense_21: Dense | input: | (None, 128) | output: | (None, 1)

Figure 5. Deep Neural Network with the combination of two Convnets and one LSTM layer without residual

3.6.3. CNN+LSTM+RNN Model With Residual Network

A CNN+LSTM+RNN model with a residual network is shown in Figure 6. This model can be defined by adding two CNN layers, followed by a single LSTM layer on the output. This architecture uses two sub-models, the CNN models for feature extraction and the LSTM model, to interpret the attributes across each time step. The optimal parameters of the CNN+LSTM model for the NITT-IDS dataset have experimented on a trial basis for Hidden units, Kernel size, Max pooling size, and other hyperparameters. There are three hidden layers in the CNN+LSTM models, with two convent layers and one LSTM layer. Two convents were designed with 512 and 256 hidden units, whereas the LSTM hidden layer has 128 hidden units. Two trials were conducted for optimizing kernel size and max pooling. The model has been trained with the hyperparameter, as described in Table 13. In the CNN+LSTM+RNN model in Figure 6, the LSTM have been configured as LSTM (dim_number, return_sequence = True) (input). The return_sequences is true, which will return the full sequence in the output sequence.

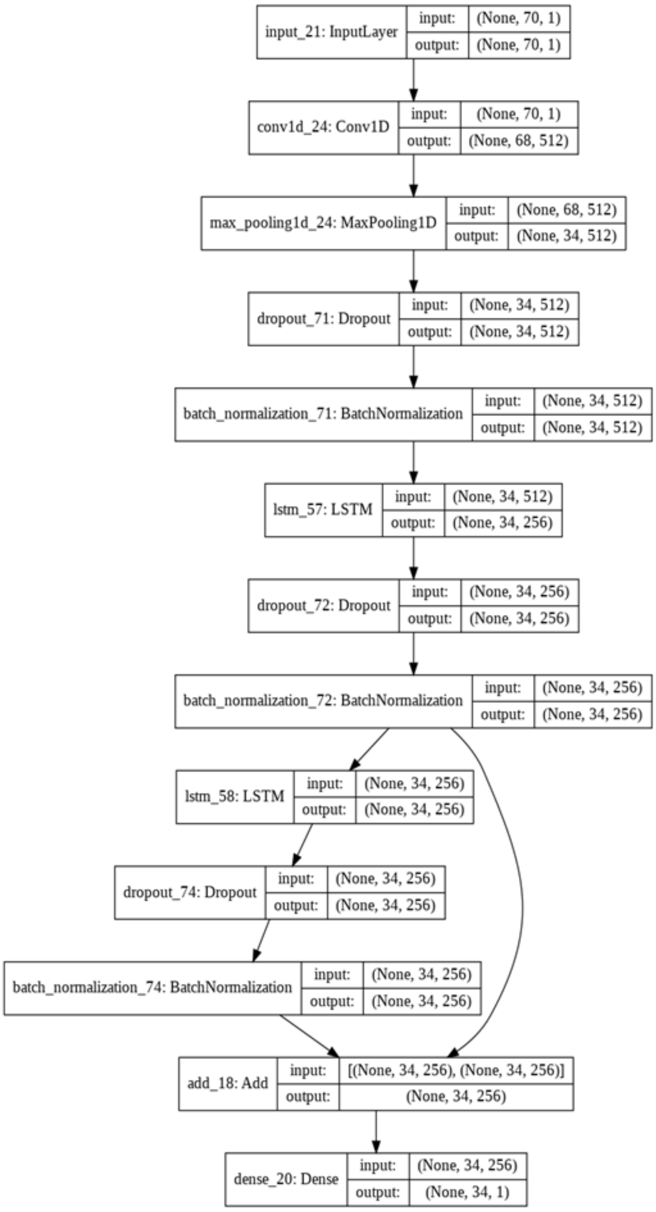


Figure 6. Deep Neural Network with the combination of two convent layers and one LSTM with Residual

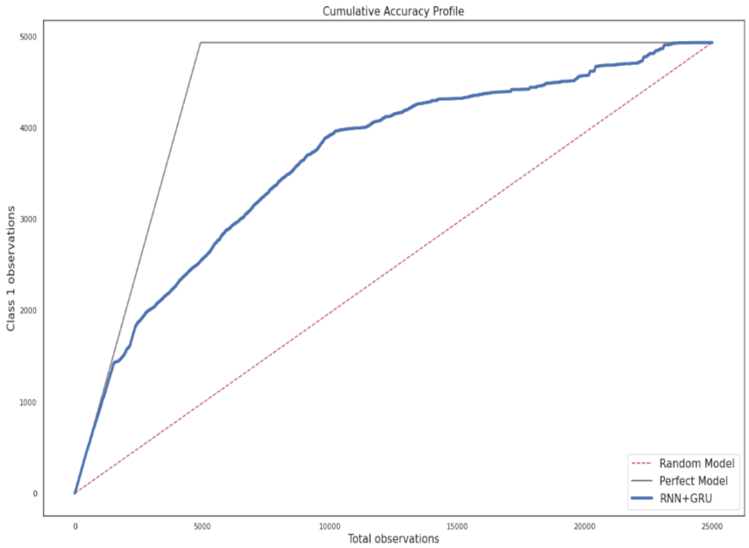
Table 13. Different hyperparameters for the CNN+LSTM+RNN model for the NITT-IDS dataset

Hyperparameter	**Value**
Learning Rate	0.05
Kernel size	3
Max Pooling	2
Batch Size	128
Drop out	0.2
Batch Normalization	Yes

Kernel_initializer	he_normal
Activation	ReLU in hidden layers Sigmoid in the output layer
Hidden units for Convnets	512, 256
Hidden units for LSTM	128

IV. EXPERIMENTAL RESULTS

The various hybrid deep learning models described in the previous section have experimented with the standard benchmark datasets like KDD cup 99, NSL-KDD, CICIDS, and our NIT-IDS dataset. The different deep learning models' results over the standard benchmark dataset, such as KDD 99, NSL-KDD, and CICIDS, are shown in Table 14. The performances of the deep learning models for our NITT-IDS dataset with the standard performance metrics like accuracy, precision, recall, F1 score are given in Table 15. The time consumed for training the models were taken into account to show the effectiveness of the proposed model against the specified GPU configuration. The models were trained on NVIDIA-RTX 2080 Ti GPU processor with 16GB graphics memory, improving the training time with a speedup of up to 20x. The model also optimizes the learning with better Receiver-Operating Characteristics (ROC) curve.



Figurer 7. ROC Curve for RNN+GRU

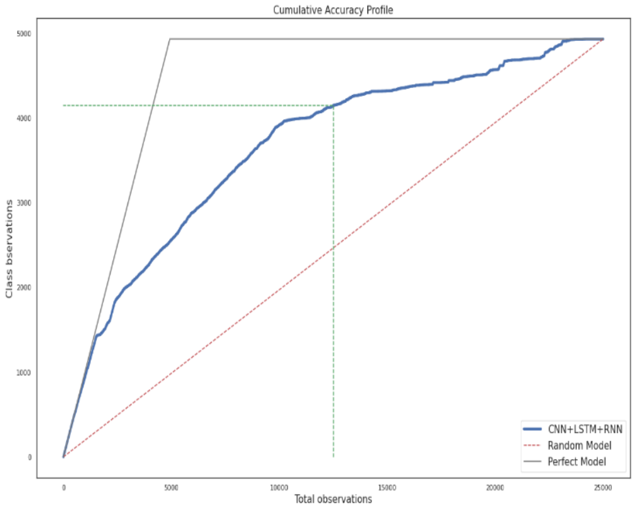


Figure 8. ROC Curve for CNN+LSTM+RNN

Receiver Operating Characteristics (ROC) is a plot between True Positive Rate and False Positive Rate across different thresholds on the y-axis and x-axis. The ROC curve for RNN+GRU and CNN+LSTM+RNN are represented in Figure 7 and Figure 8, respectively. The Area under Curve (AUC) is also used to analyze the effectiveness of the models. AUC is defined as the size of the area under the ROC curve. The value for AUC varies between 0 and 1, in which the AUC value closer to 1 is determined as an outperforming model.

Table 14. Performance measures of different Deep Learning Models for standard benchmark datasets

Models	Accuracy	Precision	Re- call	F1-Score	Training Time
KDD 99 dataset					
RNN+LSTM	92.7	99.3	91.3	95.1	1.2 hrs
CNN+LSTM	91.8	99.1	90.9	94.8	1 hr
CNN+RNN+ LSTM	93.0	99.6	91.5	95.4	1.3 hrs
NSL-KDD dataset					
RNN+ LSTM	88.0	86.7	90.3	88.4	49 min
CNN+ LSTM	86.5	83.8	89.8	86.7	45 min
CNN+RNN+ LSTM	87.8	85.6	90.8	88.1	58 min
CICIDS 2017 dataset					
RNN+LSTM	96.4	90.8	97.4	94.0	49 min
CNN+LSTM	96.1	91.1	96.8	93.8	44 min
CNN+RNN+ LSTM	96.6	91.3	97.2	94.1	52 min

The performance of the deep neural network models against the standard benchmark dataset shows that ensemble models with residual connections show better accuracy and F1 score. The model also indicates comparatively good performance for both emulated packet data captured in a constrained environment and real-time packet data captured in a large institutional network without any constraints. The training time for the network with the benchmark dataset is relatively high due to the generalization error that occurs concerning the hyperparameter tuning. The emulated traffic data with highly balanced normal and intruder packets resulted in high generalization error, leading to increased training time.

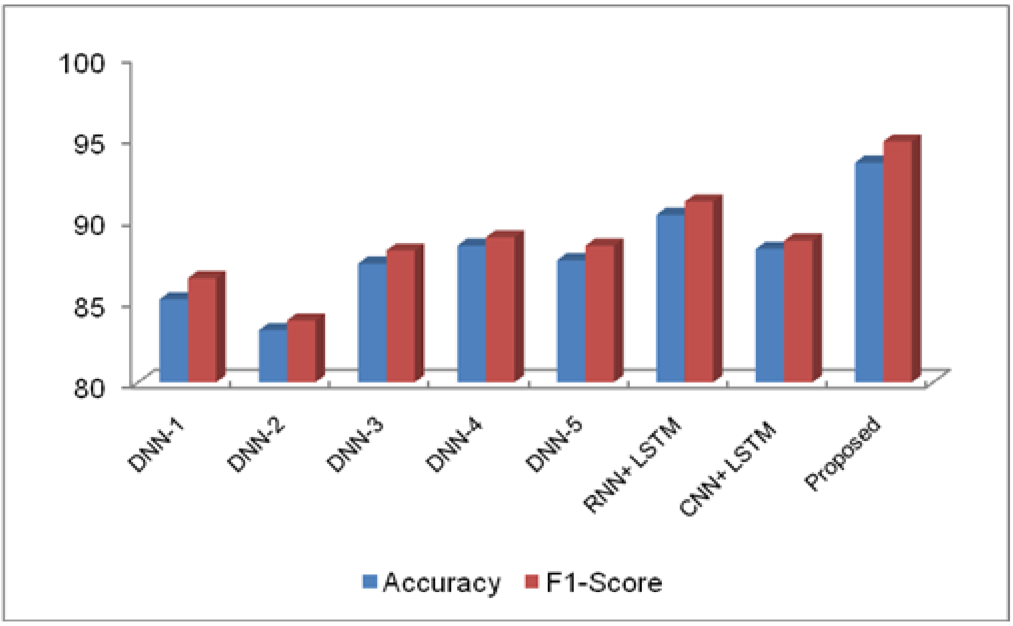


Figure 9. Visualization of the Performance Measures of the Different Deep Learning Models

Table 15. Performance Measures of Different Deep Learning Models for Our Nitt-Ids Dataset

Models	Accuracy	Precision	Re-call	F1-Score	Training Time(min)
DNN-1	85.1	85.8	86.9	86.4	10
DNN-2	83.2	83.1	84.6	83.8	14
DNN-3	87.3	87.6	88.6	88.1	19
DNN-4	88.4	88.4	89.6	88.9	24
DNN-5	87.5	87.5	89.3	88.4	27
RNN+ LSTM	90.3	91.6	90.5	91.1	47
CNN+ LSTM	88.2	88.2	89.3	88.7	36
CNN+ RNN+ LSTM (Proposed)	93.5	95.5	94.1	94.8	42

The model that uses CNN for feature extraction has again proved that convolution operations are also a suitable feature extractor for packet data. From the performance measures, we can infer that the hybrid deep neural network, which combines several base models, outperforms the simple deep neural networks with varying hidden layers. Ensemble models also prove that the networks with residual connection (skipped connection) produce better performance measures than the networks without residual connection. As a general point of representation, it has been justified that ensemble networks reduce the variance of predictions by minimizing the generalization error for the real-time packet data. The comparative illustration of the hybrid deep neural network models' performance measures is shown as the column chart in Figure 9.

V. CONCLUSION

This research work presents an intelligent, real-time intrusion detection system based on hybrid deep neural networks. It visualizes the transition effect of analyzing real-time packet data with a comparative study of machine learning models and deep learning models. The designed ensemble networks are efficient on both the standard benchmark packet dataset and real-time datasets captured in an uncontrolled environment. This work can be explored further to develop a just-in-time Intrusion Detection system in high-speed Ethernet networks after improvising the hybrid model's inference time. The improvisation of this work can be made by inculcating the firewall parameters to enable the firewall to classify the packets efficiently into intruder and normal packets. The system can be built inside the firewall UTM boxes to enhance network security in corporate networks. Evaluation results of the hybrid models validate the optimality of the proposed work in both performance measures and training time.

However, as an extension of this presented work, the details regarding the technical challenges and the analysis with the existing system based on the parameters such as reliability, scalability will be covered. In future, the research work can be directed towards the inclusion of the operational characteristics of the embedded GPUs, such as power, memory requirements for just-in-time classification of intruder packets. This will make this research work to be an end-end product that can be embedded in modern-day firewalls. Since the Gigabit volume of packet data is considered for training and validation of the models, the model's scalability for ever-growing voluminous packet data can be studied. Further investigation can be followed up in finding suitable methods and time intervals for packet data collection, model training, and easy deployment as an Intrusion Detection System to be placed on production servers.

DECLARATION OF COMPETING INTEREST

The authors declare that they have no established conflicting financial interests or personal relationships that may seem to have influenced the research presented in this paper.

ACKNOWLEDGEMENT

We would like to profusely thank the Director, National Institute of Technology, Tiruchirappalli, for providing excellent infrastructure facilities to carry out this research work in the AI-Centre of Excellence and permission under the auspices of the Computer Support Group of NITT.

REFERENCES

[1] Nektaria Kaloudi et al., "The AI-Based Cyber Threat Landscape: A Survey," ACM Computing Surveys, vol. 53, no. 1, pp.1-20, 2020.

[2] Paruchuri et al., "Efficient algorithms to solve bayesian stackelberg games for security applications," Proc. of 23rd AAAI Conference on Artificial Intelligence, pp.1559-1562, 2008.

[3] Hadeel Alazzam et al., "A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer," Expert Systems With Applications, vol. 148, no. 1, pp.1-14, 2020.

[4] Keywhan Chung et al., "Availability attacks on computing systems through alteration of environmental control: smart malware approach," Proc. of the 10th ACM/IEEE International Conference on Cyber-Physical Systems, pp.1-12, 2019.

[5] R. Vinayakumar et al., "Deep Learning Approach for Intelligent Intrusion Detection System," IEEE Access, vol. 7, no. 1, pp.41525-41550, 2019.

[6] Markus Ring et al., "A Survey of Network-based Intrusion Detection Data Sets," arXiv:1903.02460v2, pp.1-17, 2019.

[7] Sharafaldin et al., "A Detailed Analysis of the CICIDS2017 Data Set.," Communications in Computer and Information Science, vol. 977, pp.479-482, 2019.

[8] Mohammed Hamid Abdulraheem et al., "A Detailed Analysis of New Intrusion Detection Dataset," Journal of Theoretical and Applied Information Technology, vol. 97, no. 17, pp.4519, 2019.

[9] Bruno Reis et al., "Selection and Performance Analysis of CICIDS2017 Features Importance," International Symposium on Foundations and Practice of Security, pp.56-71, 2019.

[10] Preeti Mishra et al., "A Detailed Investigation and Analysis of using Machine Learning Techniques for Intrusion Detection," IEEE Communications Surveys & Tutorials, vol. 21, no. 1, pp.686-728, 2018.

[11] Tam Nguyen et al., "The Challenges in SDN/ML Based Network Security: A Survey," arXiv:1804.03539v2, 2018.

[12] Dhilung Kirat et al., "Deep Locker—Concealing targeted attacks with AI Lock smithing," Blact Hat - USA, pp., 2018.

[13] Iman Sharafaldin et al., "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," 4th Intl. Conf. on Information Systems Security and Privacy, pp.108-116, 2018.

[14] Shuo Wang et al., "A Systematic Study of Online Class Imbalance Learning With Concept Drift," IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 10, pp.4802-4821, 2018.

[15] L. Khalvati et al., "Intrusion Detection based on a Novel Hybrid Learning Approach," Journal of AI and Data Mining, vol. 6, no. 1, pp.159-163, 2018.

[16] Jaeyoung Kim et al., "Residual LSTM: Design of a Deep Recurrent Architecture for Distant Speech Recognition," arXiv:1701.03360v3, 2017.

[17] Hyrum S. Anderson et al., "DeepDGA: Adversarially-tuned domain generation and detection," Proc. of the 2016 ACM Workshop on Artificial Intelligence and Security, pp.13-21, 2016.

[18] Matt Fredrikson et al., "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures," Proc. of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp.1322–1333, 2015.

[19] M. Galar et al., "A Review on Ensembles for the Class Imbalance Problem: Bagging, Boosting and Hybrid-Based Approaches," IEEE Transactions on Systems, Man, and Cybernetics, vol. 42, no. 4, pp.463-484, 2012.

[20] Mahbod Tavallaee et al., "A Detailed Analysis of the KDD CUP 99 Data Set," 2009 IEEE symposium on computational intelligence for security and defense applications, pp.1-6, 2009.

[21] Informatik et al., "Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies," Field Guide to Dynamical Recurrent Neural Networks, pp.1-15, 2003.

[22] George Loukas et al., "Cloud-Based Cyber-Physical Intrusion Detection for Vehicles Using Deep Learning," IEEE Access Special Section on Security Analytics And Intelligence For Cyber Physical Systems, vol. 6, pp.3491-3508, 2018.

[23] Razan Abdulhammed et al., "Deep and Machine Learning Approaches for Anomaly-Based Intrusion Detection of Imbalanced Network Traffic," IEEE Sensors Letters, vol. 3, no. 1, pp.1-4, 2019.

[24] Phurivit Sangkatsanee et al., "Practical real-time intrusion detection using machine learning approaches," Computer Communications, vol. 34, no. 1, pp.2227-2235, 2011.

[25] Hao Zhang et al., "A Real-Time and Ubiquitous Network Attack Detection Based on Deep Belief Network and Support Vector Machine," IEEE/CAA Journal of Automatica Sinica, vol. 7, no. 3, pp.790-799, 2020.

[26] Aechan Kim et al., "AI-IDS: Application of Deep Learning to Real-Time Web Intrusion Detection," IEEE Access Special Section on Scalable Deep Learning for Big Data, vol. 8, no. 1, pp.70245-70261, 2020.

[27] Ansam Khraisat et al., "Survey of intrusion detection systems: techniques, datasets and challenges," Springer Cybersecurity, vol. 2, no. 20, pp.1-22, 2019.

[28] Jin Yang et al., "A Simple Recurrent Unit Model Based Intrusion Detection System with DCGAN," IEEE Access, vol. 7, pp.83286-83296, 2019.

[29] Khan et al., "A Novel Two-Stage Deep Learning Model for Efficient Intrusion Detection System," IEEE Access, vol. 7, pp.1-1, 2019.

[30] Delgado et al., "Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?" Journal of Machine Learning Research, vol. 15, pp.1-49, 2014.

[31] Giuseppe Aceto et al., "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges." IEEE Transactions on Network and Service Management pp., 445-458, 2019.

[32] Manuel Lopez-Martin et al., "Network traffic classifier with convolutional and recurrent neural networks for Internet of Things." IEEE Access, pp. 18042-18050, 2017.

[33] Giampaolo Bovenzi et al., "H2ID: Hierarchical Hybrid Intrusion Detection Approach in IoT Scenarios ", IEEE Global Communications Conference (Globecom 2020). Toward Effective Mobile Encrypted Traffic Classification through Deep Learning. Neuro Computing.

[34] Mohammad Lotfollahi et al., "Deep packet: A novel approach for encrypted traffic classification using deep learning." Soft Computing, pp. 1999-2012, 2