Intelligent Intrusion Detection in IoT: Leveraging Fog-Cloud Computing with Deep Recurrent Neural Networks

B Saritha, Bhima Sankar Manthina, M Shailaja, Janjhyam Venkata Naga Ramesh, B Senthilkumaran, Jampani Satish Babu

Abstract—**Many studies have examined deep learning (DL) algorithms for IoT cyber threat detection. Time-sensitive Critical Infrastructures (CIs) using IoT must quickly detect cyber threats near limited equipment to avoid service delays. Deep learning detects intrusions better than shallow machine learning. Communication overheads from huge IoT data and model processing needs hinder deep learning model application to constrained devices. Traditional intrusion detection systems are shallow learning or not trained on IoT datasets and not fog-cloud-ready. We offer a fog and cloud-based IoT intrusion detection framework to address these difficulties. Distributed processing segments the dataset by attack type and identifies time-series IoT properties. Attack detection deep learning recurrent neural network using Simple RNN and Bi-directional LSTM follows. The high-dimensional BoT-IoT dataset supplied massive amounts of realistic IoT attack traffic to evaluate the approach. Under computational constraints, feature selection algorithms reduced dataset size by 90% without affecting attack detection. Smaller dataset models demonstrated higher recall rates than full-featured models without affecting class distinction. Simple RNN and Bi-LSTM models fit well in the restricted feature space. The deep learning-based intrusion detection system performs well in fog-clouds and with massive IoT data.**

Index Terms— **Fog-cloud, Deep learning, IoT, IDS**

Manuscript received March 13, 2025; revised July 20, 2025.

B Saritha is an Associate Professor of Computer Science and Engineering Department, Maturi Venkata Subba Rao (MVSR) Engineering College, Saroornagar, Hyderabad, Telangana, India (e-mail: sarithasampath@gmail.com).

Bhima Sankar Manthina is a PhD Research Scholar of Signal Procession and Communications Research Centre in Electronics and Communication Engineering Department, Indian Institute of Information Technology, Hyderabad, Telangana, India (e-mail: sankar.bhima@gmail.com or bhima.sankar@research.iiit.ac.in).

M Shailaja is an Assistant Professor of Data Science Department, Malla Reddy University, Hyderabad, Telangana, India (e-mail: shailaja.m@mallareddyuniversity.ac.in).

Janjhyam Venkata Naga Ramesh is an Adjunct Professor of Computer Science and Engineering Department, Graphic Era Hill University, Dehradun - 248002, India (e-mail: jvnramesh@gmail.com).

B Senthilkumaran is an Associate Professor of Computer Science and Engineering Department, School of Computing, Vel Tech Rangarajan Dr. Sagunthala R & D Institute of Science and Technology (Deemed to be University) Vel Nagar, Chennai, Tamilnadu, India (e-mail: skumaran.gac16@gmail.com).

Jampani Satish Babu is an Assistant Professor of Computer Science and Engineering Department, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India (e-mail: jampanisatishbabu@gmail.com).

I. Introduction

THE widespread use of the Internet of Things (IoT) is opening the door to a world where everything is linked, which has many positive effects on people's lives. But, the expansion has also caught the eye of malicious actors, who are always looking for new ways to breach security measures. One indicator of this is the dramatic rise in cyberattacks targeting susceptible smart gadgets [1]. Particularly when linked to Critical Infrastructures (CI), these types of assaults might have a negative impact on networks that are enabled by the Internet of Things (IoT). For instance, CI system service level and safety can be drastically impacted by delays in smart grids, eHealth systems, transportation, and manufacturing. The Internet of Things (IoT) is vulnerable to intrusions, yet there is a need for better Intrusion Detection Systems (IDS) that can identify suspicious network connections based on traffic characteristics rather than attack signatures alone. This issue has prompted the development of several intrusion detection systems (IDS) based on Deep Learning (DL) [2], which outperform more conventional ML methods like Support Vector Machines (SVM) [3] in terms of detection accuracy. While DL approaches do offer better performance, they are computationally heavy and are typically implemented in cloud or centralized infrastructures [4]. Because of the necessity to pool data from numerous IoT devices to a central location in order to train DL models, detection delays are also caused by this [5], [6]. Therefore, intrusion detection in CIs that are delay sensitive cannot be achieved using such methods. One solution to this problem is the use of distributed edge-cloud or fog-based architectures, which can detect assaults closer to the edge network and avoid important delays in detecting malicious actions in IoT devices. By efficiently offloading computation workloads from a centralized cloud node, fog nodes can achieve improved scalability with massive deployments of IoT devices and respond rapidly to threats.

For the fog layer, current DL-based intrusion detection systems either necessitate a huge number of fog nodes [7] or redirect edge traffic to central nodes for DL algorithm training [8], [9], both of which can lead to a rise in communication overhead. Fog nodes aren't practical for effective intrusion detection because of the connection cost and the increased compute and memory requirements of the resulting DL models. Several approaches suggested using advanced deep learning algorithms like Long Short-Term Memory

Encoder (LAE) to decrease the dimensionality of the IoT dataset's features [10]. On the other hand, training and deployment times can be lengthened by using such intricate feature selection methods based on back propagation algorithms. We suggest a two-stage procedure to streamline the deployment of DL-based intrusion detection systems for fog nodes, which will enhance the performance of a delay sensitive system. This is accomplished by first converting the multi-class problem to a binary class problem, and then dividing the time-series data from the IoT network according to the attack class. Next, in order to decrease the data size for training the DL models, simple feature reduction approaches including Chi-Square Statistic, Group Method of Data Handling (GMDH), and Mutual Information (MI) are applied. After the datasets have been reduced, they are transferred to a cloud node where the DL algorithm is trained. After that, the edge or fog nodes receive an optimized DL model for detecting assaults on the Internet of Things. By following these procedures, DL workloads can be distributed, and computation time and network latency can be reduced. The BoT-IoT dataset [11] was used to assess this strategy because it includes both regular and IoT attack traffic. The key takeaways from this study are: A decentralized intrusion detection system (IDS) architecture that uses fog technology to identify attacks in delay-sensitive networks.

II. Related work

Attack detection in IoT networks has been the subject of multiple proposals. To train and categorize assaults for a BoT-IoT dataset, we previously used a Feed-forward Neural Network (FNN) in our work. With a high F1 score and 99% accuracy, the FNN model was able to recognize multiple types of attacks. On the other hand, for specific types of IoT threats, the trained FNN model produced lower recall and precision scores. A group of two shallow ML algorithms, C5 and LIBSVM, plus a feature selection stage that relies on information gain were suggested by Khraisat et al. [12] as an ensemble hybrid IDS. According to the experimental results for the BoT-IoT dataset, the performance of the shallow classifiers was low when used alone, but it improved significantly when combined with other classifiers. In comparison to the 99.9 percent accuracy attained by ensemble classifiers, the C5 and LIBSVM classifiers only managed 93% and 92% accuracy, respectively. The suggested method outperformed competing feature reduction approaches while also reducing memory usage. On the other hand, feature selection techniques that rely on deep learning might be computationally intensive. Cloud network training and classification was carried out by Alkadi et al. [13] using a bidirectional LSTM deep learning method. When tested on the UNSW-NB15 and BoT-IoT datasets, the suggested method successfully classified DoS and DDoS attack traffic with a detection rate ranging from 95% to 99%. On the other hand, the model failed miserably at identifying reconnaissance assaults and data theft, which are not considered denial-of-service attacks.

In order to enhance the multi-head attention (MHA) technique, which augments an RNN model in identifying intrusions for Industrial Internet of Things (IIoT) traffic, a deep learning based forensic model named Deep-IFS was suggested in [14]. To depict industrial IoT network traffic on a local and global scale, the suggested architecture employed a gated RNN unit in conjunction with an MHA layer. When compared to centralized DL IDS approaches, their suggested approach significantly improved performance in experiments performed on fog nodes. To improve detection accuracy, the suggested distributed model necessitates a large number of fog nodes; yet, its performance may degrade under heavy traffic conditions.

A hybrid method for intrusion detection in fog-based Internet of Things (IoT) environments was suggested by De Souza et al. [15]. The two-stage detection mechanism they used was initially a combination of Deep Neural Networks (DNN) and K-Nearest Neighbor (kNN) for binary traffic classification in the network. Before processing each input, a DNN model determines if it is invasive or not. After DNN incorrectly classifies an instance, it is passed on to kNN, which assigns the correct class based on the degree of similarity between them using Ecludian distance. The major issue with this approach is that it needs to be tested on IoT datasets to see how effective it is. Previously, it was tested on non-IoT datasets like NSL-KDD and CICIDS2017. Furthermore, picking the right value of k is crucial to the kNN method [16].

A distributed method for detecting anomalies in edge networks using ensemble learning was suggested by Moustafa et al. [8]. In order to identify Internet of Things (IoT) assaults utilizing an IoT-edge-cloud architecture, their suggested method use the Gaussian mixture-based correntropy. The approach's limitations include the need for feature engineering, the use of shallow ML techniques, and the evaluation of the proposed model on datasets that do not pertain to the Internet of Things. Using a fog-cloud architecture, Kumar et al. [17] suggested an ensemble learning method for detecting cyberattacks in the IoMT. For ensemble learning, their method employs three shallow ML algorithms and necessitates a laborious feature engineering phase during algorithm training. According to the current research, there are a lot of obstacles to using deep learning models for Internet of Things intrusion detection. The key obstacles include processing complexity, time delay, bandwidth needs, and efficiently distributing the detection workload to numerous worker nodes. As a result, we begin by dividing the multi-class BoT-IoT dataset into several binary class datasets according to the packet arrival time. Additionally, in order to decrease the size of the dataset, feature selection is executed on separate binary class datasets. Finally, we use deep learning models to determine if the instance is normal or an assault.

III. Presented architecture

Full names of authors are preferred in the author field, but are not required. Put a space between authors' initials. Here, we introduce our fog-cloud based framework for identifying malicious activity in Internet of Things (IoT) communications. In order to build models that can identify IoT threats, the framework goes through a number of stages, as shown in Figure 1. These stages involve collecting raw network packets, pre-processing them, selecting features, training, validating, and testing them using deep learning methods. There are three levels to the framework: The Internet of Things devices, the fog layer, and the cloud layer. On the most fundamental level, IoT devices communicate and function. At the fog layer, tools like TCP dump can be used to capture raw network packets from these IoT devices. Furthermore, the packets will be transformed into time-series data during a pre-processing stage. The majority of attacks have been recorded at certain timestamps, while normal data occurs throughout all time periods, according to an analysis of the timestamps identified in the BoT-IoT dataset utilized in this study. Because of this, we suggest dividing the dataset into subcategories based on the locations of attacks. On this basis, we also employ a dataset partitioning method based on time stamps for training, testing, and validation. In addition to facilitating remote dataset processing, this strategy employs an additional feature selection methodology to significantly shrink datasets in preparation for training on cloud instances. Prior to uploading to the cloud computing layer, feature selection is executed at the fog layer to decrease the dataset size. Deep learning models can be taught and assessed by utilizing the computation resources available at the cloud layer. At last, the fog layer is used to install optimal DL models for assault detection.

Group Method of Data Handling (GMDH) [18], Mutual Information (MI) [19], and Chi-Square statistic approaches [20] were selected as dimensionality reduction algorithms to be compared. In this study, we compared the attack classification performance of two RNN types: Simple RNN and Bi-directional LSTM. The sections that follow provide a detailed explanation of the feature selection techniques and DL methods that were utilized in this study. Selecting the most relevant features and deleting those that do not help to class separation is a crucial step in establishing an effective framework for network intrusion detection. To significantly decrease the dataset size and processing needs of a deep learning model, the feature selection phase can be implemented prior to training the model. The following sections provide a more detailed explanation of the three feature selection strategies that were examined in this work.

Real-time neural networks (RNNs) make it possible for data to remain persistent by utilizing feedback loops that connect the current state to its predecessors. Because of the temporal correlations between network events [21], RNNs may use the past state of the network's traffic to predict its current state, making them ideal for detecting intrusions. Each cell in an RNN takes in data and stores a single hidden state that is passed down from one iteration to the next. Activation of the hidden layers transfers certain data from one-time step to another. In order to process the input data entirely, RNNs primarily include two primary phases: computing the parameters of a single time step and looping across time steps. In a single RNN cell design, the input data is taken from the current state and stored in the previous hidden state. As shown in Figure 2, there is one RNN cell design.

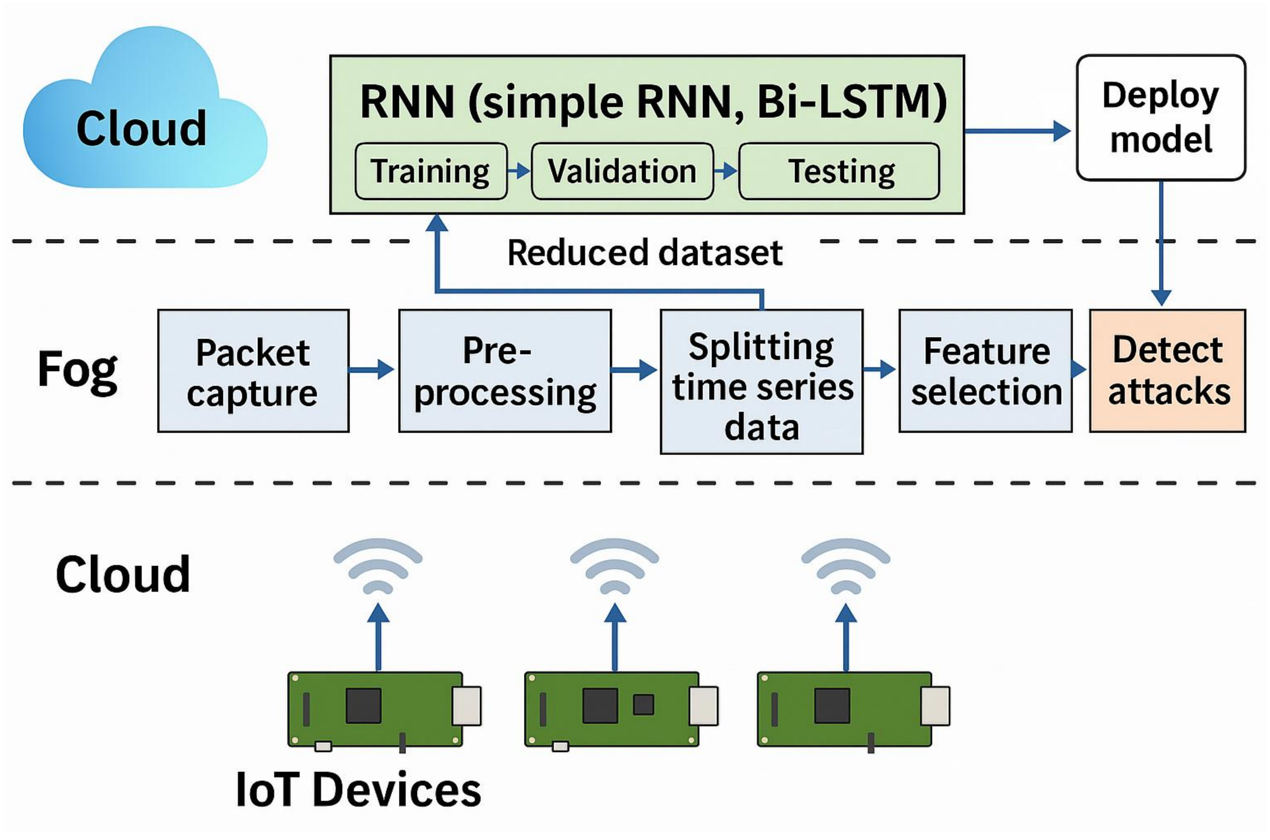

Fig. 1. Framework Workflow

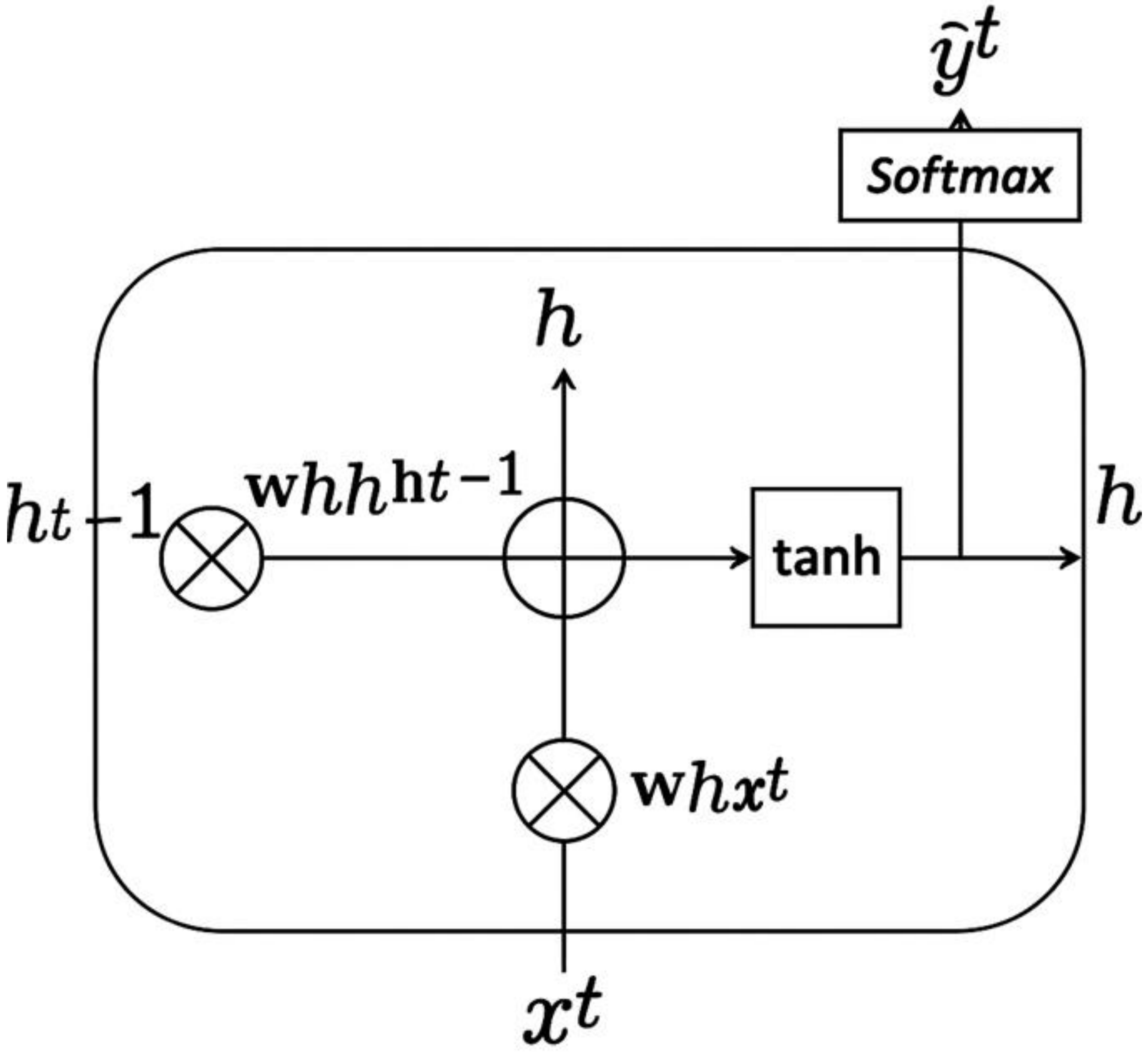

Fig. 2. A single RNN cell using a single input and the concealed state.

An upgraded form of RNN, LSTM accounts for long-term dependencies when predicting the output class. Therefore, LSTM can make better selections since they can remember information for a longer length. Since there is a lengthy lag between the input time and the decision time, RNNs are unable to learn long-term dependencies adequately due to the vanishing gradient problem [22]. This limitation is circumvented by LSTM through the use of gates, which allow for the transfer of data to any cell as needed [23] and the retention of contextual information for extended durations. LSTMs are taken to the next level when they are fed information in both the forward and backward directions using a bi-directional hidden layer [24], [25].

IV. Results and analysis

In order to build the Simple RNN and bidirectional LSTM modules, the Tensor Flow and Keras libraries were utilized. With 64% going toward training, 16% toward validation, and 20% toward testing, the dataset was partitioned in a 64-16-20 fashion. The 80/20 ratio, which the Pareto principle follows, provided the basis for this splitting method, which divides the dataset into training and testing sets. Separately, the training set is split into 64% for training and 16% for validation. In addition to helping with hyper parameter adjustment, the validation set gives an objective assessment of the trained models. This study uses the top ranking features to build a model using the Keras version of RNN (Simple RNN). Since the attack category influences the input shape, it determined the number of parameters for the RNN model. The service attack category, for instance, has 91 input features, 128 trainable parameters in the first layer, a three-second window, and so on.

The Bi-directional LSTM model, like the Simple RNN, was constructed using the Keras deep learning package. The dense layer in both Simple RNN and Bi-LSTM used a softmax activation function, while the hidden layer in both models used a tanh activation function. With Adam Optimiser, we settled on accuracy as our measure and sparse categorical cross-entropy as our loss value. Figure 3 shows the different levels of our DL architecture that is built on RNNs.

A windowed dataset was initially applied to the time-series BoT-IoT dataset; this type of dataset systematically moves through each window, picking a specified amount of time samples at a time, until the complete time-series is covered. You can specify the number of time samples to use in a window by using the argument. The next component of the design is an input layer that receives the features as input. The computation is done by a hidden layer of many neurons, and the instances are classified as normal or assault in the output layer. Throughout the training phase, the weights that connect the three layers of the deep learning network were determined and fine-tuned until they found an optimal solution. The back propagation algorithm is used for this purpose; it iteratively updates and selects the interconnection weights that result in the lowest loss.

The results show that adding more hidden layers to Simple RNN improves the model's performance. However, after adding three hidden layers, the performance did not continue to improve, therefore that's why we stuck with three in this work. Because it produced the best results across all classes, 128 neurons were the optimal number to use. No additional tuning was done because the dropout rate had no impact on the model's performance. Adam optimizer’s learning rate had a significant effect on performance; the optimal value was 0.0001, thus that's what the model was trained with. The model's performance was unaffected by increasing the epoch count since the majority of iterations terminated before 20 epochs. To further avoid over-fitting, we used a 5-iteration patience condition to promptly terminate training if the validation loss is unchanged or increases with each subsequent iteration.

Batch size affects learning speed and stability when working with big datasets. The results demonstrated that a batch size of 128 enhanced the model's accuracy for Simple RNN. A windowed dataset is created by adjusting the window size; among the values between 2 and 5, a window size of 3 demonstrated the highest performance. The same set of hyper parameter values were picked for Bi-LSTM since they improved the model's performance. But instead of 20, the batch size for Simple RNN was raised to 50.

Here we offer the evaluation results, which show how Simple RNN and bi-directional LSTM models fared, as well as the features that the feature selection algorithms used. In order to assess the efficacy of the feature selection process, we display the features chosen by each feature selection algorithm and the percentage of data size reduction achieved by using the best features. By calculating the confusion matrices for each sub-dataset, we can see how well the model classified normal and attack classes, which is useful for assessing the network traffic classification. Aside from accuracy and recall, precision and F1 score are also utilized to evaluate performance. Specifically, in order to assess the effectiveness of the detection, the following metrics were computed.

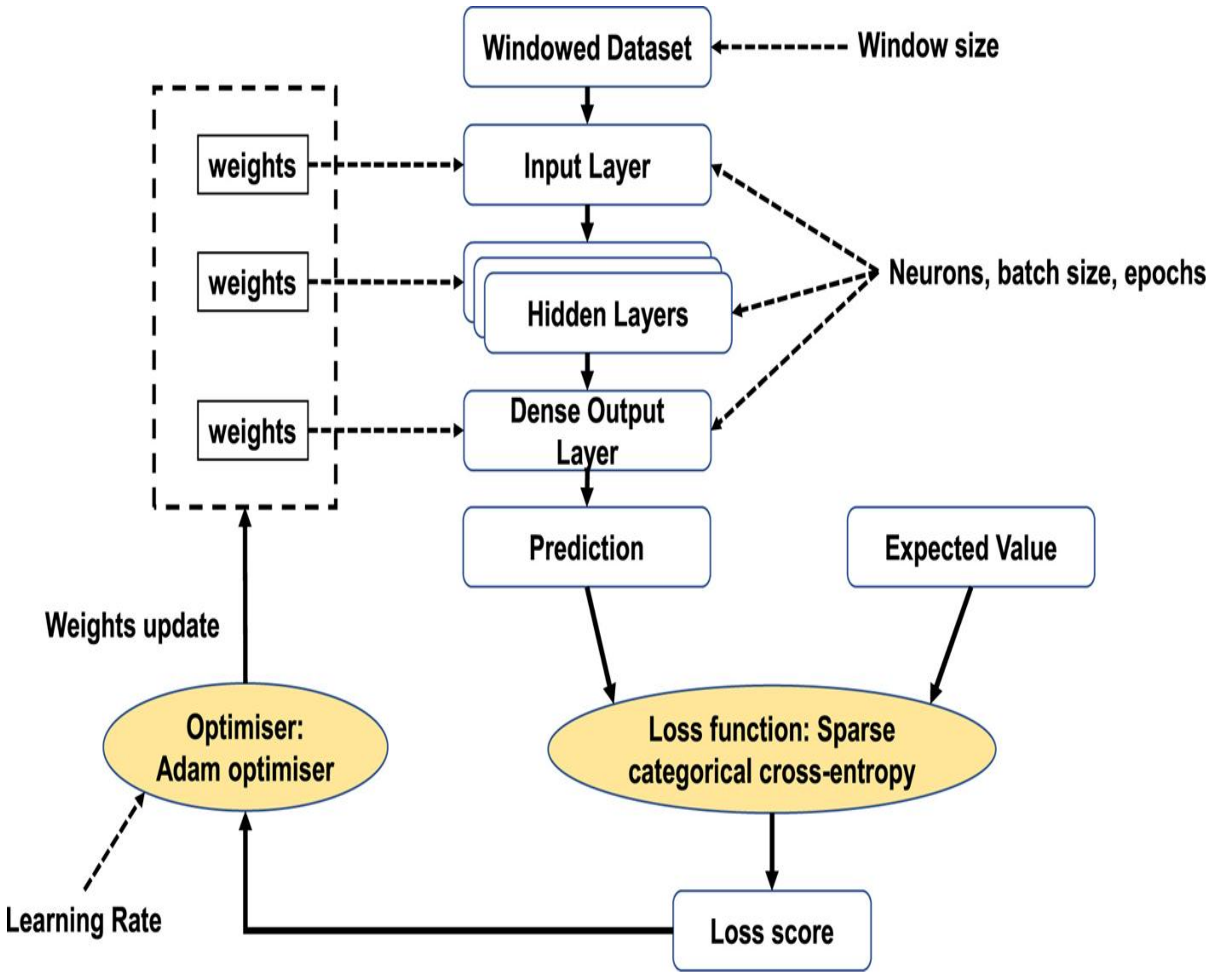

Fig. 3. Architecture of the RNN model

TABLE 6
THE PERCENTAGE OF MEMORY DECREASE FOR EACH SUBCATEGORY USING THREE FEATURE SELECTION TECHNIQUES

Category	GMDH-lr-cov	GMDH-lr	MI	Chi-Sqr
Service	89.24	89.24	87.09	87.09
OS	89.28	86.9	85.71	85.71
Data Exfiltra-tion	90.27	90.27	83.33	83.33
Keylogging	90.9	93.5	84.41	84.41
DDoS-HTTP	90.66	93.33	83.99	83.99
DDoS-TCP	87.49	86.11	83.33	83.33
DDoS-UDP	82.85	84.28	82.85	82.85
DoS-HTTP	91.99	93.33	83.99	83.99
DoS-TCP	87.49	87.49	83.33	83.33
DoS-UDP	84.28	89.99	82.85	82.85
Average	**88.45**	**89.44**	**84.09**	**84.09**

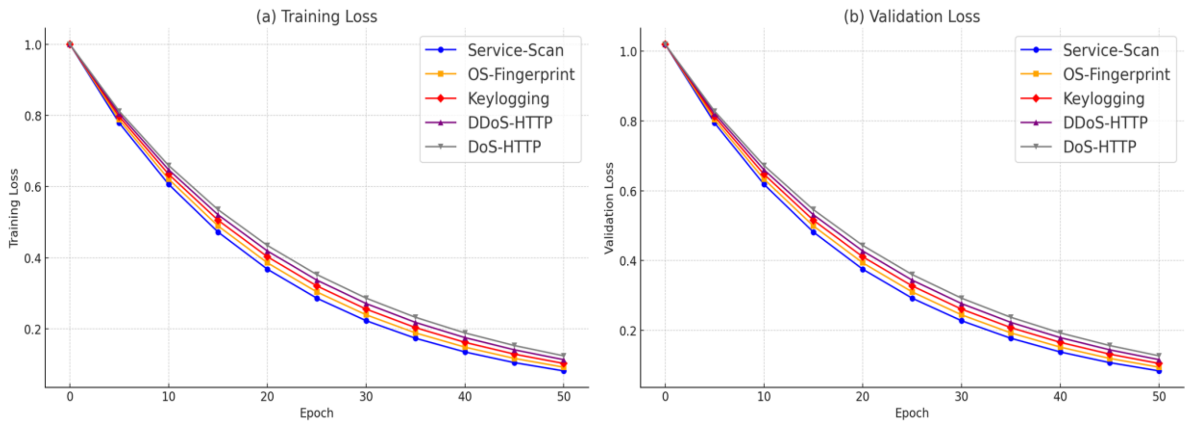

Fig. 4. Bi-LSTM training loss (a) and validation loss (b) for all attack dataset types

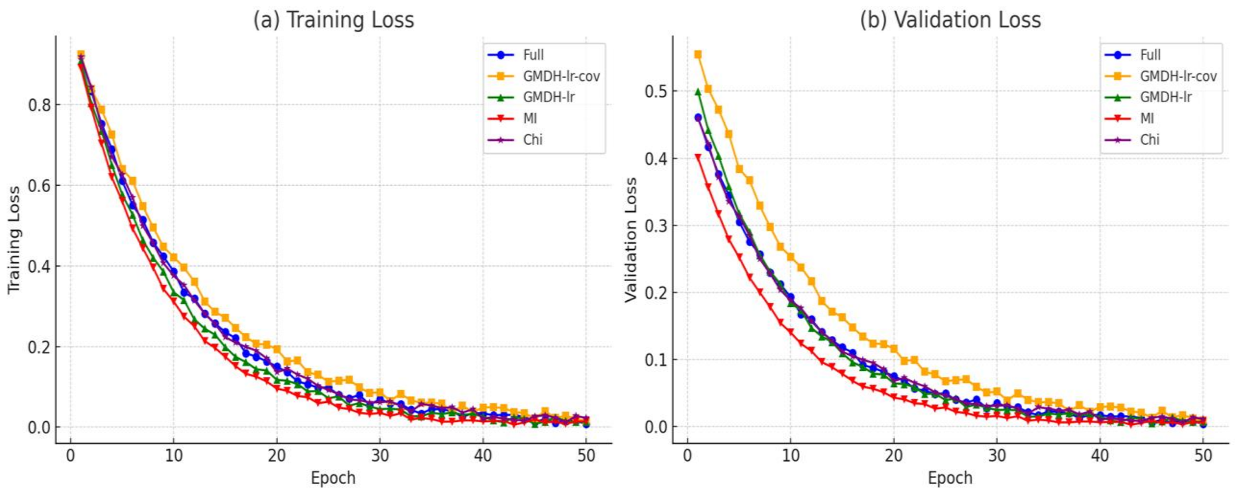

Fig. 5 The training loss (a) and validation loss (b) achieved using Bi-LSTM for the detection of service-scan attacks

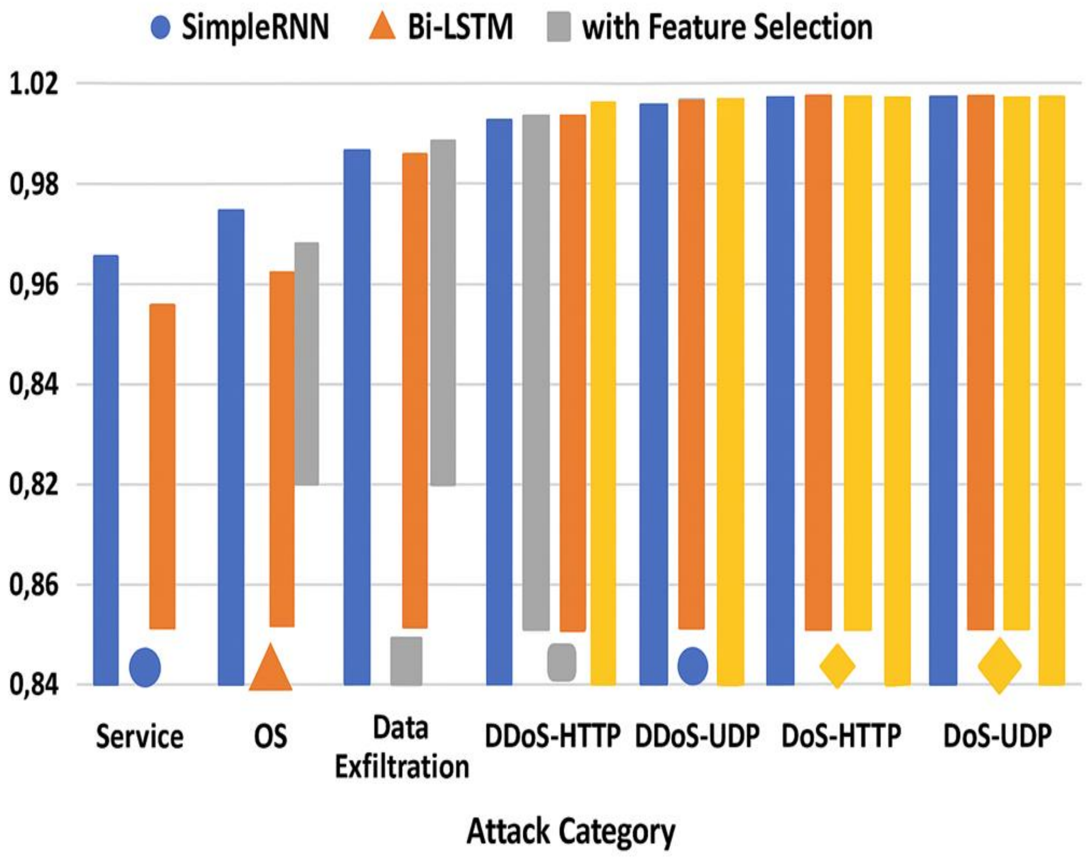

Fig. 6. SimpleRNN and Bi-LSTM attack detection recall rate on BoT-IoT dataset compared to prior work

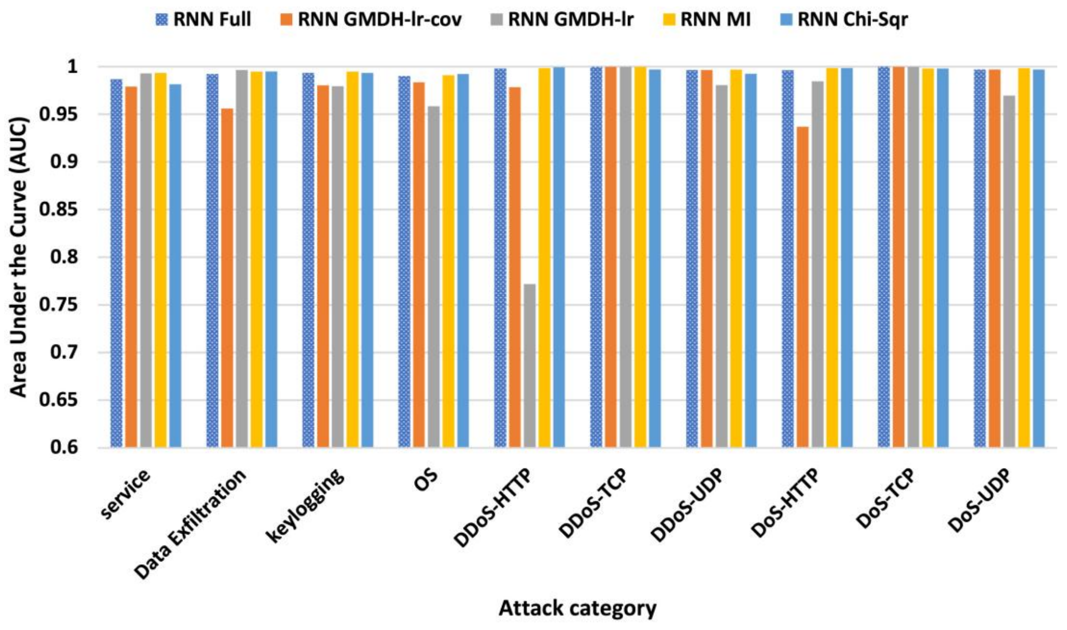

Fig. 7. Area Under the Curve (AUC) Comparison of Suggested Simple RNN for Different Types of Attacks.

As seen in Table 6, a significant reduction in data size was achieved by feature selection utilizing the three techniques. In terms of Mega Bytes (MB), the subcategory for service scans saw the largest drop in data size. Within this subcategory, the reduced data for GMDH, MI, and Chi-Sqr consumed 107, 128 and 997 MB of storage space, respectively, as opposed to the full dataset's 997 MB requirement. The GMDH-lr and GMDH-lr-cov approaches reduced data size for data theft, keylogging, DDoS-HTTP, and DoS-HTTP by more than 90% in percentage terms. The data size was reduced by 80%-90% when 10 characteristics were chosen for MI and Chi-Sqr. The results show that the selected feature selection approaches successfully remove unrelated and irrelevant characteristics, therefore reducing the data needed to train and evaluate the deep learning models.

In Figure 4, we can see that the Bi-LSTM training and validation procedure took more than 20 epochs, but for the majority of sub-datasets, it ended before 50 epochs. All assault sub-categories saw a training loss decrease below 0.2 after the third epoch. The validation loss also dropped below 0.1 across the board, with the exception of DoS_HTTP and DoS UDP, where it varied between 0 and 0.05. The Bi-LSTM model outperformed the Simple RNN models in general because it takes long-term time dependencies into account when making judgments.

Models trained on a smaller feature space were also more resistant to the problems of overfitting and under fitting, which plague intrusion detection systems. The training and validation losses with full and restricted feature space from one attack category are compared in Figure 5. The models with reduced feature space did not overfit or under fit, as shown by the training and validation losses being within 0 and 0.05 after the first 15 epochs.

We conducted a comparison of our proposed approach against the results obtained from the deep blockchain framework (DBF) introduced, which utilized Bi-LSTM for the classification of attack traffic within the BoT-IoT dataset. Figure 6 illustrates the recall rate across different attack sub-categories, juxtaposed with the findings reported. Our proposed approach demonstrated superior performance compared to their method, achieving higher recall rates in the detection of attack traffic. Particularly in areas like service scanning, operating system fingerprinting, data exfiltration, and keylogging.

Findings from both the feature selection process and the deep learning based categorization demonstrate that the former enhances the latter's ability to detect assaults based on the Internet of Things (IoT). Feature selection is a powerful tool in deep learning because it allows us to drastically shrink the dataset without sacrificing any of the crucial information that distinguishes between the input and output classes. When compared to other feature selection methods, the ones used by the MI algorithm improved performance across multiple types of attacks. Figures 7 indicate that this is the case for all of the subcategories when looking at the area under the curve metric. The accuracy of the model's class predictions is directly proportional to its area under the curve (AUC). For simplicity's sake, we just show the final AUC score in the plot, since there are various types of attacks.

V. Conclusion

The use of deep learning (DL) methods in intrusion detection systems has proven to be effective in identifying patterns of attacks. Deploying DL-based IDS closer to the IoT devices is necessary to shorten the time delay in detecting such intrusions and to combat complex assaults that target the IoT paradigm. Deploying DL-based IDS near the edge IoT devices may be hindered by issues like massive amounts of IoT data and the complicated computing demands of DL approaches. Therefore, we suggested a fog-cloud architecture-effective DL-based IDS framework for the Internet of Things in this study. Datasets are partitioned in the suggested architecture based on the arrival time of attack flow. In addition, the high-dimensional BoT-IoT dataset has features that aren't relevant removed during a feature selection stage. Two RNN algorithms, Simple RNN and Bi-LSTM, are trained and evaluated on the reduced dataset to determine which occurrences constitute attack traffic and which are considered regular traffic. The outcomes demonstrate the effectiveness and resilience of our suggested framework in the face of over- and under-fitting issues. By reducing the dataset size by 90% during the feature selection process, the network latency can be efficiently lowered while sending huge amounts of IoT data to the cloud network for deep learning tasks. When compared to the complete feature set and the DBF reported, the Simple RNN and Bi-LSTM models trained on the smaller set of features likewise exhibited an improved recall rate. Training DL models used less memory and computed less as a result of the feature selection process. To identify cyberattacks on IoT devices in CI, the suggested deep learning based fog-cloud IDS approach combines a dataset splitting method with feature selection to reduce dataset size, low computation requirements of trained models, and superior detection capability.

References

[1] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, J. Lopez, a survey of iot-enabled cyberattacks: Assessing attack paths to critical infrastructures and services, IEEE Commun. Surv. Tutor. 20 (4) (2018) 3453–3495.

[2] D. Kwon, H. Kim, J. Kim, S.C. Suh, I. Kim, K.J. Kim, A survey of deep learning-based network anomaly detection, Cluster Comput. (2019) 1–13.

[3] M. Ge, X. Fu, N. Syed, Z. Baig, G. Teo, A. Robles-Kelly, Deep learning-based intrusion detection for IoT networks, in: 2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing, PRDC, 2019, pp. 256–25609, http://dx.doi.org/10.1109/PRDC47002.2019.00056.

[4] L. Lyu, J.C. Bezdek, X. He, J. Jin, Fog-embedded deep learning for the internet of things, IEEE Trans. Ind. Inform. 15 (7) (2019) 4206–4215.

[5] 5. Baddu Naik Bhukya, Vutukuri Sarvani Duti Rekha, Venkata Krishnakanth Paruchuri, Ashok Kumar Kavuru and Kadiyala Sudhakar “Internet of Things for Effort Estimation and Controlling the State of an Electric Vehicle in a Cyber Attack Environment” Journal of Theoretical and Applied Information Technology, ISSN: 1817-3195, Vol. 101, No.10, pp. 4033 – 4040, May-2023.

[6] A. Diro, N. Chilamkurti, Leveraging LSTM networks for attack detection in fog-to-things communications, IEEE Commun. Mag. 56 (9) (2018) 124–130, http://dx.doi.org/10.1109/MCOM.2018.1701270.

[7] Baddu Naik Bhukya, V. Venkataiah, S. Mani.Kuchibhatla, S. Koteswari, R V S Lakshmi Kumari, and Yallapragada Ravi Raju, "Integrating the Internet of Things to Protect Electric Vehicle Control Systems from Cyber Attacks," IAENG International Journal of Applied Mathematics, vol. 54, no. 3, pp433-440, 2024

[8] N. Moustafa, M. Keshk, K.-K.R. Choo, T. Lynar, S. Camtepe, M. Whitty, DAD: A distributed anomaly detection system using ensemble one-class statistical learning in edge networks, Future Gener. Comput. Syst. 118 (2021) 240–251.

[9] A. Samy, H. Yu, H. Zhang, Fog-based attack detection framework for internet of things using deep learning, IEEE Access 8 (2020) 74571–74585.

[10] S.I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, H. Gacanin, Hybrid deep learning for botnet attack detection in the internet of things networks, IEEE Internet Things J. (2020).

[11] N. Koroniotis, N. Moustafa, E. Sitnikova, B. Turnbull, Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT Dataset, Future Gener. Comput. Syst. 100 (2019) 779–796.

[12] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, A. Alazab, A novel ensemble of hybrid intrusion detection system for detecting internet of things attacks, Electronics 8 (11) (2019) 1210.

[13] O. Alkadi, N. Moustafa, B. Turnbull, K.-K.R. Choo, A deep blockchain frameworkenabled collaborative intrusion detection for protecting IoT and cloud networks, IEEE Internet Things J. (2020).
[14] M. Abdel-Basset, V. Chang, H. Hawash, R.K. Chakrabortty, M. Ryan, Deep-IFS: Intrusion detection approach for IIoT traffic in fog environment, IEEE Trans. Ind. Inform. (2020).
[15] C.A. de Souza, C.B. Westphall, R.B. Machado, J.B.M. Sobral, G. dos Santos Vieira, Hybrid approach to intrusion detection in fog-based IoT environments, Comput. Netw. 180 (2020) 107417.
[16] J. Huang, Y. Wei, J. Yi, M. Liu, an improved knn based on class contribution and feature weighting, in: 2018 10th International Conference on Measuring Technology and Mechatronics Automation, ICMTMA, IEEE, 2018, pp. 313–316.
[17] P. Kumar, G.P. Gupta, R. Tripathi, an ensemble learning and fog-cloud architecture-driven cyber-attack detection framework for IoMT networks, Comput. Commun. 166 (2021) 110–124, http://dx.doi.org/10.1016/j. comcom.2020.12.003, URL https://www.sciencedirect.com/science/article/pii/S0140366420320090
[18] Baddu Naik B, Manam Ravindra, Simhadri Mallikarjuna Rao, Srikanth Kilaru, Madamanchi Brahmaiah, Bezawada Manasa, Muralidhar V “Cyberattack Prevention and Detection in Smart Power Systems Using Deep Learning” Journal of Theoretical and Applied Information Technology, May 2025. Vol.103. No.9, pp. 3934-3944.
[19] F. Amiri, M.R. Yousefi, C. Lucas, A. Shakery, N. Yazdani, Mutual informationbased feature selection for intrusion detection systems, J. Netw. Comput. Appl.34 (4) (2011) 1184–1199.
[20] Y. Yang, J.O. Pedersen, A comparative study on feature selection in text categorization, in: Proceedings of the Fourteenth International Conference on Machine Learning, ICML ’97, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997, pp. 412–420.
[21] A. Sperotto, A. Pras, Flow-based intrusion detection, in: 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops, IEEE, 2011, pp. 958–963.
[22] Y. Bengio, N. Boulanger-Lewandowski, R. Pascanu, Advances in optimizing recurrent networks, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 8624–8628.
[23] Yuxiang Ji, Yu Zhang, Ling Chen, and Jianping Zhou, "Event-triggered Stabilization for Neural Networks Subject to Replay Attacks," Engineering Letters, vol. 32, no. 10, pp1882-1887, 2024.
[24] A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional LSTM and other neural network architectures, Neural Netw. 18 (5–6) (2005) 602–610.
[25] J. Xiao, C. He, Adaptive selection of classifier ensemble based on GMDH, in: 2008 International Seminar on Future Information Technology and Management Engineering, 2008, pp. 61–64, http://dx.doi.org/10.1109/FITME.2008.132.