Babylonian Journal of Internet of Things
Vol.2024, **pp**. 94–105
DOI: https://doi.org/10.58496/BJIoT/2024/012; ISSN: 3006-1083
https://mesopotamian.press/journals/index.php/BJIoT

Research Article

Evaluating the Effectiveness of Machine Learning-Based Intrusion Detection in Multi-Cloud Environments

Hussein Jawad Kadhim AL Masoodi[1, *,]

[1] *Faculty of Computer Engineering, Department of Software Engineering, Islamic Azad university, Isfahan Branch (Khorasgan), Iran.*

ARTICLE INFO

Article History
Received 22 Jun 2024
Revised 18 Jul 2024
Accepted 13 Aug 2024
Published 15 Sep 2024

Keywords
Intrusion Detection System
Machine Learning
Convolutional Neural Network
Multi-Cloud Environment
Network Security
UNSW-NB15 Dataset
Cybersecurity
Deep Learning

ABSTRACT

In the dynamic landscape of cloud computing, multi-cloud environments have emerged as the prevalent ones due to offering a host of advantages including redundancy, flexibility, and increased security. These multi-cloud environments can, however, be vulnerable to stealthy cyber-attacks, thereby making it a challenging job to find efficient intrusion detection systems (IDS). The efficiency of the Machine Learning-based Intrusion Detection System in multi-cloud environments has been evaluated using a 1-D Convolutional Neural Network to classify network traffic based on the UNSW-NB15 dataset. The dataset included a wide variety of network features, and it was decided that it could be used directly after converting the categorical variables into their numerical forms and scaling them with MinMaxScaler. This was used for training and testing the 1D-CNN model, which had attained an accuracy of 84.02% during training and 82.79% on validation. Further, the performance of the model is elicited with metrics that include precision at 80.91%, recall at 82.79%, and an F1-score at 81.28%, showing its capability in identifying network intrusions effectively. The small difference between the accuracy of training and validation shows that overfitting was minimal; thus, the model will generalize well when applied to unseen data. This research underlines the possibility of using deep learning techniques, and particularly CNNs, for intrusion detection in complex multi-cloud infrastructures. Probably future work would be optimizing the model through hyperparameter tuning, using additional sources of data, and even exploring much more sophisticated architectures that will further improve the accuracy of intrusion detection.

1. INTRODUCTION

The rapid acceptance of cloud computing had brought a sea change in the fundamental dynamics of how organizations implement and manage their IT infrastructure. As businesses increasingly turn to multi-cloud environments-leveraging the offerings of multiple cloud providers simultaneously-they realize advantages in redundancy, flexibility, and the ability to avoid vendor lock-in. Of course, these advantages carry consequences in terms of increased security risks.

Because of their very nature, multi-cloud architectures exacerbate complexities with network management, data synchronization, and security enforcement in ways that can provide vulnerabilities for even the most sophisticated types of cyber-attacks. These same security mechanisms, many designed for single-cloud or on-premises environments, struggle to handle the dynamic and distributed nature of multi-cloud ecosystems. Therefore, sophisticated Intrusion Detection Systems should be developed for efficient detection and mitigation of any form of possible threat across varied and connected cloud platforms [1, 2].

It is the portions of intrusion detection systems that monitor, detect, and respond to unauthorized or malicious activities within a network. It is in this regard that, for the last couple of years or so, much attention has been given to IDS based on machine learning-based techniques owing to their ability to automatically learn both normal and abnormal behavior patterns from big datasets. Among various machine learning approaches, CNNs, a family of deep learning algorithms, have promised a great deal in network security by automatically extracting and learning features hierarchically from the input data in a very effective way to detect complex patterns that indicate network intrusion.

The performance of the 1D-CNN-based IDS in a multi-cloud environment is investigated in this research, using the UNSW-NB15 dataset, one of the comprehensive network traffic datasets for cybersecurity-related research. It consists of a rich set

*Corresponding author. Email: husseinmassudi93@gmail.com

of features that represent different aspects of network traffic, such as protocol types, packet rates, and byte counts, among other things, necessary to classify normal and malicious activities.
Preprocessing was done by converting categorical variables into numerical representations and then normalizing the data to be used in training for classifying network traffic into the different categories of various attacks and normal traffic. The majority of the work has concentrated on the performance of this 1D-CNN model in detecting intrusions correctly, with the metrics relevant in this case being accuracy, precision, recall, and F1-score.
Results have shown the highly generalized performance of the proposed 1D-CNN-based IDS with high accuracy for both the training and the validation dataset, thus showing its applicability in real-world multi-cloud environments. It also identified areas for further improvement: bringing the slight overfitting observed and exploring higher complexities in model architecture.
The contribution of this research represents the quest for IDS effectiveness within a multi-cloud environment as part of ongoing efforts toward making cloud infrastructures secure from the ever-evolving landscape of cyber threats. The results of the current study also emphasize the need for continued innovation in the security technologies to keep pace with the ever-growing complexity of the cloud computing environments and the sophisticated nature of modern cyber-attacks.

2. RELATED WORK

Lav Gupta and Tara Salman [3] introduce a new system called MUSE that responds to security challenges arising in healthcare networks, especially with increased network complexity due to the integration of IoT devices, edge computing, and core clouds. The system MUSE makes use of deep hierarchical stacked neural networks to detect malicious activities that alter data moving within different parts of the healthcare network.
MUSE innovation lies in the fact that it can aggregate small, already trained models in the edge clouds into a large pre-trained model in the core cloud. It drastically reduces the training time of the model in the core cloud to only 6-8 epochs, while each individual edge has to train 35-40 epochs. Comprehensive verification based on the test data showed that MUSE speeds up training by 26.2% while retaining high accuracy, in the range of 95% to 100%, when detecting unknown attacks. This would underline the potential of this system in enhancing the security and efficiency of next-generation health care networks.
Some newer ways have been developed for cloud security enhancement, combining the works on Privacy Preserving techniques with IDS by C. Jansi Sophia Mary and K. Mahalakshmi [4] to protect vital business data within the cloud. Their research has introduced the SHO-DESNID method, combining sea horse optimization with a deep echo state network for improved intrusion detection in cloud environments.
It employs min-max normalization in pre-processing the data and then classifies an intrusion into various classes using the DESN model. The SHO algorithm helps in hyperparameter optimization, strengthening the DESN model. In a simulation using benchmark IDS databases, the proposed SHO-DESNID technique outperforms the existing methods hence fortifying cloud security with its accuracy in detecting and classifying network abnormalities.
Qigang Liu and Deming Wang, [5] in their work, point out that intrusion detection systems or IDS introduce complexities due to the intrusion nature in various ways and class distribution, which is highly imbalanced. For an effective IDS, classic methods such as non-robustness dependant on predefined features, and automatic feature learning suffers from overfitting. Also, unsupervised multi-class classification can't work effectively.
These problems are addressed by the authors, who propose a new approach that leverages distinctive features from three perspectives: anomaly identification, clustering, and classification. Their method involves three different models: an autoencoder-based contrastive learning model for feature extraction, a supervised-learning-based clustering model, and a classifier with an MLP model.
The models are wrapped in one framework with a custom-designed loss function to handle class imbalances. Intensive experiments on classic intrusion detection datasets show that their approach significantly outperforms the existing methods in binary and multi-class classification tasks, which further confirms the efficiency of boosting the performance of IDS.

3. PROPOSED METHODOLOGY

The proposed methodology is focused on how to leverage the Convolutional Neural Network in order to develop an Intrusion Detection System for a multi-cloud environment. Since network traffic in multi-clouds is very complex, it is challenging for traditional detection methods to identify sophisticated, constantly evolving threats.
Addressing the challenges, this approach will integrate into the field of advanced machine learning, which includes feature extraction, data preprocessing, and optimization of models, to give an enhanced outcome regarding better detection accuracy. We begin with the acquisition and preparation of the UNSW-NB15 dataset, which would be the basis on which we train the IDS. This is one of the most complete representatives of current network traffic, which includes both categorical and continuous features of network communications [6,7].

Extensive preprocessing was done to make this data more suitable for feeding into the model and learning from it: the categorical variables were transformed by means of Label Encoding; feature values were normalized with MinMaxScaler. The preprocessing on the input data, by performing normalization, is an important step in that the data becomes consistent; hence, the learning process for 1D-CNN will be easier.
Also, the architecture of 1D-CNN in itself is a design for the extraction of complex patterns within the network traffic. In addition, the layers are set such that they progressively refine features that are of importance. In the process of training and evaluating the model, the dataset was divided into both a training set and a validation set to make sure the model performance undergoes stringent testing.
All these steps are followed while continuously monitoring key performance metrics, including accuracy, precision, recall, and F1-score, so that the model fits the training data well and generalizes effectively to the unseen ones. Hence, the proposed methodology uses the robustness of deep learning on preprocessed data to develop an IDS for protection against a wide array of cyber threats in multi-cloud environments.

3.1 Dataset Description

This dataset is utilized in this paper for the development and performance evaluation of the proposed Intrusion Detection System, whereby the dataset itself has become a standard benchmark for research in cybersecurity. It is specially designed to simulate modern network traffic in such a way that it is able to capture a wide range of both normal and malicious activities. It consists of 100,000 records of network connections, each with 49 features [8,9].
Thus, the 49 features are both categorical and continuous variables that comprehensively represent various aspects of network traffic: protocol type, service type, packet statistics, and flow features. The dataset was generated by the IXIA PerfectStorm tool in a controlled environment at the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS). This dataset is of value because it has been diverse enough to contain several attack categories: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms, together with normal traffic; hence, it is a very ideal candidate for training and testing machine learning models that are to be applied for intrusion detection. Fig 1. Shows Features Correlation Matrix.
The UNSW-NB15 dataset contains an extensive coverage of contemporary cyber-attacks in its structure, which constitutes various attack types based on real-world scenarios. Each of the attack categories represents a particular methodology for network security compromise; thus, they are broadly variable for testing and evaluation purposes of intrusion detection systems. The main attack types in the dataset are:[10][11]

- Fuzzers: These are types of attacks where random or invalid data is sent to a system in search of possible weaknesses. The goal of fuzzers is to crash systems or find security loopholes by deluging them with unexpected input.
- Analysis: This class also includes passive attacks, wherein the attacker does not interfere directly with the system himself but rather sniffs and analyzes network traffic or system logs for information that can be used in subsequent attacks.[12]
- Backdoors: A backdoor is a method of bypassing the usual procedures for authenticating a user to attain unauthorized access to the system. This can be affected through the installation of malware or intentionally being inbuilt in software to allow the hacker to administer systems remotely [13,14].
- Denial of Service: Such classes of attack are characterized by the unreadiness of a system or network resource to serve its legitimate users due to the attack site being saturated by a vast volume of deceitful requests, thus resulting in a service disruption [15,16].

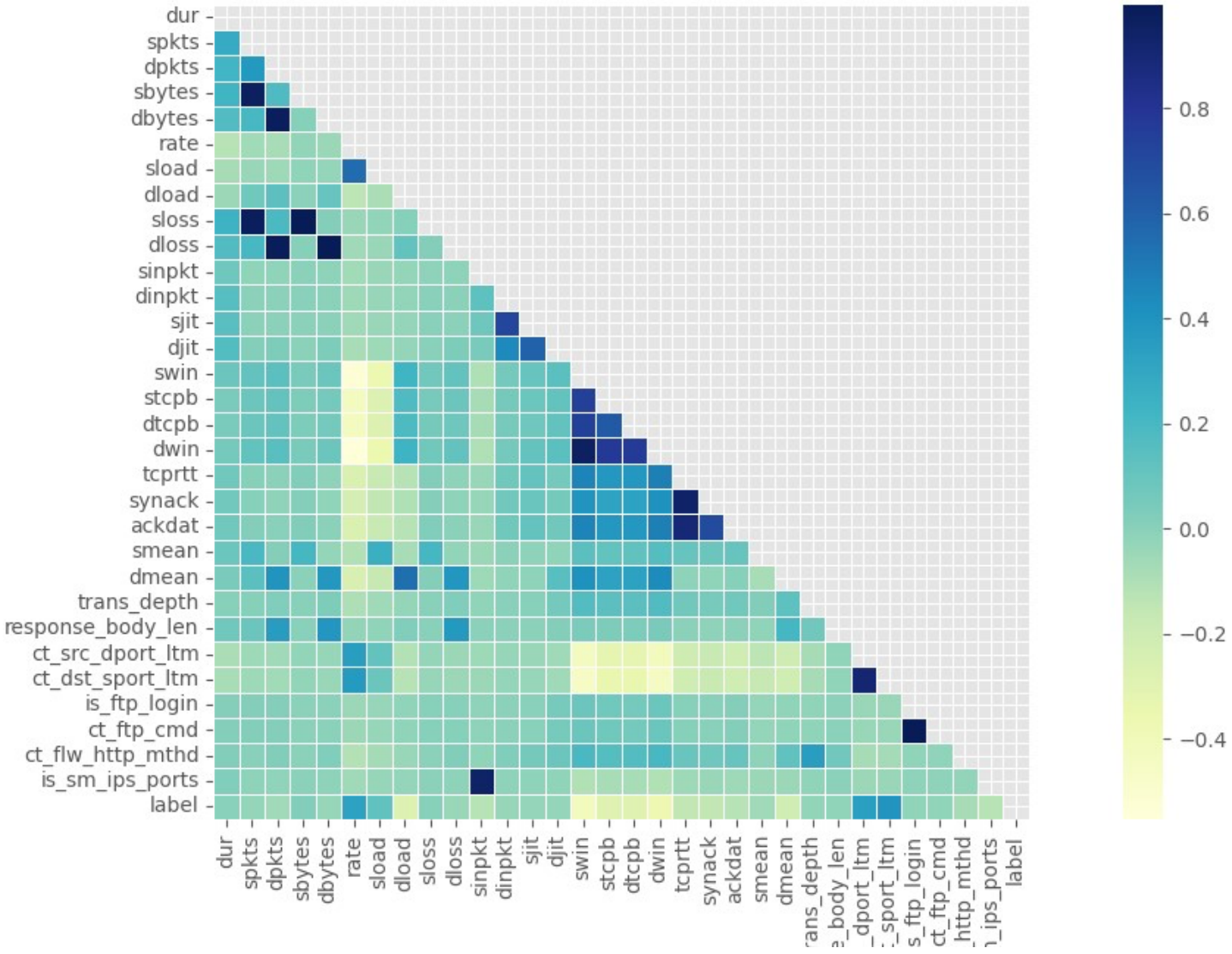

Fig .1. Features Correlation Matrix

- Exploits: These refer to the capability of taking advantage of vulnerabilities in software, hardware, or network protocols to execute malicious code or gain unauthorized access. Normally, exploits are used as tactics together with other methods of attack to achieve a particular objective.[17]
- Generic: This kind of attack is related to methods that can target any cipher or encryption algorithm without regard for its structure. The aim here is to compromise the security mechanisms regardless of the applied method of encryption.[18]
- Reconnaissance: This is essentially information gathering; hacking attempts target scanning and probing the networks for active devices, open ports, and vulnerabilities that can be used in future attacks.[19]
- Shellcode: Small pieces of codes used as a payload in exploits to take control of a system. The code would generally be injected into a vulnerable program to open a command shell or execute any other arbitrary commands on the target machine.
- Worms: This is self-replicating malware that propagates across the network, using or exploiting certain vulnerabilities in other systems. Unlike viruses, worms do not need attachment to an existing program and propagate on their own. They do really extensive damage. Fig 2. Shows distribution of attacks categories.[20,21]

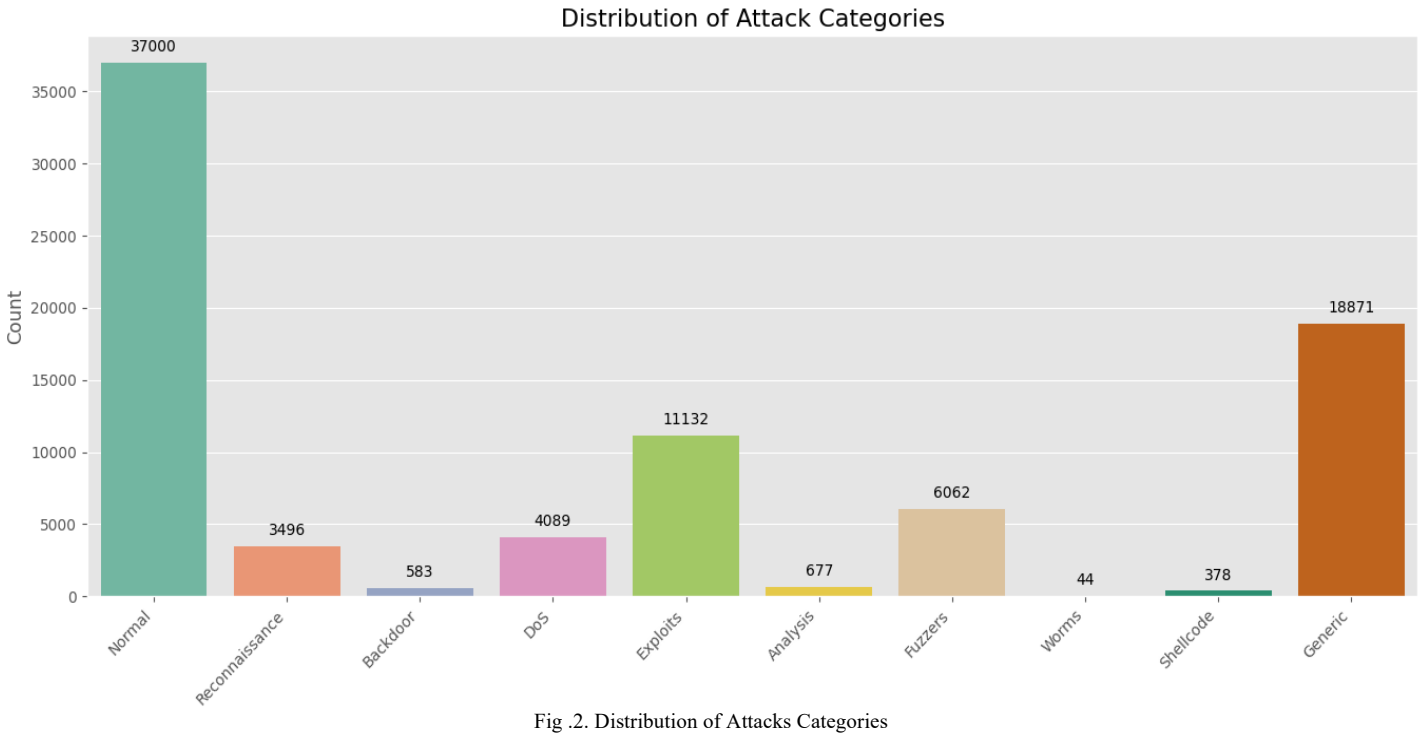

Fig .2. Distribution of Attacks Categories

Each of these attack types presents challenges for network security in another way and illustrates the need for enhanced detection mechanisms, such as the one proposed, 1D-CNN-based IDS. The involvement of these different categories makes the UNSW-NB15 dataset thoroughly test an IDS to make it resilient against various kinds of threats coming into practice in real-world multi-cloud environments. The dataset used in this study includes features as shown in Table 1.

TABLE I. UNSW-NB15 DATASET FEATURES

Feature	**Description**
dur	Duration of the connection in seconds
proto	Protocol used, such as TCP, UDP, etc.
service	Network service on the destination, e.g., HTTP, FTP, - for no service
state	State of the connection, e.g., FIN, INT
spkts	Source to destination packet count
dpkts	Destination to source packet count
sbytes	Source to destination byte count
dbytes	Destination to source byte count
rate	Number of packets per second for the connection
sload	Source bits per second
dload	Destination bits per second
sloss	Source packets retransmitted due to loss
dloss	Destination packets retransmitted due to loss
sinpkt	Time between packets sent by the source
dinpkt	Time between packets received by the destination
sjit	Source jitter
djit	Destination jitter
swin	Source TCP window advertisement
stcpb	Source TCP base sequence number

dtcpb	Destination TCP base sequence number
dwin	Destination TCP window advertisement
tcprtt	TCP connection setup round-trip time
synack	TCP connection setup time for SYN-ACK
ackdat	TCP connection setup time for ACK
smean	Mean of the bytes sent per packet by the source
dmean	Mean of the bytes sent per packet by the destination
trans_depth	Layer in the protocol stack where the data is located
response_body_len	Content size of the HTTP response
ct_src_dport_ltm	Count of source IP addresses to specific destination ports within 100 connections
ct_dst_sport_ltm	Count of destination IP addresses to specific source ports within 100 connections
is_ftp_login	Indicates if the login is to an FTP server (1 if yes, 0 if no)
ct_ftp_cmd	Number of FTP commands
ct_flw_http_mthd	Number of HTTP methods
is_sm_ips_ports	Indicates if source and destination IP addresses and ports are identical
attack_cat	Attack category (e.g., Normal, Fuzzers, DoS, etc.)
label	Binary label (1 for attack, 0 for normal)

3.2 Model Description

The model used for this research is 1D-CNN, developed to detect network intrusions in a multi-cloud environment. Traditionally, CNNs are applied to image recognition tasks, though they have proved very effective in the analysis of sequential data, given the fact that it automatically learns the spatial hierarchies of features through convolutional layers. On the other hand, in intrusion detection, 1D-CNN treats network traffic data as one-dimensional sequences. It thus creates a normal class against many attack types by finding patterns.[22]

The architecture of the 1D-CNN model typically consists of a variety of layers; each plays a different role in the transformation and extraction of the input data into meaningful features. It consists of multiple convolutional layers to apply filters on the input data; these are followed by pooling layers, which reduce the dimensionality of the feature maps by giving important features that are more striking. [23,24]

These layers will help in the extraction of local dependencies from the network traffic data representing potential intrusions. The last layers of the 1D-CNN are fully connected, or dense, which merge the extracted features to make the final classification. This model has a softmax activation function in the output layer, which makes this model give a probability distribution across classes and, therefore, predict the category of network traffic-whether it is normal or one of the attack types.

Table 2. summarizes the architecture of the 1D-CNN model used in this study.

TABLE II. PROPOSED 1D-CNN MODEL ARCHITECTURE

Layer Type	**Layer Details**	**Output Shape**
Input	1D Network Traffic Data	(Batch Size, 43, 1)
Conv1D	32 filters, kernel size 3, ReLU activation	(Batch Size, 41, 32)
MaxPooling1D	Pool size 2	(Batch Size, 20, 32)
Conv1D	64 filters, kernel size 3, ReLU activation	(Batch Size, 18, 64)
MaxPooling1D	Pool size 2	(Batch Size, 9, 64)
Flatten	Flatten the input	(Batch Size, 576)
Dense	128 units, ReLU activation	(Batch Size, 128)
Dense	10 units (number of classes), softmax activation	(Batch Size, 10)

4. RESULTS AND DISCUSSION

The performance of the proposed CNN-based IDS is assessed using the UNSW-NB15 dataset. Key performance indicators will therefore focus on the evaluation metric of accuracy, precision, recall, and F1-score derived from the UNSW NB 15

dataset. Training accuracy achieved is about 84.02%, good enough to depict excellent generalization capability with validation accuracy of about 82.79% on unseen data.
The validation results showed that it is quite good in segregating between normal and malicious network traffic, reflected in the precision with 80.91% and recall with 82.79%. The F1-score, which is the balanced mean of precision and recall, further underlined the robustness of the model with the value of 81.28%. Indeed, these results can indicate a new possibility of CNN-based IDS with better reliability, in respect to multi-cloud security to detect varieties of network intrusions.

4.1 Discussion on Loss Graph Results

The graph of loss provides critical insights into the learning dynamics of the CNN-based IDS over the course of training. In the initial stages, both the training and the validation losses take a drastic turn down within the first 10 epochs, which shows that the model learned the fundamental pattern in the network traffic quite rapidly. This sharp decrease in loss could indicate that the learning process is effective, such that over time the model can indeed become increasingly confident in distinguishing between normal activities versus malicious ones.
While with further training the training loss continued to go down, reaching a value of about 0.41 at the 50th epoch, this is already a good omen: the model keeps on improving as it minimizes the error on the training data. However, most importantly, one considers the fact that the validation loss, while also falling first, starts to get levelled after about the 30th epoch at around 0.45, as shown in Fig 3.

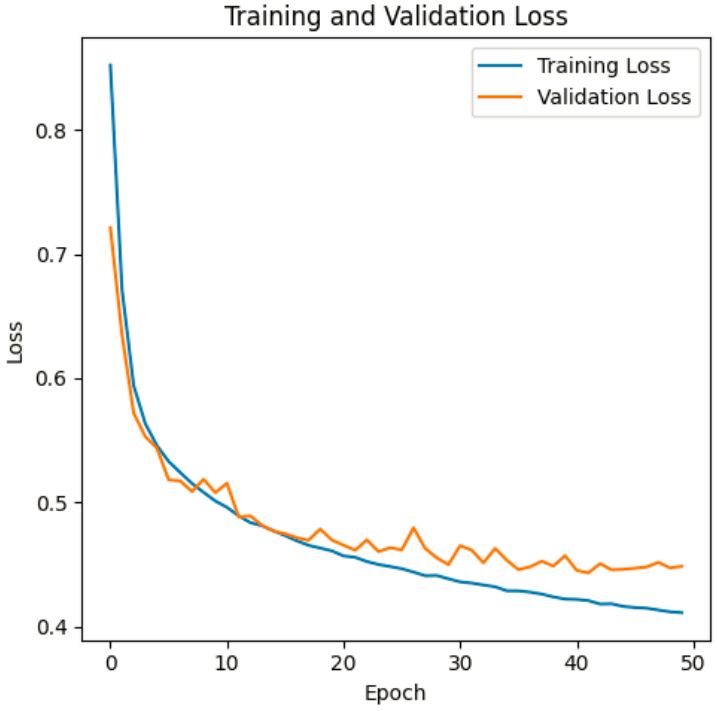

Fig .3. Training and Validation Loss

This plateauing effect, put against the fact that the validation loss is still a little higher than the training loss, implies the fact that while the model generalizes rather well to unseen data, its potential for more improvements on the validation set diminish beyond a certain point.
This small gap between training and validation loss is remarkable. This gap therefore indicates that the model does not overfit significantly, which is a common problem of deep learning models when they have been trained for a very long time. Generally speaking, overfitting occurs when the performance of the model is good on the training data but poor on new, unseen data, reflected by a large difference between the training and validation loss.
Here, the relatively small gap suggests that the model generalizes reasonably well while still being fit to the training data. It means that the applied regularization methods and architecture are doing a good job of at least somewhat mitigating overfitting.
However, the plateau of validation loss being higher than training loss shows there is still room for improvement. This can be attained by further tuning the hyperparameters of the model, either by changing the learning rate, adding dropout layers to avoid overfitting, or using a different architecture of the network. Additionally, slight increases in model complexity or

numbers of variations in the dataset can help further improve the model's performance on the validation set and bring it closer to convergence of training and validation loss for generalization.
This loss graph speaks to a well-optimized model with efficacy but also points to areas of fine-tuning that might improve performance even further-especially in narrowing down the difference between training and validation loss, and further improvement by more epochs.

4.2 Discussion on Accuracy Graph Results

The accuracy graph brings into view the evolution and maturation process of the 1D-CNN-based IDS in training. The first 10 epochs are full of a rapid increase in both training and validation accuracy because the model quickly picked up most of the basic features it needs to tell normal from malicious network traffic. This rapid improvement in early epochs is indicative of the fact that the model indeed does a good job of capturing the essential patterns within the dataset.
As training progresses, the training accuracy continues its upward trajectory until reaching approximately 84% by the 50th epoch. The increase in training accuracy steadily, therefore, would suggest that a model learns incrementally to make correct predictions on the training data while it is fine-tuning to capture the subtle features defining different attack types. Such a smooth increase in training accuracy without any sudden drops or fluctuation is generally a good sign as far as the stability in model learning and the appropriateness of the selected learning rate and architecture go, as shown in Fig 4.

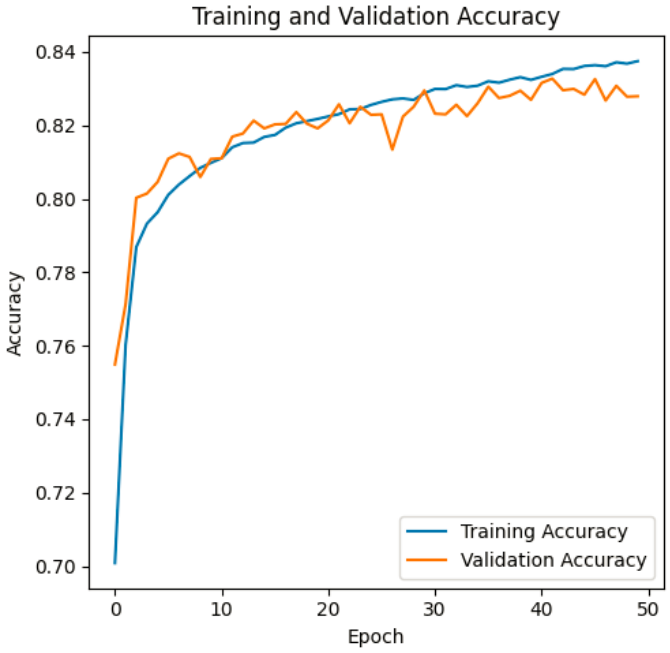

Fig .4. Training and Validation Accuracy

While the training accuracy increases, the validation accuracy, which follows it during the first parts of training, starts to flatten after around epoch 30 and levels out at approximately 83%. This stagnation of the validation accuracy after a while is indicative of the fact that for this model, there comes a point beyond which the law of diminishing returns sets in regarding generalizing on unseen data.
The close alignment of training and validation accuracy across epochs assures that overfitting is not a considerable problem, which is also supported by the fact that the loss graph illustrates similar patterns. However, this plateau now reached denotes that further increases in the number of training epochs are not likely to bring substantial improvements in generalizing performance from this model.
The training and validation accuracy values are closely aligned. It gives reason that the model is balancing between the abilities of learning from the training data and generalizing to new, unseen examples. That these two metrics are not very different reinforces the earlier conclusion that this model is not overfitting, something so crucial for real-world, reliable performance. On the other hand, this model probably has exploited the information that it could get from the present architecture and feature set, as the validation accuracy plateaus at around 83%.
In light of these observations, further improvement is conceivable, even though the already trained model produces an impressive accuracy of about 83%. This may require some changes in the model architecture to try to break through this

plateau to push higher on the validation accuracy, either with additional layers or trying other neural network architectures that can capture more complex features in the data.
It is probably due to further refinement in the process of feature engineering, including other relevant features or taking a feature selection process so efforts can be concentrated on the most predictive variables. Hyperparameter tuning could be extended to learning rate schedules and batch sizes, besides regularization techniques that may help in further optimizing the performance of the model.
In short, the accuracy graph reflects that the model is doing an effective job of learning the process and stands firm in performance. However, because of a plateau in validation accuracy, to achieve more gains, the use of more advanced techniques or modifications becomes necessary. Thus, this points out the venue for future research and development in optimizing this IDS to obtain higher accuracy in network intrusion detection.

4.3 Discussion on Confusion Matrix Results

The confusion matrix is an insight into the elaborated performance of the proposed model based on 1D-CNN for network activities of multiple classes and hence highlights its strengths and weaknesses with regard to intrusion detection. The resultant matrix spans across 10 different classes representing various types of network traffic including normal activities and several categories of network attacks.
Each diagonal element in the matrix represents the number of correct classifications for a class in question, while the off-diagonal elements represent misclassifications of a model since it has incorrectly identified one class for another.
One of the striking observations of the confusion matrix is that the high accuracy with which the model classified normal network traffic denotes 7,146 good classifications, reflecting strong strength and a very positive ability to reveal benign activities from potentially malicious ones. This may be one of the most key points in the accuracy classification of normal traffic, which minimizes the chance of false positives-the normal behavior being flagged as an attack-and hence minimizes unnecessary alerts by focusing attention on actual threats, as shown in Fig 5.

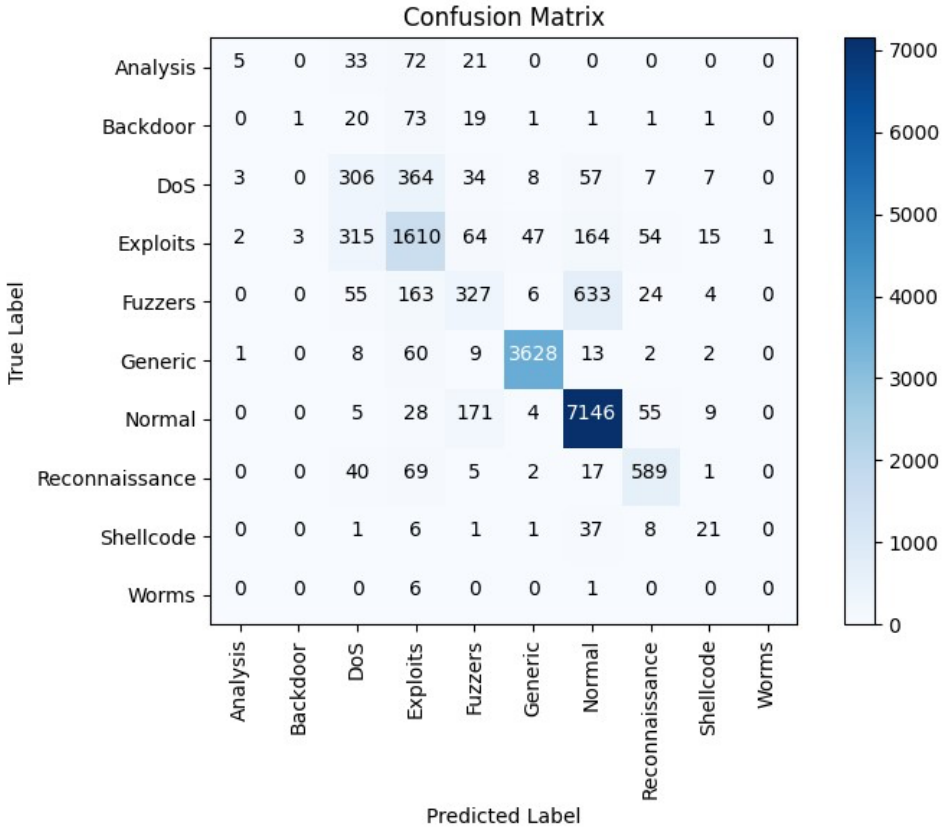

Fig .5. Confusion Matrix of Proposed 1D-CNN

It does, however, indicate a significant level of confusion between some attack types, predominantly between DoS and Exploits. The model misclassified 364 DoS attacks as being from the class Exploit and 315 instances of Exploits as

belonging to DoS. This therefore means that these two kinds of attacks are adequately similar in feature space that the model has a hard time telling them apart. This is possibly with regard to the nature of such an attack, which may employ similar vulnerabilities or have similar network behaviors that would make the two overlaps in a way capable of bringing in classification errors.

Another strong point of the model can be seen in the Generic attack category, which was correctly classified 3,628 times. This would lead me to believe that the model learned about this type of attack quite well, perhaps because there was a well-defined pattern or it simply had more sizeable examples with which to generalize well for this class.

On the other hand, this matrix shows the challenges brought about by class imbalance, especially regarding attack types like Worms, Shellcode, and Analysis, due to their rather limited number of instances in the dataset. In all probability, the relatively small number of examples for such classes contributed to lower prediction accuracy because the model had fewer opportunities to learn from the distinctive features of these less common kinds of attacks. The result might be a model that performs on imbalanced datasets, usually skewing towards the classification of more frequent classes and performing badly on the rare ones.

One interesting observation that might arise here is the misclassification of Fuzzers into normal traffic, with 633 instances being labeled as benign. This may be a big problem because this could lead to a situation whereby threats were overlooked by the model, and malicious activities would pass through without detection. The Fuzzer attack involves sending malformed or unexpected inputs to a system in search of vulnerabilities, and the nature of this attack might be very subtle to detect.

Put differently, the confusion matrix has underlined the dual nature of model performance-very good for recognizing normal traffic and some types of attacks, such as Generic, mediocre regarding the recognition of similar attack categories such as DoS and Exploits. This is also underlined by the misclassification of Fuzzers as normal traffic.

These insights further confirm that more feature engineering-for example, working on an improved set of features to better capture the salient properties of confused attack types-or model tuning, such as better adjustments to class weights for the purpose of overcoming imbalance, might lead to higher accuracy and robustness in the wide range of network intrusion detection tasks.

4.4 Discussion on ROC Curve Results

The ROC curve shows the deep analysis of the CNN-based model, which can identify various classes of network activities and, at the same time, present both the overall and the class-specific performance about intrusion detection. The dotted pink line represents the micro-average ROC curve with an AUC value of 0.99. This almost perfect AUC signifies that the model performs well in classification for all classes, showing high precision in differentiating between benign and malicious network traffic.

This high AUC suggests that this model is well-calibrated and thus especially suited to the multi-class classification problem where the correct identification of several types of attacks is rather necessary, as shown in Fig 6.

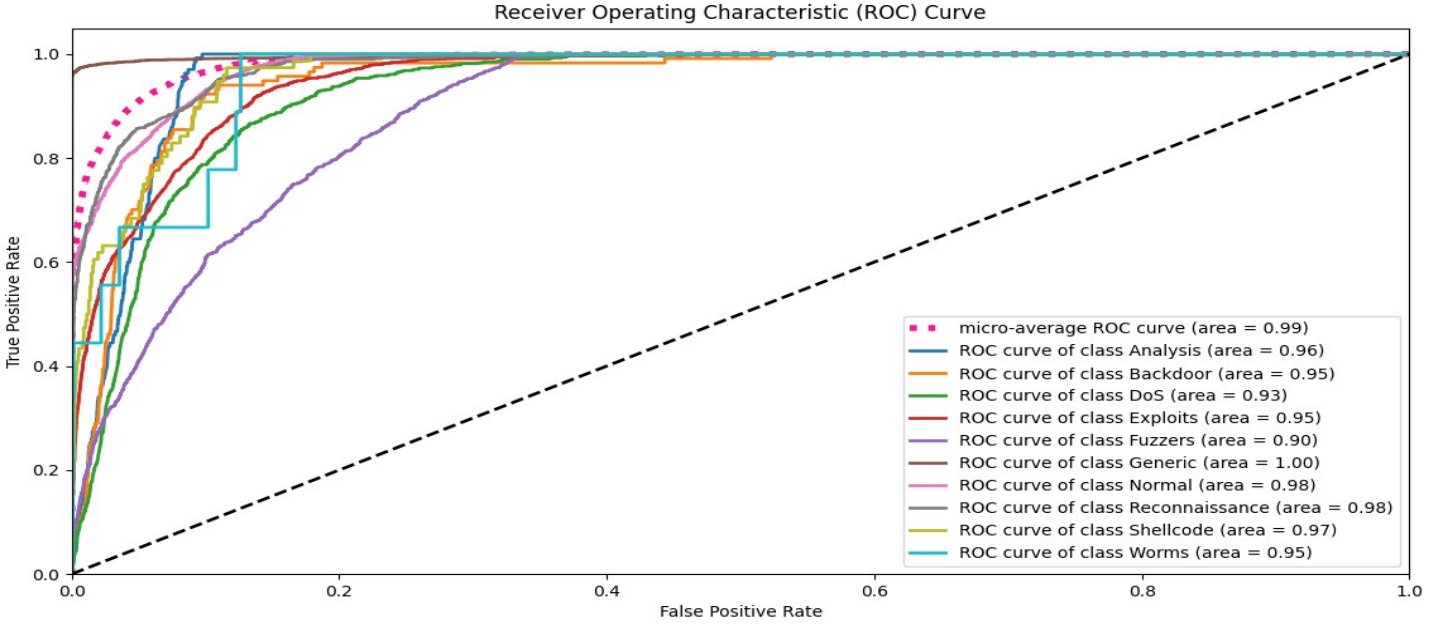

Fig .6. ROC Curve of Proposed 1D-CNN Model

Diving into the classes-specific performance, most classes see their AUC values within a range of 0.95-1.00, further evidencing the strong discriminatory power of the model for such classes of an attack. Notably, the Generic class of an attack provides the AUC value of 1.00; this therefore suggests that the model perfectly separates this class from all other classes without any misclassifications.

This perfection in identifying Generic attacks might be attributed to the distinct patterns in the data that the model has learned effectively. For instance, normal traffic is highly represented by a very high AUC of 0.98, which justifies the robustness of the model in identifying benign activities and reducing the likelihood of having false alarms-a critical characteristic in minimizing unnecessary security alerts.

However, with 0.90, the Fuzzers class is the lowest in all classes, though still a good performance. This agrees with the confusion observed in the confusion matrix, where Fuzzers were often confused as normal traffic. Though this was the lowest AUC among the classes, the value of 0.90 still indicates good discrimination ability, though it highlights a relative weakness in the model's ability to distinguish Fuzzers from other classes.

This is perhaps because Fuzzer attacks are insidious and come in various forms, unlike other attack types, and may thus not clearly give rise to discernible patterns that can easily be detected.

The actual shapes of the ROC curves provide further insights into the model performance. Most of the curves go up very fast in the region near the y-axis, which indicates that the model has achieved a high true positive rate when the false positive rate is still very low. This is the most favorable feature in a security system, since the efficiency to detect true threats will not be dampened by too many false alarms. Therefore, any system which can hold high detection rates along with reduced false positives increases the overall trustworthiness and efficiency of security.

All ROC curves lie well above the diagonal dashed line representing a random classifier. Such a huge difference again reassures that the model, for all classes, has strong predictive power, considerably higher than the performance of a random guess. The reliability and effectiveness of the model in a real-world network security environment are further underlined by the capability of the model to keep on outperforming a random classifier across all kinds of attacks and normal traffic.

The ROC analysis also portrays a capable model that is doing an exceptional job of discriminating among network activities. Based on the consistently high AUC values across the board, this model is promising and will perform well in deployment in network security settings. It enables the detection of various attack types with a high degree of accuracy, while the ratio of false positives remains low. This makes it a strong tool for improving multi-cloud environments' security posture.

5. CONCLUSIONS

This paper has demonstrated the performance of a 1D-CNN-based model on intrusion detection in a multi-cloud environment using the UNSW-NB15 dataset. The model performance was fairly brilliant with a training accuracy of about 84%, while the validation accuracy is approximately 83%, showing good generalization. Precise analysis of various metrics, such as loss, accuracy, ROC curves, and the confusion matrix, underlines the strengths, at the same time revealing parts that need further refinement.

This is exceptionally good in classifying normal traffic and certain types of attacks, such as Generic, which the high AUC values and significant correctness of classification prove. This analysis again shows challenges, especially in the differentiation of similar attack types like DoS and Exploits, and the spotting of the rarer classes like Fuzzers sometimes got misclassified as normal traffic.

These observations thereby hint that while the present model is strong and powerful, it has more room for enhancement-such class imbalance issues, further refinement of feature engineering, deeper architecture, or hyperparameter tuning-which elevates its performance to newer heights. This work demonstrates a commitment to the potential of machine learning models, in particular CNNs, to improve security in multi-clouds by providing accurate and reliable intrusion detection.

Research presented herein is a contribution to one of these active developments: to create sophisticated, adaptive security mechanisms which will be able to resist a wide range of cyber threats effectively on increasingly complex cloud infrastructures, and thus will be of added value for both network security professionals and researchers alike.

Funding

The author's paper explicitly states that no funding was received from any institution or sponsor.

Conflicts Of Interest

None.

Acknowledgment

The author would like to express gratitude to the institution for their invaluable moral support throughout this research project.

References

[1] A. Vinolia, N. Kanya, and V. N. Rajavarman, "Machine learning and deep learning based intrusion detection in cloud environment: A review," in 2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT), 2023, pp. 952–960.

[2] S. Devi and D. A. K. Sharma, "Understanding of intrusion detection system for cloud computing with networking system," Int. J. Comput. Sci. Mob. Comput., vol. 9, no. 3, pp. 19–25, 2020.

[3] L. Gupta and T. Salman, "Cybersecurity of multi-cloud healthcare systems: A hierarchical deep learning approach," Department of Computer Science, University of Missouri-St. Louis, St. Louis, USA, 2022.

[4] C. J. S. Mary and K. Mahalakshmi, "Modelling of intrusion detection using sea horse optimization with machine learning model on cloud environment," Department of Computer Science and Engineering, Idhaya Engineering College for Women, Kallakurichi, Tamil Nadu, India, 2024.

[5] Q. Liu and D. Wang, "A multi-task based deep learning approach for intrusion detection," SILC Business School, Shanghai University, Shanghai, China, 2022.

[6] S. Krishnaveni, S. Sivamohan, S. Sridhar, and S. Prabhakaran, "Network intrusion detection based on ensemble classification and feature selection method for cloud computing," Concurr. Comput. Pract. Exp., vol. 34, no. 11, p. e6838, 2022.

[7] W. Elmasry, A. Akbulut, and A. H. Zaim, "A design of an integrated cloud-based intrusion detection system with third party cloud service," Open Comput. Sci., vol. 11, no. 1, pp. 365–379, 2021.

[8] P. S. Negi, A. Garg, and R. Lal, "Intrusion detection and prevention using honeypot network for cloud security," in 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2020, pp. 129–132.

[9] B. T. Devi, S. Shitharth, and M. A. Jabbar, "An appraisal over intrusion detection systems in cloud computing security attacks," in 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), 2020, pp. 722–727.

[10] M. Kumar and A. K. Singh, "Distributed intrusion detection system using blockchain and cloud computing infrastructure," in 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI), 2020, pp. 248–252.

[11] A. Guezzaz, A. Asimi, Y. Asimi, M. Azrour, and S. Benkirane, "A distributed intrusion detection approach based on machine learning techniques for a cloud security," in Intelligent Systems in Big Data, Semantic Web and Machine Learning, Cham: Springer, 2021, pp. 85–94.

[12] N. D. Patel, B. M. Mehtre, and R. Wankar, "Od-ids2022: Generating a new offensive defensive intrusion detection dataset for machine learning-based attack classification," Int. J. Inf. Technol., vol. 15, pp. 4349–4363, 2023.

[13] A. S. Iliyasu and H. Deng, "N-GAN: A novel anomaly-based network intrusion detection with generative adversarial networks," Int. J. Inf. Technol., vol. 14, pp. 3365–3375, 2022.

[14] S. P. Usha Kirana and D. A. D'Mello, "Energy-efficient enhanced particle swarm optimization for virtual machine consolidation in cloud environment," Int. J. Inf. Technol., vol. 13, pp. 2153–2161, 2021.

[15] R. Keshri and D. P. Vidyarthi, "Communication-aware, energy-efficient VM placement in cloud data center using ant colony optimization," Int. J. Inf. Technol., vol. 15, pp. 4529–4535, 2023.

[16] K. Srinivas, N. Prasanth, R. Trivedi, et al., "A novel machine learning inspired algorithm to predict real-time network intrusions," Int. J. Inf. Technol., vol. 14, pp. 3471–3480, 2022.

[17] J. Wei, C. Long, J. Li, and J. Zhao, "An intrusion detection algorithm based on bag representation with ensemble support vector machine in cloud computing," Concurr. Comput. Pract. Exp., vol. 32, no. 24, p. e5922, 2020.

[18] N. M. Ibrahim and A. Zainal, "A distributed intrusion detection scheme for cloud computing," Int. J. Distrib. Syst. Technol. (IJDST), vol. 11, no. 1, pp. 68–82, 2020.

[19] M. G. Raj and S. K. Pani, "A meta-analytic review of intelligent intrusion detection techniques in cloud computing environment," Int. J. Adv. Comput. Sci. Appl., vol. 12, p. 10, 2021.

[20] O. Alkadi, N. Moustafa, B. Turnbull, and K. K. R. Choo, "A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks," IEEE Internet Things J., vol. 8, no. 12, pp. 9463–9472, 2020.

[21] S. Krishnaveni, S. Sivamohan, S. S. Sridhar, and S. Prabakaran, "Efficient feature selection and classification through ensemble method for network intrusion detection on cloud computing," Clust. Comput., vol. 24, no. 3, pp. 1761–1779, 2021.

[22] M. Alweshah, S. Alkhalaileh, M. Beseiso, M. Almiani, and S. Abdullah, "Intrusion detection for IoT based on a hybrid shuffled shepherd optimization algorithm," J. Supercomput., vol. 78, no. 10, pp. 12278–12309, 2022.

[23] N. Usman, S. Usman, F. Khan, M. A. Jan, A. Sajid, M. Alazab, and P. Watters, "Intelligent dynamic malware detection using machine learning in IP reputation for forensics data analytics," Future Gener. Comput. Syst., vol. 118, pp. 124–141, 2021.

[24] B. K. Pandey, M. R. M. Veeramanickam, S. Ahmad, C. Rodriguez, and D. Esenarro, "ExpSSOA-Deep maxout: Exponential shuffled shepherd optimization based deep maxout network for intrusion detection using big data in cloud computing framework," Comput. Secur., vol. 124, p. 102975, 2023.