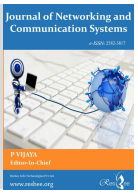
Vol.7 No.4 Oct 2024

Journal of Networking and Communication Systems

Received 26 August, Revised 30 August, Accepted 15 October



Mahalanobis City Block distance-rider Optimization Algorithm-Based Neural Network enabled intrusion detection in the Internet of Things

T. Manojkumar

Assistant Professor, Department of Electrical and Electronics Engineering,
J. P COLLEGE OF Engineering, Tamil Nadu.

Abstract: The Internet of Things (IoT) has great potential for a variety of applications ranging from healthcare automation to defensive networks. The security of IoT networks is substantially paramount to the security of underlying communication and computing infrastructure. However, owing to less constrained resources and computation capacities, the IoT network is highly prone to several attacks. Therefore, protecting IoT networks from various attacks is of vital importance. One of the significant security systems is the Intrusion Detection System (IDS). In this research, Mahalanobis City Block distance-Rider Optimization Algorithm-Based Neural Network (MCB distance-RideNN) is presented for intrusion detection in IoT. The system model of IoT is simulated initially and the communication is accomplished with each IoT node. Thereafter, data is stored in the form of features on a server. Next, the intrusion detection process is carried out by initially obtaining input log file data from the NSL-KDD dataset and thereafter, Min-Max normalization is employed to conduct data normalization. After that, feature selection is conducted based on the Mahalanobis City Block (MCB) distance. However, MCB distance is newly presented by integrating Mahalanobis distance and City Block distance. Lastly, intrusion detection is carried out employing RideNN. In addition, MCB distance-RideNN attained 91.185% accuracy, 92.574% True Negative Rate (TNR), and 90.412% of True Positive Rate (TPR).

Keywords: Internet of Things, Intrusion Detection, Mahalanobis Distance, City Block Distance, Ridenn.

Nomenclature

Abbreviation	Expansion
NIDS	Network Intrusion Detection System
ML	Machine Learning
DL	Deep Learning
DL-IDS	Deep Learning-based Intrusion Detection System
DIS-IoT	Deep Integrated Stacking for the IoT
FNR	False Negative Rate
FPR	False Positive Rate
NSBPSO-DCNN	Neighborhood Search-Based Particle Swarm Optimization based Deep Convolutional Neural Network
ROA	Rider Optimization Algorithm
NN	Neural Network

1. Introduction

IoT has emerged as a newer computing paradigm, which is transforming our lifestyles [3]. Currently, several IoT applications have increased considerably. IoT comprises more count of diverse sensors or physical end-points. These are linked to each other as well as the internet. They are capable of gathering data from neighboring platforms and sharing them with each other. The tiny devices with fewer resources and minimum power are also capable of transmitting information to IoT gateways, whereupon it is aggregated and gathered to an end-point network on the internet. However, the most essential and crucial challenge of the IoT environment is security [4]. Security in the IoT environment becomes a highly challenging trouble in association with IoT industry development. It is mostly owing to the heterogeneity of the IoT structure, diverse sorts of accessible devices, and numerous communication techniques [12] [1] as well as the large quantity of information being transferred by a network [1]. Amongst every security process, intrusion detection is one of the significant security systems. NIDS is termed an effective solution

© Resbee Info Technologies Pvt Ltd
https://doi.org/10.46253/jnacs.v7i4.a4

for detecting intrusions of malevolent attributes in the IoT network. NIDS is mostly offered by network layers in IoT that have the spine for connecting diverse IoT devices [4].

Intrusion detection is classified in accordance to detection techniques namely, anomaly-enabled and signature-enabled methods [13] [2]. The signature-enabled detection method employs traffic signatures for recognizing attacks. To work better, this kind of technology needs the latest signature databases. On the other side, anomaly-enabled detection techniques try to detect attacks by evaluating actual traffic initially and determining anomalies whenever deviations between assessed traffic and monitored traffic exceed the pre-defined threshold [2]. Anomaly-enabled detection employs several approaches for identifying attacks. These techniques can be based on mathematical systems as well as ML. Recently, it has become an expansively utilized engine for anomaly-enabled IDS. Various ML approaches can be exploited including semi-supervised learning, unsupervised learning as well as supervised learning. In supervised learning, labeled input data subjected to a classifier is. Moreover, input data given to semi-supervised learning is partially labeled whereas unsupervised learning operates on unlabeled data and tries to understand it by recognizing the same patterns [14] [2]. DL is the branch of ML, which has become broadly well-known in several domains such as science, medicine, engineering, and finance. DL methods for intrusion detection have become progressively popular as it permits for highly complicated assessment of network traffic and highly accurate anomaly detection when compared with conventional ML schemes [2].

The significant aim is to present MCB distance-RideNN for detecting intrusion in IoT. IDS is one of the key components of distinctive security construction that offers visibility into system activities, enabling prompt detection and responses to any unacceptable events. Initially, the IoT system model is simulated and the communication is performed with individual IoT nodes. Then, data is stored in feature form on a server. Thereafter, the intrusion detection process is accomplished by acquiring log file data as input from the NSL-KDD dataset and next, data normalization is carried out utilizing Min-Max normalization. Afterward, feature selection is conducted based on MCB distance, which is modeled by combining Mahalanobis distance and City Block distance. Finally, intrusion detection is carried out utilizing RideNN.

An essential contribution is manifested as follows.

- **Proposed MCB distance-RideNN for intrusion detection in IoT:** IDS is the solution for addressing privacy and security challenges with the detection of diverse IoT attacks. Here, feature selection is performed by MCB distance which is newly introduced by joining Mahalanobis distance and City Block distance. Furthermore, intrusion detection is accomplished by utilizing RideNN.

The layout of the remaining sections is manifested as follows: Literature overview of classical intrusion detection techniques and their demerits are explained in section 2, system model of IoT is revealed in section 3 whereas section 4 elaborates methodology of MCB distance-RideNN for intrusion detection in IoT, section 5 mentions outcomes of MCB distance-RideNN and section 6 mentions conclusion of MCB distance-RideNN.

2. Motivation

Pervasive development of IoT is visible around the world and IoT security has become an essential concern. Therefore, there is a vital need for IDS devised for IoT networks to mitigate IoT-related security attacks. Traditional IDSs are not a choice for IoT environments owing to less storage and computing capacities of IoT devices and certain models employed. This motivated to design of IDS for the IoT environment by collecting conventional intrusion detection approaches. This part presents a literature survey of traditional IDS and their shortcomings.

2.1 Literature Survey

Otoum, Y., *et al.* [1] presented DL IDS for detecting security threats in the IoT environment. It was capable of detecting every anomaly without considerable errors, but still, this technique failed to estimate with diverse classifiers, namely decision tree, random forest, and Naive Bayes, employing several databases. Lazzarini, R., *et al.* [2] introduced DIS-IoT for detecting intrusions in IoT. This technique had a competence for attack detection while maintaining low FNR as well as FPR, though it needed more resources. Awajan, A., [3] designed a Deep Neural Network for intrusion detection in IoT devices. This scheme was able to learn from any IoT networks and identify intrusions automatedly. However, inference time was increased due to the complexity of the model. Baniasadi, S., *et al.* [4] devised an NSBPSO-DCNN for accurate detection of intrusion in IoT environments. This model was capable of handling large quantities of IoT data. Nevertheless, it was less appropriate for devices with minimum processing power.

2.2 Challenges

Some of the significant challenges confronted by current IDS are explained as follows.

- DIS-IoT developed in [1] performed consistently better with diverse databases, but still, it did not estimate the computational overhead of the model.
- In [3], the introduced technique failed to design a lightweight model for detecting intrusions more efficiently and effectively.
- For several years, IDS has been proven highly advantageous to guard networks and information systems. However, traditional IDS processes on IoT are unrealistic due to some definite elements. Hence, it is vital to design intrusion detection approaches to guarantee the security, reliability, and integrity of IoT.

3. System Model of Iot

In a chain of security, the primary and most crucial step is the detection of intruder behaviors. The complete detection process is successful, only when an intruder is identified. Anti-malware software and firewalls are the foremost protectors for user terminals. In addition, IDS is a protector for internet servers. A system model of IoT is exposed in Fig. 1. There are various IoT servers and devices that are revealing openly to the public internet owing to the remote control features. An attacker captures the susceptibilities for intruding IoT servers. Therefore, IDS is necessary to secure and detect IoT servers from attackers [11].

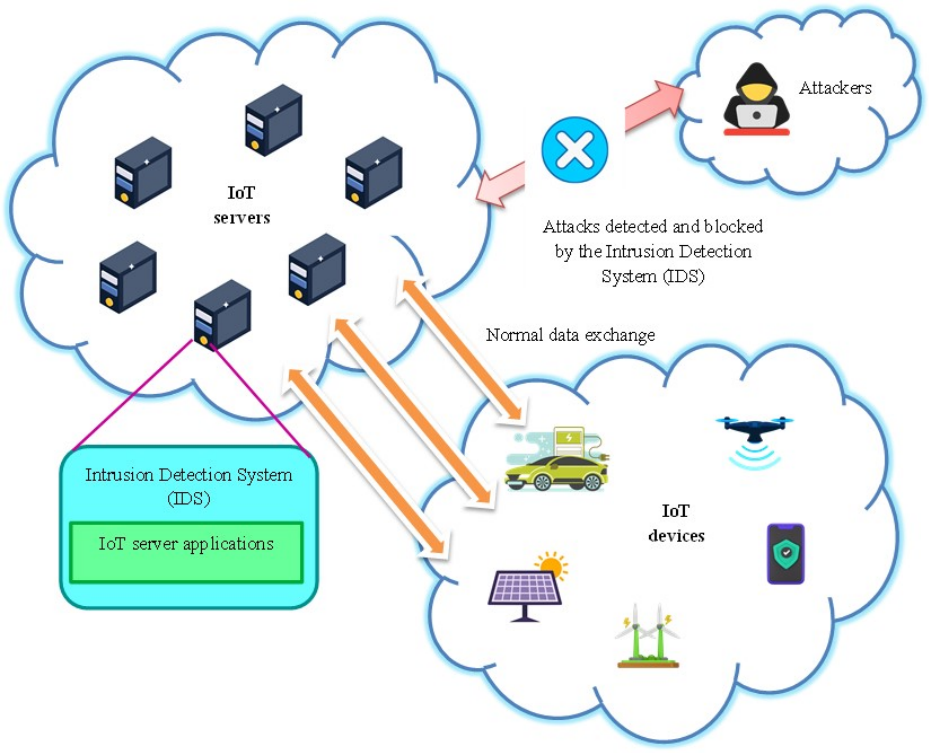


Fig. 1. *System model of IoT*

4. Proposed MCB distance-RideNN for intrusion detection in IoT

IDSs developed for IoT networks are important to avoid IoT-related security threats. Here, MCB distance-RideNN is presented for intrusion detection in IoT. Firstly, a simulation of the IoT system model is carried out and thereafter, communication is performed with each IoT node. Thereafter, data is stored in feature form on a server. Afterward, the intrusion detection process is carried out by considering an input log file data from the NSL-KDD dataset, and next, data normalization is conducted utilizing Min-Max normalization. Next, feature selection is done based on MCB distance. However, MCB distance is newly devised by incorporating Mahalanobis distance and City Block distance. At last, intrusion detection is conducted employing RideNN. Fig. 2 presents a diagrammatic illustration of MCB distance-RideNN for intrusion detection in IoT.

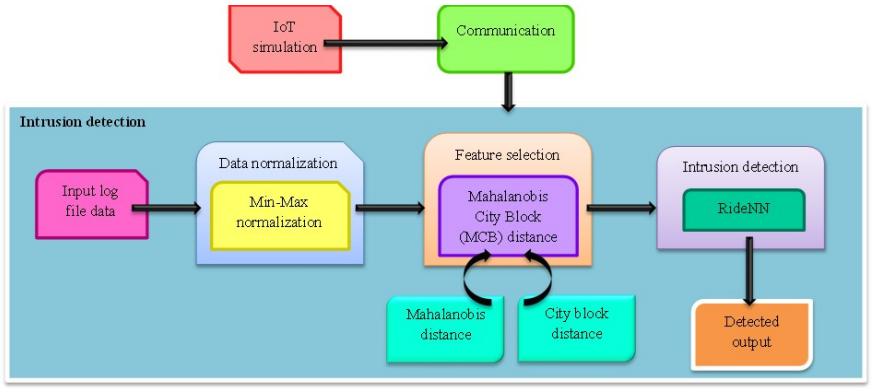


***Fig. 2**. Diagrammatic illustration of MCB distance-RideNN for intrusion detection in IoT*

4.1 Input Log File Data Acquisition

An input log file data considered to perform intrusion detection in IoT is taken from the NSL-KDD dataset [15], which can be represented as,

$$D = \{D_1, D_2, ..., D_i, .., D_c\} \tag{1}$$

where, D_i indicates i^{th} input log file data whereas D_c illustrates overall data in the dataset D. NSL-KDD dataset [15] has both testing sets and training sets in ARFF format along with complexity levels in CSV format.

4.2 Data normalization utilizing Min-Max normalization

Data normalization is a crucial pre-processing step in constructing effectual IDS. It assists in standardizing the formats of data that can decrease the impact of scale variations amongst features and enhance the performance of models. Here, D_i is considered as input to conduct data normalization by Min-Max normalization.

Min-Max normalization [6] [9] technique rescales an output or features in any range into a newer range. Generally, features are scaled from 0 to 1 or -1 to 1. An equality employed in this technique can be modeled as,

$$M_i = \frac{D_i - \min(D_i)}{\max(D_i) - \min(D_i)} \tag{2}$$

Here, D_i refers to input data, $\min(D_i)$ implies minimal value, and $\max(D_i)$ mentions maximal value whereas M_i denotes normalized data with dimension $[w \times x]$.

4.3 Feature Selection Based on MCB Distance

Feature selection is defined as a process to identify and select the subsets of appropriate attributes or features from a large group of probable features, which can be employed to efficiently design and detect intrusions within IoT devices and networks. Here, feature selection is carried out based on newly devised MCB distance. An input subjected to perform feature selection is normalized data M_i with dimension $[w \times x]$.

4.3.1 MCB Distance

MCB distance is designed by combining two distance measures Mahalanobis distance and City Block distance. Mahalanobis distance [7] is the measure of distance between a distribution and a point. In addition, it concerns a correlation of dataset and scales by data distribution. For the given samples h_a and h_m, the formulation of Mahalanobis distance can be given by,

$$A_i = \sqrt{(h_a - h_m)^{\mathrm{T}} \Sigma^{-1} (h_a - h_m)} \tag{3}$$

Here, Σ describes the covariance matrix of every data point in a database and A_i depicts Mahalanobis distance for each feature with dimension $[w \times y]$, where $x > y$. Thereafter computing the Mahalanobis distance for individual features, top y features with minimum distance are selected. Then, city block distance is applied to each selected feature.

City block distance [8] is also termed L1 distance or Manhattan distance. This measure is utilized for measuring the distance between two points in a grid-enabled system. Moreover, city block distance is computed as a sum of absolute variations of their Cartesian coordinates, which can be modeled by,

$$C_i = \sum_{b=1}^{n} \left| K_b - S_b \right| \tag{4}$$

Here, K_b and S_b indicates candidate feature and target whereas C_i symbolizes city block distance for selected feature with dimension $[w \times z]$, where $y > z$.

The selected features include src_bytes, root_shell, is_hot_login, srv_serror_rate, dst_host_srv_count and so on.

4.4 Intrusion Detection Utilizing RideNN

Intrusion detection is important for the safety and security of IoT networks. IDS detects attacks, malicious activities in the network, and unauthorized intrusion and constitutes one of the major security measures recognized in modern networks. Here, intrusion detection is carried out utilizing RideNN by taking C_i with dimension $[w \times z]$ as an input.

4.4.1 Structure of RideNN

RideNN [5] is devised by incorporating an ROA in the NN classifier. It contains layers namely input, hidden, and output, consisting of numerous neurons in all layers. RideNN network is developed by joining a neuron's output with another neuron's input in diverse layers. An input given to RideNN is manifested by,

$$l^u = \left\{ l_1^u, l_2^u, ..., l_p^u, ..., l_P^u \right\}; \quad 1 \le p \le P \tag{5}$$

where, P indicates total features. A detection process is accomplished regarding weights that are allocated to neurons in a network layer. An allocation of a neuron's weight in hidden layers can be illustrated as,

$$B = \left\{ B_1, B_2, ..., B_P \right\} \tag{6}$$

In hidden layers, the neuron is supposed to contain bias depicted by B_{P+1} and it creates a possibility. The neurons present in an output layer also comprise weights represented as B_{P+2}. In addition, B_{P+3} denotes bias in an output layer. Hence, the classifier's output can be evaluated by employing the transfer function as follows,

$$R_i = B_{P+2} * \left[\log sig \left(\sum_{p=1}^{P} l_p^u * B_p + B_{P+1} \right) \right] + B_{P+3} \tag{7}$$

Here, $\log sig$ implies a log-sigmoid transfer operation, which calculates an output from an acquired input, l_p^u signifies the input of p^{th} neuron, B_p depicts the weight of p^{th} neuron in the hidden layer, B_{P+2} manifests the weight of output, B_{P+1} and B_{P+3} represents biases. R_i Illustrates intrusion-detected output and structural overview of RideNN are delineated in Fig. 3.

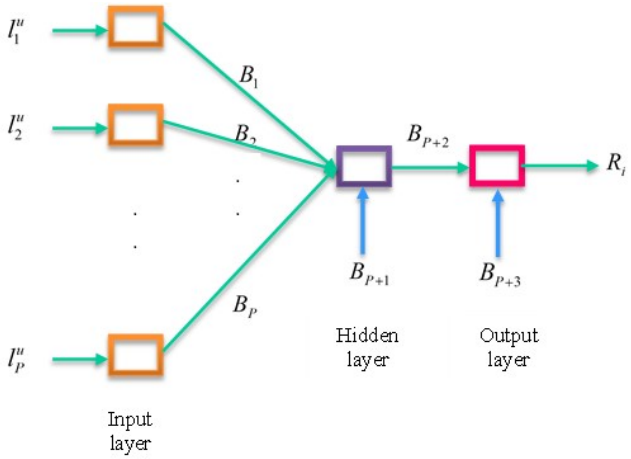


Fig. 3. *Structural Overview of RideNN*

5. Results and Discussion

The assessment outcomes of MCB distance-RideNN devised for intrusion detection in IoT are interpreted in this part.

5.1 Experiment Setup

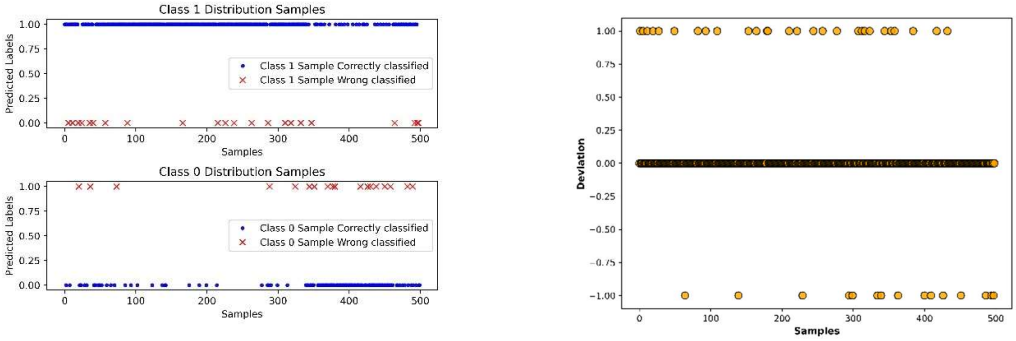
An implementation of the introduced MCB distance-RideNN is carried out in the PYTHON tool.

5.2 Dataset Description

NSL-KDD dataset [15] consists of testing sets and training sets with binary labels in ARFF format. In addition, this dataset comprises attack-type labeling and also, its complexity levels in a format of CSV.

5.3 Experimentation Outcomes

Fig. 4 specifies the experimental outcomes of MCB distance-RideNN. Fig. 4 (a) Prediction plot comprising class 0 distribution samples and class 1 distribution samples. In class 1 distribution samples, densely present samples represent accurate detection of intrusion whereas dispersed samples indicate inaccurate intrusion detection, Moreover, a minimal count of dispersed samples specify the detection of non-intrusions perfectly and a smaller number of samples illustrate inaccurate detection of non-intrusions in class 0 distribution samples. Fig. 4 (b) shows a deviation graph for samples. The value of deviation=0 represents accurate detection of intrusion whereas a deviation value of either 1 or -1 manifests inaccurate intrusion detection.



(a) (b)

Fig. 4*. Experimental results of MCB distance-RideNN, a) Prediction plot, b) Deviation graph*

5.4 Evaluation Metrics

The metrics taken into concern for evaluating MCB distance-RideNN are accuracy, TPR, and TNR.

5.4.1 Accuracy

Accuracy [10] is defined as a ratio of actually detected intrusions to total detections, which can be modeled by,

$$\mathrm{A} = \frac{Q+q}{Q+q+R+r} \tag{8}$$

Here, Q and R denotes true positive and false positive whereas q and r depicts false positive and false negative.

5.4.2 TPR

TPR [10] refers to a percentage of the intrusions that are actually intrusions and accurately detected as intrusions that are formulated as,

$$\mathrm{P} = \frac{Q}{Q+r} \tag{9}$$

5.4.3 TNR

TNR [10] mentions a percentage of packets that are non-intrusions and accurately identified as non-intrusions. The formula to compute TNR is given by,

$$\mathrm{N} = \frac{q}{q+R} \tag{10}$$

5.5 Performance Analysis

The evaluation of MCB distance-RideNN based upon considered metrics with various epochs by varying training data is demonstrated in Fig. 5. This part presents performance values of MCB distance-RideNN with 20, 40, 60, 80, and 100 epochs while training data is 90%. Fig. 5 a) reveals an assessment of MCB distance-RideNN to accuracy. MCB distance-RideNN attained 83.180%, 85.237%, 87.764%, 89.072%, and 91.279% of accuracy. An estimation of MCB distance-RideNN relating to TNR is signified in Fig. 5 b). TNR acquired by MCB distance-RideNN is 84.856%, 86.632%, 88.967%, 90.469%, and 92.628%. Fig. 5 c) presents an assessment of MCB distance-RideNN about TPR. TPR value acquired by MCB distance-RideNN is 82.572%, 84.271%, 86.082%, 88.380%, and 90.493%.

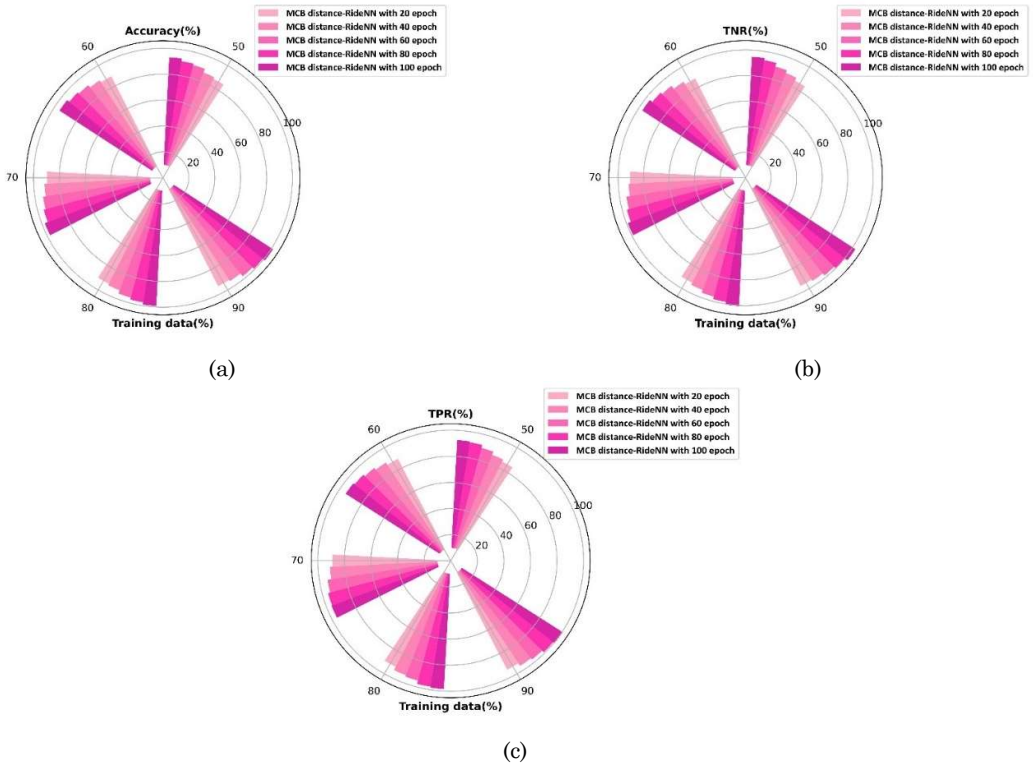


Fig. 5. *Performance evaluation of MCB distance-RideNN, a) Accuracy, b), TNR, c) TPR*

5.6 Comparative Techniques

DL-IDS [1], DIS-IoT [2], Deep Neural Network [3], and NSBPSO-DCNN [4] are the considered existing models to compare with MCB distance-RideNN for revealing its effectiveness.

5.7 Comparative Analysis

Fig. 6 represents a comparative analysis of MCB distance-RideNN concerning evaluation measures by changing training data. Herein, values obtained by MCB distance-RideNN and existing techniques are manifested for training data=90%. Fig. 6 a) illustrates an estimation of MCB distance-RideNN referring to accuracy. MCB distance-RideNN obtained 91.185% of accuracy whereas DL-IDS, DIS-IoT, Deep Neural Network, and NSBPSO-DCNN attained 83.664%, 84.282%, 86.295%, and 88.973%. This describes the performance enhancement of MCB distance-RideNN by 8.248%, 7.570%, 5.363%, and 2.426%. An evaluation of MCB distance-RideNN in respective of TNR is delineated in Fig. 6 b). DL-IDS, DIS-IoT, Deep Neural Network, and NSBPSO-DCNN acquired TNR of 84.710%, 86.517%, 88.597%, and 90.206% whereas MCB distance-RideNN obtained 92.574%. It interprets enhancing performance by 8.494%, 6.543%, 4.296%, and 2.558%. Fig. 6 c) exposes an estimation of MCB distance-RideNN with respective TPR. TPR achieved by MCB distance-RideNN is 90.412% whereas DL-IDS, DIS-IoT, Deep Neural Network, and NSBPSO-DCNN attained 83.658%, 85.312%, 86.319%, and 88.382%. This explains the performance improvement of MCB distance-RideNN by 7.470%, 5.640%, 4.527%, and 2.245%.

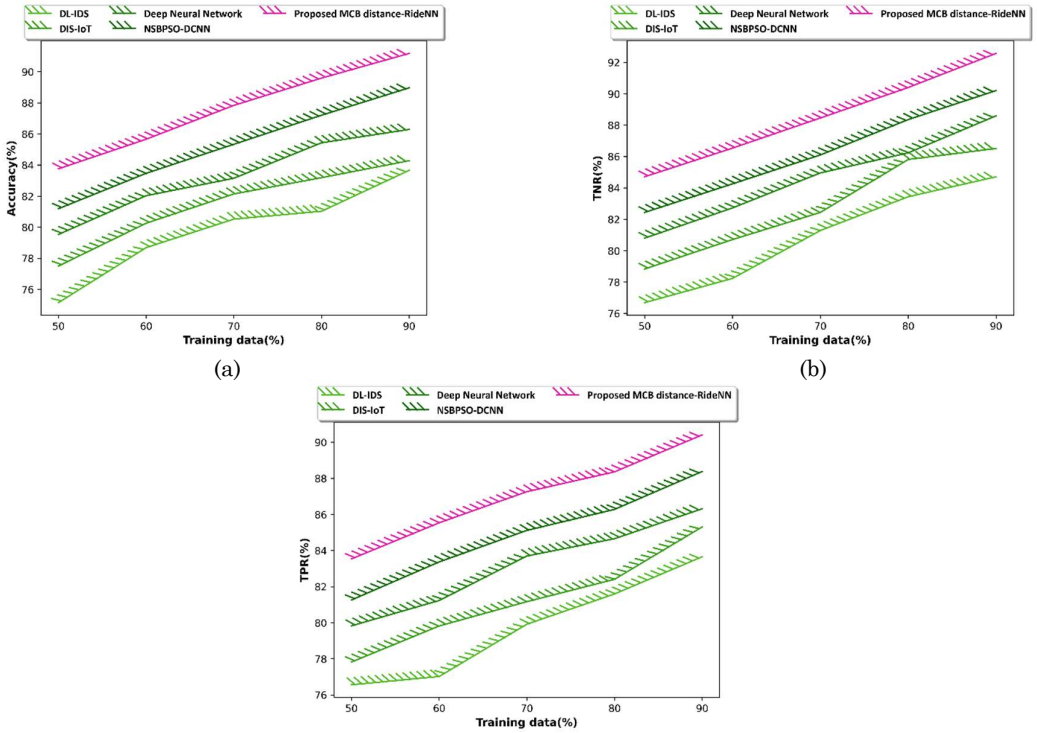


Fig. 6. *Comparative analysis of MCB distance-RideNN, a) Accuracy, b), TNR, c) TPR*

5.8 Comparative Discussion

Table 1 presents a comparative discussion of values acquired by MCB distance-RideNN as well as other approaches like DL-IDS, DIS-IoT, Deep Neural Network, and NSBPSO-DCNN. When training data=90%, an accuracy obtained by DL-IDS, DIS-IoT, Deep Neural Network, and NSBPSO-DCNN is 83.664%, 84.282%, 86.295%, and 88.973% whereas MCB distance-RideNN achieved 91.185%. The maximum accuracy signifies that MCB distance-RideNN can effectively scale to diverse IoT environments and adapt to changing network conditions. For considered training data as 90%, MCB distance-RideNN acquired a TNR of 92.574% whereas DL-IDS, DIS-IoT, Deep Neural Network, and NSBPSO-DCNN attained TNR of 84.710%, 86.517%, 88.597% and 90.206%. A maximum TNR reveals that MCB distance-RideNN efficiently identified benign activities and decreased an overhead related to managing false positive alerts. MCB distance-RideNN obtained a TPR of 90.412% while training data is 90% whereas DL-IDS, DIS-IoT, Deep Neural Network, and NSBPSO-DCNN acquired 83.658%, 85.312%, 86.319%, and 88.382%. The high TPR illustrates that MCB distance-RideNN ensured the effective detection of known attacks. It can be recognized that MCB distance-RideNN is an effective IDS as it achieved high accuracy, TNR, and TPR of 91.185%, 92.574%, and 90.412% for 90% of training data.

Table 1. *Comparative discussion of MCB distance-RideNN*

Setups	Metrics/ Methods	DL-IDS	DIS-IoT	Deep Neural Network	NSBPSO-DCNN	Proposed MCB distance-RideNN
Training data=50%	*Accuracy (%)*	75.152	77.505	79.539	81.195	83.761
	TNR (%)	76.693	78.836	80.803	82.442	84.714
	TPR (%)	76.565	77.827	79.832	81.264	83.541
Training data=90%	*Accuracy (%)*	83.664	84.282	86.295	88.973	**91.185**
	TNR (%)	84.710	86.517	88.597	90.206	**92.574**
	TPR (%)	83.658	85.312	86.319	88.382	**90.412**

6. Conclusion

IoT is an effective solution to access and connect all devices through the internet. However, it faces critical security issues and is prone to diverse attacks. IDS is utilized to detect the attacks when network security is intruded. In this research, MCB distance-RideNN is newly designed for intrusion detection in IoT. The simulation of IoT is initially accomplished and then, communication is performed with individual IoT

nodes. After that, data is stored in the form of features on the server. Next, the intrusion detection process is done by considering, input log file data from the NSL-KDD dataset, and next, data normalization is done employing Min-Max normalization. After that, feature selection is accomplished based on MCB distance. However, MCB distance is newly formed by combining Mahalanobis distance and City Block distance. Lastly, intrusion detection is accomplished by utilizing RideNN. Furthermore, MCB distance-RideNN acquired an accuracy of 91.185%, TNR of 92.574%, and TPR of 90.412% for 90% of the training data while the training data is 90%. In the future, feature selection methods will be designed employing metaheuristics algorithms for enhancing the performance of IDS.

Compliance With Ethical Standards

Conflicts of interest: Authors declared that they have no conflict of interest.

Human participants: The conducted research follows ethical standards and the authors ensured that they have not conducted any studies with human participants or animals.

References

[1] Y. Otoum, D. Liu, and A. Nayak, "DL-IDS: a deep learning–based intrusion detection framework for securing IoT", Transactions on Emerging Telecommunications Technologies, vol.33, no.3, pp. e3803, 2022.
[2] R. Lazzarini, H. Tianfield, and V. Charissis, "A stacking ensemble of deep learning models for IoT intrusion detection", Knowledge-Based Systems, vol.279, pp.110941, 2023.
[3] A. Awajan, "A novel deep learning-based intrusion detection system for IOT networks", Computers, vol.12, no.2, pp.34, 2023.
[4] S. Baniasadi, O. Rostami, D. Martín, and M. Kaveh, "A novel deep supervised learning-based approach for intrusion detection in IoT systems", Sensors, vol.22, no.12, pp.4459, 2022.
[5] D. Binu, and B. S. Kariyappa, "RideNN: A new rider optimization algorithm-based neural network for fault diagnosis in analog circuits", IEEE Transactions on Instrumentation and Measurement, vol.68, no.1, pp.2-26, 2018.
[6] A. K. S. U. Gokhan, C. O. Guzeller, and M. T. Eser, "The effect of the normalization method used in different sample sizes on the success of artificial neural network model", International journal of assessment tools in education, vol.6, no.2, pp.170-192, 2019.
[7] K. Zhang, W. Huang, B. Zhang, J. Xu, and X. Yang, "Robust Outlier Detection Method Based on Local Entropy and Global Density", arXiv preprint arXiv:2310.14960, 2023.
[8] S. H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions", City, vol.1, no.2, pp.1, 2007.
[9] H. Henderi, T. Wahyuningsih, and E. Rahwanto, "Comparison of Min-Max normalization and Z-Score Normalization in the K-nearest neighbor (kNN) Algorithm to Test the Accuracy of Types of Breast Cancer", International Journal of Informatics and Information Systems, vol.4, no.1, pp.13-20, 2021.
[10] M. A. Albahar, "Recurrent Neural Network Model Based on a New Regularization Technique for Real-Time Intrusion Detection in SDN Environments", Security and Communication Networks, vol.2019, no.1, pp.8939041, 2019.
[11] M. Zhong, Y. Zhou, and G. Chen, "Sequential model based intrusion detection system for IoT servers using deep learning methods", Sensors, vol.21, no.4, pp.1113, 2021.
[12] N. Abbas, M. Asim, N. Tariq, T. Baker, and S. Abbas, "A mechanism for securing IoT-enabled applications at the fog layer", Journal of Sensor and Actuator Networks, vol.8, no.1, pp.16, 2019.
[13] A. Khraisat, and A. Alazab, "A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges", Cybersecurity, vol.4, pp.1-27, 2021.
[14] T. Saranya, S. Sridevi, C. Deisy, T. D. Chung, and M. A. Khan, "Performance analysis of machine learning algorithms in intrusion detection system: A review", Procedia Computer Science, vol.171, pp.1251-1260,2020.
[15] NSL KDD dataset is taken from "https://www.unb.ca/cic/datasets/nsl.html", accessed on October 2024.