



# Ensuring Intrusion Detection for IoT Services Through an Improved CNN

Sunday Adeola Ajagbe<sup>1</sup> · Joseph Bamidele Awotunde<sup>2</sup> · Hector Florez<sup>3</sup>

Received: 21 September 2023 / Accepted: 21 October 2023  
© The Author(s), under exclusive licence to Springer Nature Singapore Pte Ltd 2023

## Abstract

Internet of Things (IoT) devices are challenging to manage information security due to some factors such as processing capability, exponential growth in homes, and their low energy consumption which raises the risk of cyberattacks. One way to avoid cyberattacks is using an intrusion detection system that is able to recognize assaults while warning users so that appropriate countermeasures can be taken. Several deep learning and machine learning techniques have been used in the past to try to detect new assaults; however, these attempts have not been successful. In order to optimize IoT devices, in this study, we make a classification of network assaults using the convolutional neural network models mCNN and CNN. This study aims to assess the application of deep learning intrusion detection systems for IoT devices. The NF-UNSW-NB15-v2 dataset was used in this experiment to train the neural network. The network stream's data were transformed into RGB images, which the neural network was trained on. The mCNN model outperformed the CNN model when compared to the proposed one for classifying network attacks. In addition, both networks perform better in most categories, with the exception of network attack detection, where the CNN performed worse than the suggested mCNN model.

**Keywords** Convolutional neural network (CNN) · Internet of Things (IoT) · Intrusion detection system (IDS) · Machine learning (ML) · Deep learning (DL)

## Introduction

The Internet of Things (IoT), which is a topic of computer science and engineering, is a trend in industry and academic sectors that has grown in popularity in the last years [5]. The term Internet of Things refers to a new paradigm of communication that describes devices with actuators and sensors

that are able to perceive their environment, interact with another device, and exchange information over the Internet [14, 23]. The number of IoT devices currently linked to the Internet is estimated to be around 50 billion, and in the years to come, this number is likely to increase dramatically [9]. The enormous amount of data produced by linked devices can be used by many applications. A few examples of the many application scenarios for the IoT include those in the areas of food and agriculture, demography, and augmented learning. For instance, it is estimated that more than 15 billion devices with IoT technology will be used for agriculture by 2025 [11].

Anomaly combating this danger requires the use of intrusion detection systems (IDS), which are crucial elements of intrusion detection in IoT systems. The great majority of currently available IDSs create detection models using conventional machine learning (ML) methods [8], which are further discussed below. The development of IDS models typically makes use of ML techniques because they are so powerful. Traditional ML techniques, which demand complex feature engineering, would require considerable research advances to extract required features from the information due to the

---

This article is part of the topical collection “Emerging Technologies in Applied Informatics” guest edited by Hector Florez and Marcelo Leon.

---

✉ Sunday Adeola Ajagbe  
sunday.ajagbe@tech-u.edu.ng

Joseph Bamidele Awotunde  
awotunde.jb@unilorin.edu.ng

Hector Florez  
haflorezf@udistrital.edu.co

- <sup>1</sup> First Technical University Ibadan, Ibadan, Nigeria
- <sup>2</sup> University of Ilorin, Ilorin, Nigeria
- <sup>3</sup> Universidad Distrital Francisco Jose de Caldas, Bogotá, Colombia

vast volume of unstructured data created by IoT devices. As a result, implementing standard ML-based solutions still presents a number of difficulties. Deep learning (DL) approaches have gained popularity in IDSs during the past few years, especially in the military industry.

Since it is simpler to extract information from DL than from conventional learning, it is often thought to be more reliable and accurate [6]. Deep learning is employed because it is simple to use and more accurate in extracting data. As a result, some research has focused on the use of DL techniques in domains like malware and anomaly detection, with mixed results. Deep learning approaches have been used in anomaly and malware detection to solve issues from technical and regulatory viewpoints, respectively. The majority of IDS systems have been created by employing, among other things, self-organizing mobile networks, wireless sensor networks, and already-existing computer networks [4]. There has not been much research, especially looking at DL methods in the IoT arena [7]. Thus, the main objective of this study is to explore the best and most efficient ways to deploy decentralized approaches in IoT and DNN contexts. This study provides a thorough, concentrated, and high-quality analysis that uses DL approaches to find reliable anomaly-based IDS.

The rest of the paper is structured as follows. The next section comprises the “[Related Work](#)”. “[Materials and Methods](#)” for IDS are followed in the third section. The fourth section of the study presents the “[Results](#)”, while the fifth section offers the “[Discussion](#)”. The last section contains the “[Conclusion](#)” as well as the future scope of the study area.

## Related Work

Security is a vital and crucial part of the information business that safeguards its hardware and software systems from external threats. Data security is becoming increasingly crucial to a company’s success as the volume of fresh data collected on the Internet rises. In this section, our review focuses on four topics: IoT security, intrusion detection factors, conventional techniques for detection, and techniques for sequential detection.

## IoT Security

There is no centralized system core architecture for IoT systems [23]. In contrast to conventional centralized system core architectures, IoT systems produced an ecosystem of interconnected gadgets and smart cities [16]. It is currently a crucial component of the infrastructure for smart cities. In the future, anyone with an internet connection will be able to access vast volumes of private and sensitive data by leveraging embedded device networks or other wireless

techniques [4]. Users, service providers, and enterprises can all benefit from the IoT, but there are also several disadvantages. Unlike a standard network system, an embedded system is able to communicate using a protocol that might vary depending on the application and device. There is not a centralized, standardized architecture in place yet for creating security measures. As the volume of data increases, this raises security issues related to IoT networks [3]. Malicious programs, trojan horses, hackers, and viruses are typically used in network attacks and odd behavior. These are the most pervasive and harmful objects. IoT networks are seriously threatened by device manipulation, device and identity robbery, and eavesdropping [17, 25]. In summary, the following are the networking technologies to link hardware devices in the home automation system: direct cable connection, phone line, Bluetooth, radio network, AC network, Wi-Fi online network, and Ethernet. An IoT-based network was established using wireless sensors as the infrastructure for the smart home system.

The integrity of data is impacted when the IoT is hacked, as well as people’s lives. IoT security systems can be compromised by hackers allowing them to get access to private workspaces like IT infrastructure. In addition, IoT weaknesses enable hackers to gain access to sensitive systems and data. Before transmission and exchange, an IoT device must authenticate and encrypt data to guard against illegal access. Even if data are stolen, encryption keeps hackers from accessing it. The autonomous control trap is avoided by a machine that cannot be controlled [2]. This tactic can be applied in a variety of circumstances. For instance, for traffic recognition, a variety of ML techniques have been proposed. Long-established traffic recognition and classification algorithms include decision trees (DT) and Random Forests (RF) [1]. Restricted Boltzmann Machine (RBM) and Support Vector Machine (SVM) were employed to detect as well as to recognize the traffic of a network [27]. Nevertheless, ML has made it possible to create more complex techniques for analyzing security concerns. In addition, Intrusion Detection Factors (IDS) were developed using hybrid techniques, where the total procedure is broken down into three sections, and employs a variety of DL techniques [6]. The technique has been used to find a variety of malicious network traffic anomalies. The method processes network data calculations using a range of statistical techniques. This is accomplished by watching how the neutral points in the system interact. A number of network security factors must be analyzed in order to find network security anomalies.

## Intrusion Detection Factors

An IDS must first discriminate between regular data and system abnormalities before it can attempt to detect a system intrusion. As a result, the IDS should recognize the

features of malicious data. For both normal and abnormal data, classification systems need to separate the two information sets. The technology automatically determines the distance between nodes using a network-specific code [12]. The application of this technique is based on the following key premise: data integrity and normalcy, which are estimated using the consistency distance between two nodes [10]. Consequently, a wide distance between nodes is a warning sign that the data may be unusual. The distance between two nodes in a network is determined using the Manhattan distance. In addition, in unsupervised deep learning, the output of the feature extraction algorithms is crucial since it determines the final product of the learning process. As a result, the technique ought to safeguard typical data while enabling ongoing defense. Adding traffic and carrying out information operations are examples of network manipulation [4].

### Conventional Techniques for Detection

ML may be useful for identifying malicious anomalies in a network, so it is very helpful for identifying hackers. Furthermore, Automatic Dimensionality Reduction (ADR) can be described as a video streaming approach that minimizes dimensionality by making use of both the encoder and the decoder. Input, output, and hidden layers make up the final three layers. In addition, autoencoders employ deployment, fine-tuning, and pre-training. Professionals work in a controlled environment to complete all of this. Deep belief networks (DBNs) are also important since they are regarded as conventional DL techniques that integrate unsupervised RBM and supervised backpropagation network layers. DBNs integrate supervised and unsupervised backpropagation network layers [23]. Two DBN strategies are supervised RBM processing and unsupervised RBM processing. In order to construct a hybrid anomaly detection system, these two methodologies were integrated. By employing autoencoder techniques to separate vectors, this strategy reduces their dimensionality. The DL systems categorize the data that DBN systems collect. In conclusion, hybrid systems are more accurate at detecting and have lower time-related complexity. Despite the fact that approximately one-eighth of all network traffic cannot be identified.

### Techniques for Sequential Detection

Network traffic and program code are examples of sequential data that predominate in the field of cyberspace security. System routine sequence data are the most accurate and useful data to have since it shows how a program interacts with the system kernel. It is simple to get real-time traces of the system routine for data acquisition. In addition, the order of the system procedures matches the language of the system. The system routine and its order are contrasted with the

words and sentences of spoken language. If the system under study is out of synchronization using these data, the backend of the sequential model models the normal language sequence of the system in order to forecast anomalies using an ensemble of thresholding classifiers. The probability distribution of a system sequence might be calculated using the language model. A one-hot encoding form is used to feed the input into the model [21]. The Backpropagation Through Time (BPTT) algorithm delivers standard samples during the training phase. Neural networks (NN) and  $k$ -means models serve as the backend classifiers. Comparatively, a web shell is harder to find than a backdoor.

## Materials and Methods

This section explains how the NF-UNSW-NB15-v2 dataset,<sup>1</sup> which was reported in the study, was utilized to create DL techniques that could detect, recognize, and categorize cyberattacks. We go into great detail about our categorization research and the NN techniques we tested. We also go into greater detail on classification experiments.

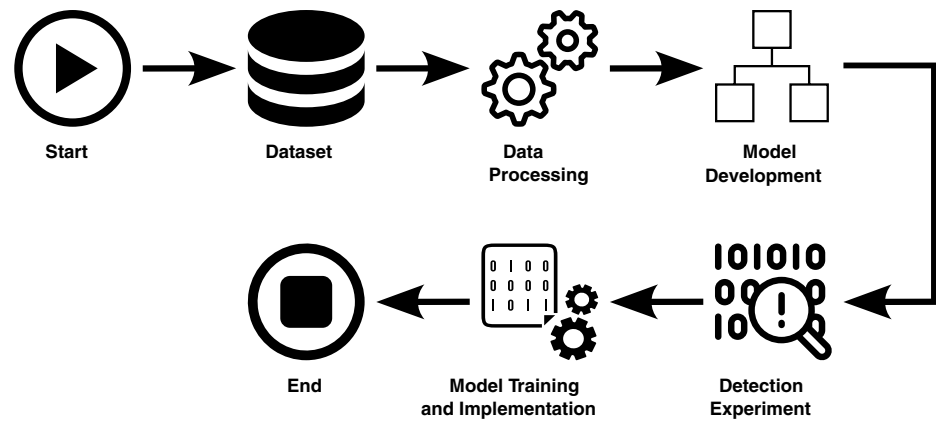
### Dataset

The NF-UNSW-NB15-v2 dataset was released in 2015 by the Cyber Range Lab at the Australian Centre for Cyber Security (ACCS). It is one of the most widely dataset used for Network Intrusion Detection System (NIDS) that is available according to experts at the ACCS. By comparing the Argus and Bro-IDS datasets and extracting the 35 most adequate characteristics from each, 12 additional features were produced [19]. In addition, we included network traffic produced by IXIA PerfectStorm tools and recorded it as pcap files. We constructed a dataset dubbed the NF-UNSW-NB15-v2 using 43 NetFlow characteristics that were obtained from the nProbe pcap file as criteria and comparing it to the UNSW-NB15 dataset.<sup>2</sup> A recently developed dataset called NF-UNSW-NB15-v2 is based on the UNSW-NB15 dataset but contains additional data. The results showed that adopting this updated dataset led to considerable gains in multi-class classification and decreased prediction times, which led to the selection of this dataset [22]. Figure 1 shows the framework for developing an improved CNN approach for the intrusion detection system for IoT devices. The study accessed the dataset then preprocessed it. The CNN model was developed for an instruction detection experiment, the model was trained and evaluated.

<sup>1</sup> <https://rdm.uq.edu.au/files/8c6e2a00-ef9c-11ed-827d-e762de186848>.

<sup>2</sup> <https://research.unsw.edu.au/projects/unswnb15-dataset>.

**Fig. 1** Framework for developing an improved CNN approach for the intrusion detection system for IoT devices



## Data Preprocessing

The next stage following data collection is preprocessing before using the preprocessed data to train our model. A four-step approach involves cleaning the data, converting the data, dividing the data into training and testing sets, and producing visual representations. Then, we used a CSV file with 43 NetFlow routines, each with a malicious label and an attack category. The values that cannot be converted to integers or floating points were removed. In order to lessen the bias in model training, we eliminated six attributes from each dataset entry that are not helpful for classifying network attacks. Feature engineering was carried out to preprocess or extract particular attributes from the raw data before data was entered into a CNN. Resizing, cropping, color enhancement, and other domain-specific preprocessing methods fall under this category. There are not any port numbers, IPv4 source or destination addresses, or minimum or maximum traffic TTLs. Following cleansing, we have a dataset with 9 classes and 37 features. We just utilized 40% of the dataset for speed. The attacks were distributed equally thanks to the hierarchy split. We have to divide the dataset into training (70%), testing (15%), and validation (15%) groups after the image conversion. Each assault in every dataset has a hierarchical representation. There were 560,927 samples utilized for training, validation, and testing. Since high variance is less of a problem for large datasets than for small datasets, a 3-way holdout is typically used for training. In this study, the proposed approach in Nguyen et al. [20] was followed because we use a non-image-based dataset. The min-max scaling method was used to normalize our data (see Eq. 1). The array is the proper size thanks to padding. The result of multiplying 255 by a color map is an 8-bit integer. Denial-of-Service (DoS) attacks and malware are improved by color maps in CNN graphics [23]:

$$x' = \frac{x - \text{Min}(x)}{\text{Max}(x) - \text{Min}(x)} \quad (1)$$

## Model Development

### mCNN

In the CNN framework, there are three convolutional modules which are followed by two completely connected layers, two fully connected layers, and a softmax activation layer, which slightly alters the fundamental CNN structure [13, 15, 23]. Similar smaller models can be used to classify different types of malware as well as IoT and Android malware [26]. This network will be utilized for comparison; however, it differs significantly from the fundamental CNN structure. This structure is made up of a  $3 \times 3$  convolutional layer, a normalization layer, and a max-pooling layer ( $2 \times 2$ ). The entire structure of mCNN is presented in Fig. 2 where the first two convolutional modules tend to widen as the distance between them increases from 3 to 16 and then to 32.

### CNN

The network topology is composed of convolutional modules (blocks) and groups rather than the conventional  $3 \times 3$  convolution. Each block in Fig. 3 corresponds to a hierarchical block organization that includes a batch normalization, an activator for the ReLU, and a jump link to the block above it [23]. This block organization provides a more thorough illustration of the network structure. To control the width of the network, we use the initial 33 convolutions and the initial block of every group of  $n$  convolution blocks. The width factor  $k$  is computed using the method suggested in Sergey et al. [24]. The hierarchy's topmost fully linked layer performs the final categorization. The CNN network was trained using the parameters  $n = 3$  and  $k = 0.2$ .

## Model Training and Implementation Environment

We implemented a variety of distinct methods detailed in this section, for training classifiers on unbalanced datasets. We also cover the testing tools and procedures for choosing

Fig. 2 mCNN structure

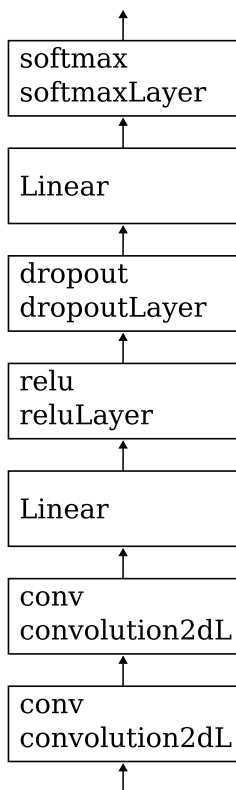
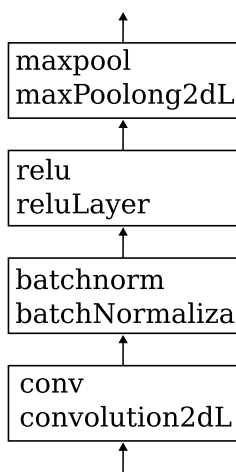


Fig. 3 CNN structure



hyperparameters. In addition, we explain how we chose the hyperparameters that were tested. We used hardware rented by vast.ai,<sup>3</sup> which is an online provider, which distributes instances based on peer-to-peer hardware. Based on this hardware, we trained four NN simultaneously without encountering any lag.

<sup>3</sup> <https://www.vast.ai/>.

Table 1 Model training with the best validation loss

Model	Validation loss
mCNN	1.44
mCNN sampler	1.46
mCNN CELoss	1.44
mCNN CBLoss	0.57
CNN	0.023
CNN sampler	0.067
CNN CELoss	0.051
CNN CBLoss	0.019

Table 2 Model training with average time

Model	Time (s)
mCNN (CELoss/CBLoss)	122
mCNN sampler	138
CNN (CELoss/CBLoss)	234
CNN sampler	243

### Detection Experiment

In our investigations, we trained and assessed two different CNN models to perform multiclass classification. Each model has 4 possible configurations using the NF-UNSW-NB15-v2 dataset. After selecting the hyperparameters, we trained each model for 100 epochs and stored the parameters as the validation loss decreased. Later on, we made tests using the model applying the lowest validation loss. During the training phase, the Adam optimizer, which is one of the most popular NN training optimizers, was used. The iteration of 1000 was set for each epoch to achieve better model performance in the setting of the dataset used. We also applied Loshchilov and Hutter’s decoupled weight decay regularization, which has been demonstrated to enhance Adam’s generalization performance with a factor of 0.0005 [18].

### Results

Datasets for training and validation were pre-segmented over 100 epochs. The loss of training and validation sets, the validation accuracy, and the average training time are the results of NN training and validation. The correlation between training and validation loss can be utilized to detect if the network is underfitted or overfitted. The network is overfitted when the validation loss is greater than the training loss; otherwise, the network is underfitted. A deeper look reveals that every trained model has a rapid reduction in loss after epoch number 20. CBLoss has a lower loss than the other models since it divides

**Table 3** Average time per Epoch in model training

Class	Accuracy (%)	Precision	Recall	F <sub>1</sub> score
Analysis	99.86	0.87	0.95	0.18
Backdoor	98.93	0.07	0.68	0.21
Benign	97.52	1.02	0.98	0.99
DoS	99.73	0.22	0.39	0.31
Exploits	97.57	0.32	0.70	0.44
Fuzzers	99.67	0.72	0.89	0.79
Generic	99.77	0.33	0.72	0.45
Reconnaissance	99.88	0.71	0.98	0.83
Shellcode	99.89	0.18	0.65	0.26
Worms	99.98	0.21	1.03	0.34

attacks into fewer categories. Table 1 presents the validation loss for CNN and mCNN models.

CNN uses more convolutional layers than mCNN, which increases the training time, as presented in Table 2.

Since we could only train four models at once and each training round required more than 10 h, it took approximately 42 h to train all eight models.

### Performance Evaluation Metrics for Detection

We chose those models that have the lowest validation loss. Then, those models were used to classify the test data in order to evaluate their classification capabilities. In order to select the best model for each NN, we also created generic metrics using the values in the matrix that evaluate the effectiveness of various anomaly detection techniques. In this context, accuracy corresponds to the percentage of predictions that are correct across all samples, while precision refers to the proportion of actual positives to all positives. The F<sub>1</sub> Score presented in Eq. 5, which is a relation between the accuracy (see Eq. 2), precision (see Eq. 3), and recall (see Eq. 4), is used to determine the percentage of accurate predictions across all relevant samples:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FN} + \text{TN}} \quad (2)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (3)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (4)$$

$$F_1 \text{ Score} = \frac{2 * (\text{Precision} * \text{Recall})}{\text{Precision} + \text{Recall}} \quad (5)$$

**Table 4** CNN multi-class classification results

Class	Accuracy (%)	Precision	Recall	F <sub>1</sub> score
Analysis	98.81	0.08	0.91	0.12
Backdoor	98.74	0.06	0.65	0.10
Benign	98.10	1.00	0.95	0.90
DoS	98.63	0.11	0.11	0.11
Exploits	98.26	0.76	0.52	0.64
Fuzzers	98.53	0.61	0.63	0.62
Generic	98.42	0.11	0.41	0.18
Reconnaissance	98.77	0.69	0.94	0.79
Shellcode	98.71	0.04	0.90	0.10
Worms	98.86	0.11	0.89	0.21

### Summary of the Results

As presented in Table 3, using a weighted random sampler produces different results for mCNN from those provided by mCNN [22]. Based on the analysis that obtained a max recall of 0.93, or 62% higher, all F<sub>1</sub> Scores were lower with the exception of one class.

The outcomes of CNN resampling using a weighted random sampler are presented in Table 4, which follows the format in Table 3 for each category. In addition to performing poorly on category generalizers and fuzzers, CNN demonstrated higher precision and recall differences than mCNN.

According to Sarhan et al. [22], the worst-classified categories were analysis, backdoor, and DoS. Using mCNN and resampling, we increased recall in each of the aforementioned classes. When it comes to categorizing vulnerability exploits, shellcodes [23], and generalizations, our methodology is inadequate.

### Discussion

Network assaults can be categorized by small CNNs. In contrast, as demonstrated in Tables 3 and 4, our models have better F<sub>1</sub> Scores. Our models were successful in classifying a variety of intrusion attempts. Since the suggested models are generalizable, they can effectively and accurately identify attacks. Instead of testing every conceivable parameter, we chose to concentrate on learning rates as well as batch sizes in order to overcome class imbalances. CNN should be modified to increase the number of groups and blocks and decrease the rate of dropout for each dropout layer. CNN's model training time, including fine-tuning the model and hyperparameters, is less than 7 h. This period of time was set aside to make sure the model worked. Our findings have improved as a result of better parameters. A larger dataset could not be used to analyze the final dataset since it was too big and uneven.

Learning the class is impossible for one or two examples in the smallest subset. We required a sizable dataset to obtain a decent representation of each class due to the dataset's imbalance. This, in our opinion, is the finest course of action. Since our model is constrained, experimental findings demonstrate that trained models perform better than those in the reference papers.

Class precision remains higher, even when employing a random sampler. The worst-performing classes can be given greater weight and resampling; thus, cost-sensitive learning can yield positive results. Each IoT device should receive externally trained cloud models since internal device training takes too long. The IoT device can pick up some fresh assault examples. It is best to train the IDS overnight so that it can be immediately updated when fresh attacks are found.

Model training and validation for CNN took longer than for mCNN. It took an average of 51 s to categorize a test set. This speed can provide an indefinite backlog when used as an IDS. Only three seconds separated the two networks' inference times. Although mCNN (3.79 MB) is smaller than CNN (3.05 MB), it is critical to notice that the models' trainable parameters vary widely. CNN and mCNN have trainable parameters apart from that.

Generalization is the primary objective of a classifier. Our models have greater generalizability and accuracy. Generalization could benefit from an improved sampler or loss function. In addition, generalization can be enhanced using input photos with more excellent width factors.

Resampling and cost-sensitive learning are useless due to the unbalanced dataset. By randomly choosing among classes, we may avoid hierarchical splitting for training and validation. The limited sample size for the worm class with the lowest population was one of the reasons we decided against it. To conserve or gain space in the final image, it is best to perform more in-depth analyses of the characteristics during preprocessing. The convolutional layer input quality ought to be improved as a result of lessening category preference, adding noise, or using other image-enhancing methods. Pads' modest size causes them to add too much information to the image. Testing padding is possible because it may impact the outcomes. It is important to consider the group count, block count, and expansion factor. It would be wonderful to include a few additional simple classification models. We would have wanted to utilize a scheduler for the learning rate instead of a set learning rate. Therefore, during training, we did not configure the learning rate scheduler. Finally, we can adjust the weights of the loss function and resample to improve performance.

## Conclusions

To categorize cyberattacks, we developed two CNN using the NF-UNSW-NB15-v2 dataset. This study intends to investigate the performance of CNNs created for IoT devices during cyberattacks and determine whether they may be utilized as anomaly-based IDS. Some studies reveal that the model can accurately categorize cyberattacks; in addition, the model might be used as IDS. In this research, we tested various approaches to enhance the accuracy of training different DL models on unbalanced datasets using resampling and cost-sensitive learning.

Further research is required to determine the ideal weights and the potential for integrating resampling with cost-sensitive learning using a hybrid strategy. In this method, unbalanced datasets would be handled. Unbalanced datasets could be addressed by utilizing various splitting techniques or additional oversampling techniques that add more variance to the data using a different preprocessing technique without padding to avoid mistakes. The CNN uses filters that pass over the image allowing it the creation of larger images from fewer amounts of input. Depending on how the image is put together, this may lead to additional relationships between features.

Finally, there is another opportunity to research on fine-tuning the hyperparameters in CNN models. Since the Adam optimizer is the only one employed in this study, it is possible to investigate the usage of additional optimizers for training in order to increase the suggested models' classification accuracy.

In the future, the development of more sophisticated CNN architectures designed specifically for IoT intrusion detection can be the subject of research. Designing networks that can effectively process heterogeneous data types from IoT devices, such as text, pictures, and sensor data, may be necessary to achieve this.

**Author Contributions** SAA: conceptualization, writhing of original draft, validation, methodology, and coding. JBA: project administration, methodology, resource management, editing, and supervision. HF: resource management, review, editing, and supervision.

**Funding** The authors declare that no fund was received for this study.

**Data Availability** Not applicable.

## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

**Ethics Approval** Not applicable.

**Informed Consent** Not applicable.

## References

- Adeniji OD, Adeyemi SO, Ajagbe SA. An improved bagging ensemble in predicting mental disorder using hybridized random forest - artificial neural network model. *Int J Comput Inform.* 2022;46(4):543–550. <https://doi.org/10.31449/inf.v46i4.3916>.
- Adhie RP, Hutama Y, Ahmar AS, Setiawan M, et al. Implementation cryptography data encryption standard (des) and triple data encryption standard (3DES) method in communication system based near field communication (NFC). *J Phys Conf Ser.* 2018;954: 012009.
- Adimoolam M, John A, Balamurugan N, Ananth Kumar T. Green ICT communication, networking and data processing. In: Balusamy B, Chilamkurti N, Kadry S, editors. *Green computing in smart cities: simulation and techniques*. Berlin: Springer; 2021. p. 95–124.
- Adly AS, Adly MS. Approaches based on artificial intelligence and the internet of intelligent things to prevent the spread of covid-19: scoping review. *J Med Internet Res.* 2020;22(8): e19104.
- Ajagbe SA, Adesina AO, Ilupeju OA, Thanh DN et al. Challenges and perceptions in the use of ICT in student assessments during the covid-19 pandemic. In: 2021 8th international conference on information technology, computer and electrical engineering (ICI-TACEE). IEEE; 2021. pp. 89–94.
- Ajagbe SA, Adigun MO. Deep learning techniques for detection and prediction of pandemic diseases: a systematic literature review. *Multimed Tools Appl.* 2023. <https://doi.org/10.1007/s11042-023-15805-z>.
- Al-Emran M, Malik S.I, Al-Kabi MN. A survey of internet of things (IOT) in education: opportunities and challenges. In: *Toward social internet of things (SIoT): enabling technologies, architectures and applications: emerging technologies for connected and smart social objects*. Springer, Berlin; 2020. pp. 197–209.
- Aljumah A. IOT-based intrusion detection system using convolution neural networks. *PeerJ Comput Sci.* 2021;7: e721.
- Awotunde JB, Ajagbe SA, Florez H. Internet of things with wearable devices and artificial intelligence for elderly uninterrupted healthcare monitoring systems. In: *International conference on applied informatics*. Springer, Berlin; 2022. pp. 278–291.
- Bansal SK. Towards a semantic extract-transform-load (ETL) framework for big data integration. In: 2014 IEEE international congress on big data. IEEE' 2014. pp. 522–529.
- Farooq MS, Riaz S, Abid A, Abid K, Naeem MA. A survey on the role of IOT in agriculture for the implementation of smart farming. *IEEE Access.* 2019;7:156237–71.
- Gaber T, Awotunde JB, Folorunso SO, Ajagbe SA, Eldesouky E, et al. Industrial internet of things intrusion detection method using machine learning and optimization techniques. *Wirel Commun Mob Comput.* 2023;2023:1–15.
- Hernandez J, Daza K, Florez H. Spiking neural network approach based on *Caenorhabditis elegans* worm for classification. *IAENG Int J Comput Sci.* 2022;49(4):1099–111.
- Hernandez J, Daza K, Florez H, Misra S. Dynamic interface and access model by dead token for IOT systems. In: *International conference on applied informatics*. Springer; 2019. pp. 485–498.
- Hernandez J, Florez H. An experimental comparison of algorithms for nodes clustering in a neural network of *Caenorhabditis elegans*. In: *21st international conference computational science and its applications*. Springer; 2021. pp. 327–339.
- Iyawa GE, Herselman M, Botha A. Digital health innovation ecosystems: from systematic literature review to conceptual framework. *Proc Comput Sci.* 2016;100:244–52.
- Kodali RK, Yerroju S. Energy efficient home automation using IOT. In: *2018 international conference on communication, computing and Internet of Things (IC3IoT)*. IEEE; 2018. pp. 151–154.
- Loshchilov I, Hutter F. Decoupled weight decay regularization. In: *7th international conference on learning representations*; 2019.
- Moustafa N, Slay J. Unsw-nb15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: *2015 military communications and information systems conference (MilCIS)*. IEEE; 2015. pp. 1–6.
- Nguyen SN, Nguyen VQ, Choi J, Kim K. Design and implementation of intrusion detection system using convolutional neural network for DOS detection. In: *Proceedings of the 2nd international conference on machine learning and soft computing*. 2018. pp. 34–38.
- Rawat R, Oki OA, Sankaran S, Florez H, Ajagbe SA. Techniques for predicting dark web events focused on the delivery of illicit products and ordered crime. *Int J Electr Comput Eng.* 2023;13(5):5354–65.
- Sarhan M, Layeghy S, Portmann M. Towards a standard feature set for network intrusion detection system datasets. *Mob Netw Appl.* 2022;27:357–70.
- Sayed N, Shoaib M, Ahmed W, Qasem S, Albarrak A, Saeed F. Augmenting IOT intrusion detection system performance using deep neural network. *Comput Mater Contin.* 2022;74(1):1351–74.
- Sergey Z, Komodakis N. Wide residual networks. In: *British Machine Vision Association*. 2016.
- Vinod P, Jaipur R, Laxmi V, Gaur M. Survey on malware detection methods. In: *Proceedings of the 3rd Hackers' workshop on computer and internet security (ITKHACK'09)*. 2009. pp. 74–79.
- Wang W, Zhao M, Wang J. Effective android malware detection with a hybrid model based on deep autoencoder and convolutional neural network. *J Ambient Intell Hum Comput.* 2019;10:3035–43.
- Yang J, Deng J, Li S, Hao Y. Improved traffic detection with support vector machine based on restricted Boltzmann machine. *Soft Comput.* 2017;21:3101–12.

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.