Received: 7 August 2023 | Revised: 13 October 2023 | Accepted: 30 October 2023 | Published online: 7 November 2023

RESEARCH ARTICLE

Journal of Data Science and Intelligent Systems
2024, Vol. 2(3) 174–180
DOI: 10.47852/bonviewJDSIS32021485

BON VIEW PUBLISHING

# Performance Metrics of an Intrusion Detection System Through Window-Based Deep Learning Models

**Fatima Isiaka**[1,*]

[1]*Department of Computer Science, Nasarawa State University, Nigeria*

**Abstract:** Intrusion and prevention technologies perform reliably in harsh conditions by fortifying many of the world's highest security sites with few defects in high performance. This paper aims to contribute by designing an intrusion/preventive system using a window-based convolutional neural network (CNN), an integrated recurrent neural network (RNN), and autoencoders (AutoE) to detect and test the performance of the intrusion detection system. The data packets were converted to images where the pixels were used as input. The CNN architecture shows a three-layer model with high predictive performance. The result shows high performance on CNN as compared to both RNN and AutoE; CNN seems to resist overfitting more than the rest of the models. The future perspective would be to test the model on other standard methods such as support vector machine (SVM) and dynamic control systems.

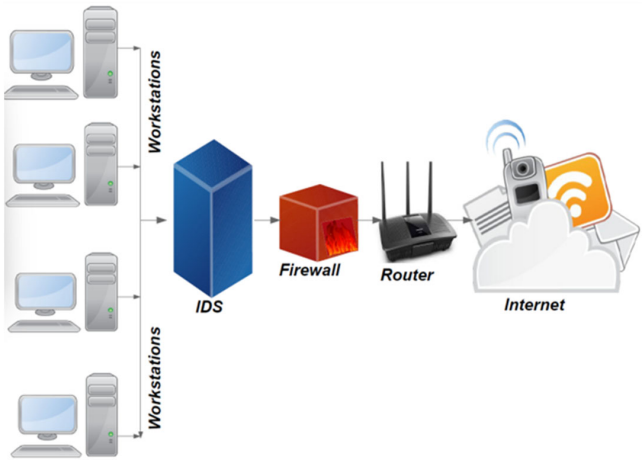
**Keywords:** intrusion and prevention system, convolutional neural network, recurrent neural network, autoencoders, performance metrics, data packets

## 1. Introduction

The CNN architecture shows a three-layer model with high predictive performance. The result shows high performance on CNN as compared to both RNN and AutoE; CNN seems to resist overfitting more than the rest of the models. The future perspective would be to test the model on other standard methods such as support vector machine (SVM) and dynamic control systems.

An intrusion detection system (IDS) is basically a device or a software application that is meant to monitor network and system activities and send alerts to system administrators at the right time [1–3]. It can monitor both inbound and outbound traffic activities to detect an intrusion. There are basically two kinds of IDS, which are the network intrusion detection system (NIDS) and the host intrusion detection system (HIDS). The NIDS can monitor all traffic that comes from and goes to all the devices of a network. If for instance any traffic matches that of the library or the known attacks or say any abnormal behavior is sensed a single alert is triggered and warnings are sent to the administrator. It is sometimes placed in a subnet close to the firewall (Figure 1) to detect if anyone is making an attempt to break the firewall. It is also capable of comparing signatures of similar packets of data and be able to detect harmful data packets that match and signature recorded on the database.

**Figure 1**
**An intrusion detection system placed close to a firewall to detect anomalies**




## 2. Literature Review

The HIDS monitors individual hosts or system devices in particular [1–6]. Inspections are conducted on inbound and outbound packets of the device and an alert is sent to the administrator on suspicious activities. It is capable of taking snapshots of system files at a given point in time and matching them to determine if any critical file has been modified or

*Corresponding author: Fatima Isiaka, Department of Computer Science, Nasarawa State University, Nigeria. Email: isiakafatima@nsuk.edu.ng

mistakenly deleted and send alert when required [7–12]. Building an IDS can be approached in two ways such as the standard method or the novel approach. The standard approach involves collected browsing and network activity saved for a certain number of days, which are then retrieved for analysis purposes.

The novel approach involves conducting an experimental study involving participants with certain given takes that involve all forms of intrusion techniques. The IDSs have in recent years been evenly and highly researched; the commonest changes occur in the dataset collected, and the dataset must contain many samples of intrusion techniques such as infiltration, DOS attack, and brute force from within a network space [13–17]. In recent years, as network behaviors and their patterns change and network intrusions evolve, it has become paramount to develop from basic static and single-handle datasets to datasets generated dynamically. This reflects the traffic compositions and online intrusions of a particular time, which can be reproducible, modified, and extensible. To be able to do this, training a deep learning model to identify anomalies from a given dataset is one of the initial stages of data preprocessing which we will be looking at in the preceding sections [18, 19]. Anomaly-based systems are the most effective IDSs, due to the application of machine learning algorithms embedded within the system. They have no need to search for a specific pattern of anomaly, but they are able to treat anything that does not mark the profile as "Anomalous" [20–25]. The source code for the generated program would not be shared in this book, but the basic steps will be illustrated, a GitHub link will also be provided, and the data preprocessing stages will be shared. This is to enlighten the basic idea and implementation process used to achieve successful predictive accuracy [23, 26–29].

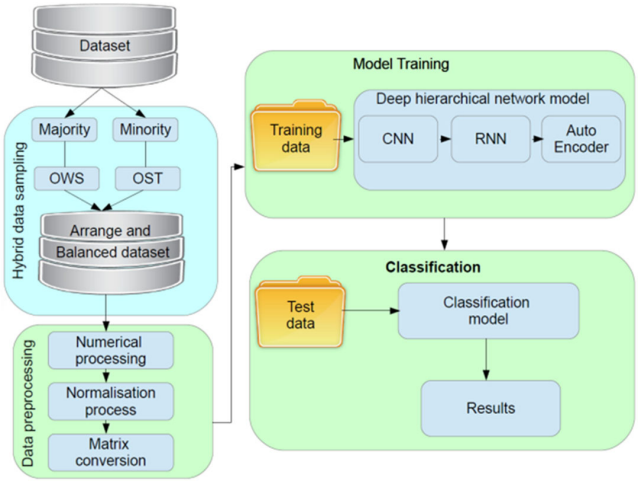
## 3. Method and Dataset

The dataset used here is based on the Canadian Institute for Security 2017 (CICS), which can be obtained online as a form of freeware. They are used around the world for security testing and malware prevention. The dataset was visualized using Matlab to determine how much data there are for each class of anomaly and normal contained in it. In order to solve the problem of imbalance, a one-way selection was used to reduce the major data samples and the over-sampling technique (Figure 2) to increase the minor samples. This is to set some balance to the dataset and build a training model. This will also reduce the training time of the model. The three main deep neural networks (convolutional neural network (CNN), recurrent neural network (RNN), and autoencoder (AutoE)) discussed in Chapter 1 are used for the training. It is important to mention that quite a few papers on IDSs using machine learning techniques have made use of the dataset. This is most likely due to its unavailability at that time. One of the first research works that made use of it was for developing a hybrid IDS. The environment they created uses Keras API to set up their system and a Microsoft CNTK for a back-end engine.

The dataset is based on cybersecurity modeling and used around the world by different universities, independent researchers, and private industry. Most of the anomaly contents are malware-inclined and obfuscated as malware that hides to avoid detection and extermination.

The data model that frames the characteristics of the dataset was created to represent as close to a real-life situation as possible using prevalent malware in a real-life scenario. The dataset consists of Trojan horses, spyware, and ransomware that provide a balanced

**Figure 2**
**The general network model for the proposed system**




dataset that can be used to test the obfuscated malware and a proposed detection system.

The modified dataset focuses on the simulation of real-world scenarios that indicate the breakdown of benign and malicious memory bins, and it is balanced with it being made up of 50% malicious memory bins and 50% benign memory bins. The total breakdown for malware families is shown in Table 1. The proposed IDS and prevention systems are based on important defence mechanisms as a tool against sophisticated and increasing attacks. Due to a lack of reliable test and validation datasets, the anomaly detection approaches are facing some defects from consistent and accurate performance evaluations. The evaluations from previous studies show that most of these types of systems are out of date and unreliable. They mostly suffer from a lack of traffic diversity and volumes, and some are unlikely to cover a variety of known attacks. The currently proposed method took these defects into account and integrated a standby default solely to report standby anomalies that would require instant traffic addressing and computational approaches for benign case

**Table 1**
**A breakdown for malware families for typical CICS dataset: Courtesy Google Archive**

| Malware category | Malware families | Count |
|---|---|---|
| Trojan horse | Zeus | 196 |
| | Emotet | 200 |
| | Refroso | 200 |
| | Scar | 157 |
| | Reconye | 195 |
| Spyware | 180Solutions | 200 |
| | Coolwebsearch | 200 |
| | Gator | 241 |
| | Transponder | 141 |
| | TIBS | 200 |
| Ransomware | Conti | 195 |
| | MAZE | 171 |
| | Pysa | 200 |
| | Ako | 220 |
| | Shade | 200 |

grouping through a strategized method using the classification algorithm described in Figure 2.

## 4. Result

In this section, we have particularly considered benign outliers to improve the stability and robustness of anomaly-based intrusion systems. There were interesting findings that were actually used and embedded into the malware samples that are supposed to be propagated as part of network packets collected, and a lot of the benign code and benign cases have different detection segments (Figure 3). The figure shows the index page of the CICS dataset that was strategized and classified some of the cases as benign with a view on the malware group (Figure 5). The benign cases are characteristically harmless or well-intentioned, the opposite of malicious cases which is naturally how it works. The "*pslist*" is categorized between 12 and 17 numerical values from the initial coding process. The interphase shows how the IDS can be used for major monitoring of networks for attack activities or policy violations. This can successfully identify anomalous deviations from the normal traffic behavior. The CNN is used by capturing the image representation of the data packets (Figure 4).

The CNN is applied to analyze visual imagery (Figure 4), which is a picture-listed image in the form of the data packets (CICS data) of Figure 3. The CNN here uses a mathematical operation mentioned in Chapter 1 called convolution in the place of general matrix multiplication in more than one input layer with multiple hidden layers. The process is specifically designed to feed in process flow as pixel data, which is used in image recognition and preprocessing for image segmentation and classification.

The spyware has 0% compared to ransomware (17%). Figure 3 has malicious behavior that aims to gather certain information about a person's activity from an organization and send it to another entity with malicious intent in a way that brings harm to the user. An imposter with malicious intent can violate privacy by endangering the device and its security. This kind of behavior presents certain unrecognized interests that can only be detected by intrusion detection like the ones presented here. Some websites may engage in certain spyware behaviors like web tracking, which are often

**Figure 3**
**Index view of the CICS dataset with strategized benign and malware groups**

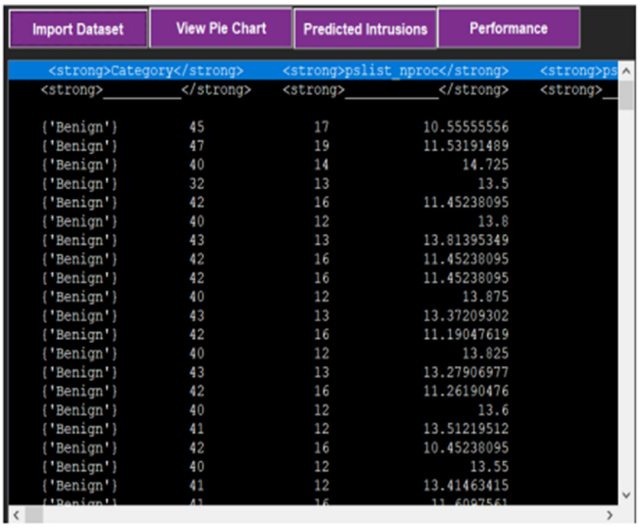
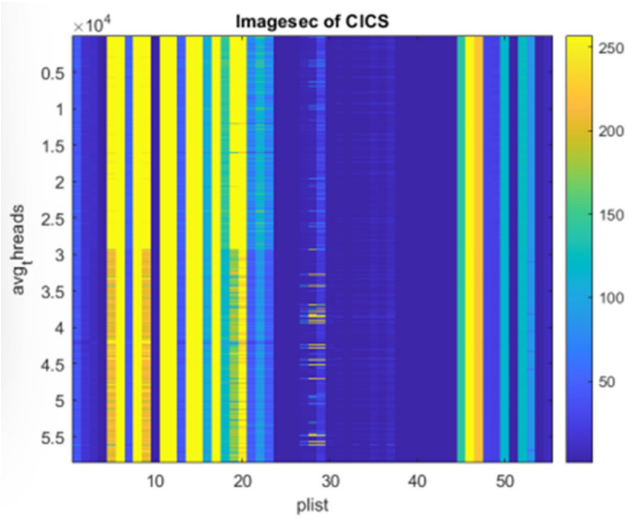


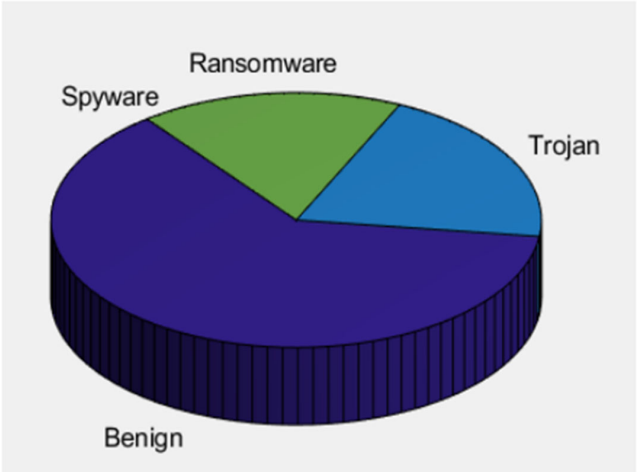

**Figure 4**
**Image representation of the CICS data**




hard to track and classified as malicious. The software presented in the IDS can also be affected and hence the integrated IDS with detection characteristics can be much faster with the maximum bandwidth that is required since the spyware is sometimes frequently associated with advertising that may cause some malicious cases to appear benign. They involve many of these issues because the behavior is most common and they can have non-harmful curses. Providing a precise definition of spyware is sometimes a difficult task, and the intrusion system can precisely differentiate benign cases from malicious cases.

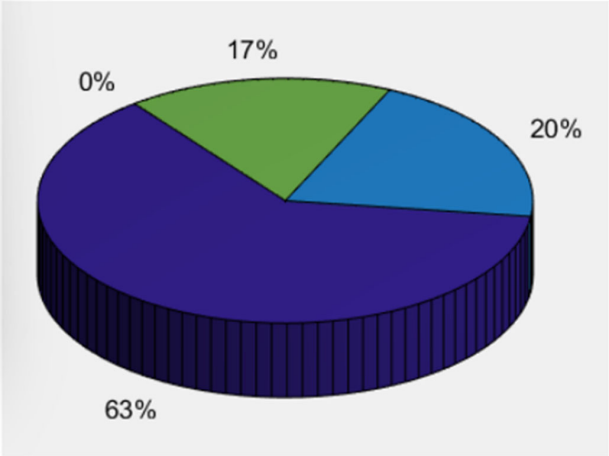
Ransomware (Figures 5 and 6) 17% is the type of malware from crypto virology that mostly threatens the publishing of a person's personal data and permanently blocks access to it until a ransom is paid off. Certain simple ransomware sometimes locks the system without damaging the files in them. Advanced malware uses a certain technique called cryptoviral extortion that encrypts the person's personal files by making them inaccessible and then demands a ransom, and the person of the system affected by ransomware is higher than spyware. The intrusion system is

**Figure 5**
**Malware classes compared to benign cases**



properly implemented with cryptoviral extortion attacks for the recovery of important files with the decryption key. This bypasses digital currencies, which are sometimes difficult on Paysafecard or Bitcoin and also other cryptocurrencies; tracing and prosecuting are also bypassed. The ransomware attacks are carried out using Trojans, which are disguised as legitimate files and have 20% (Figure 6) of harmful effects on the collected packets of data. Users can be tricked into downloading and opening these malicious files when they arrive as a form of email attachments. However, the high-profile intrusion system can travel system process automatically between software prevention firewalls with user interaction.

**Figure 6**
**Percentage of malware classes compared to benign cases for the CICS dataset**




The CNN on the batch process for the image-converted file uses 2D convolution in the normalization phase. The convolution tool separates and also identifies the different features (Figure 7) of the image for analysis in a process known as feature extraction. The entire network of features extracted consists of different pairs of convolutional and pooling layers (Figure 8). A fully connected layer uses the output from a convolution process and then predicts the class of the images based on the features extracted from the ones in Figure 7. The CNN model of the features extracted has the purpose of reducing the number of features contained in the original set of features. There are different CNN hidden layers as shown in Figure 8 architecture diagram.

Three types of layers make up the CNN, which was used as the convolutional layers and pooling layer for a fully-connected layer. The layers are stacked and the CNN architecture forms a chain. In addition to the design, the three layers have two more significant parameters that are both dropout layers and the activation function is also defined with it. The output is known as the feature map and gives information about the image of the data packets such as the corners and edges. The feature map is fed to the layers to learn the different features of the input image. All the layers in the CNN pass the result to the next layer once the convolutional operator in the input is applied. The convolutional layers benefit from the design and ensure the spatial relationship between the pixels is intact for the resulting outcome.

Sometimes running the CNN-based application on edge devices near the source data can cause a significant delay in latency and

**Figure 7**
**Descriptive representation of layers of CNN model from image dataset**

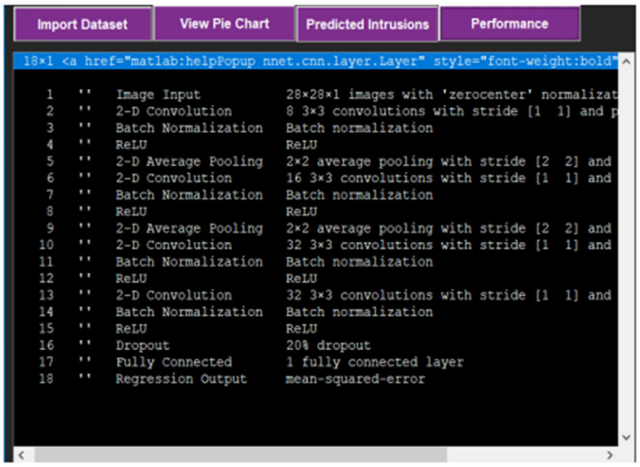
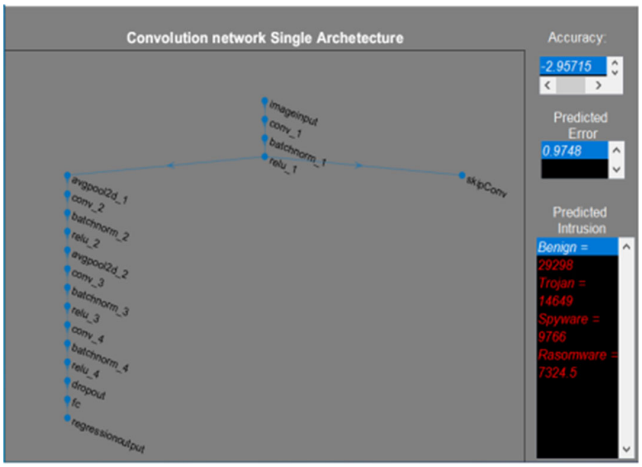



**Figure 8**
**CNN architecture with predicted intrusion**




privacy challenges, and hence the RNN and AutoE were used as a form of comparison to test the performance of the CNN model (Figures 10 and 11) using training and test set. However, in some events, due to their reduced computing resources, the energy constraints in some devices can hardly satisfy the CNN needs in processing the data storage. From the performance, CNN still performs higher in the two test samples of Figures 10 and 11. In this work, CNN has the best trade-off between accuracy and the execution time during respective literature on the hardware constraints tool and crucial parameters were used for the final runtime. The method explores the time needed for the training and tuning for the corresponding hyperparameters. The number of times to run the prediction models was compared on different platforms and the utilization of these three methods (CNN, RNN, and AutoE) highly facilitates design space during exploration of the parameters by proving the best CNN on a target GPU. The result shows the root mean square error provides a less than 5.53% average prediction error even for unexplored and unseen CNN models' architectures (Figure 7). RNN depicts comparable accuracy and also needs more effort and time to be trained. The

**Figure 9**
**RMSE add loss prediction of CNN model**

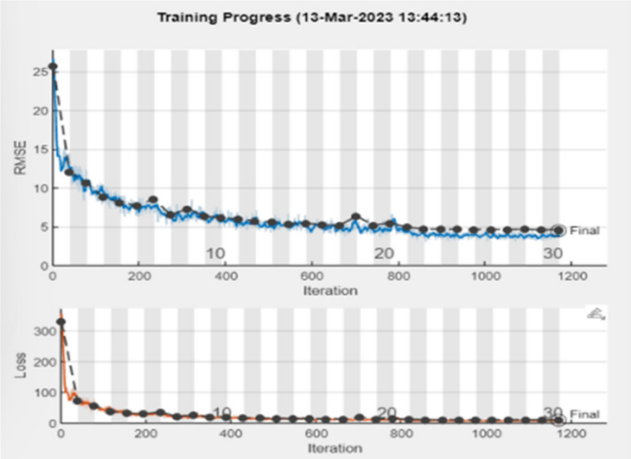



**Figure 10**
**Performance of models on first sample dataset**

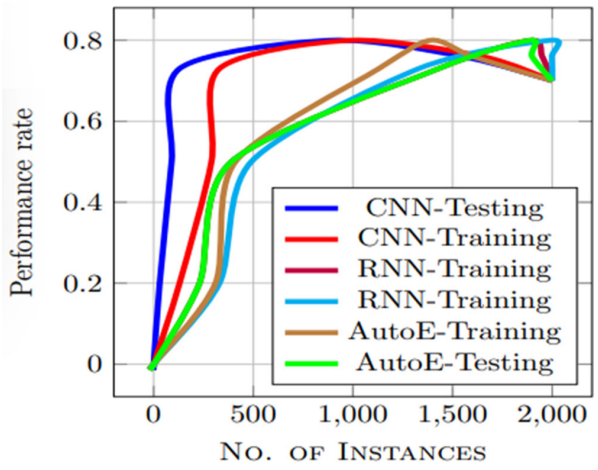
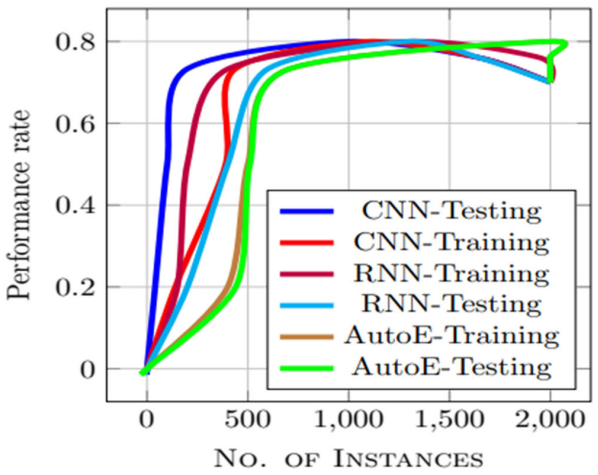



**Figure 11**
**Performance of models on second sample dataset**




other two approaches (loss and iteration) are less accurate for CNN performance estimation.

### 4.1. Handling overfitting and underfitting problems

The overfitting in the three models shows the model training data, the models seem to resist overfitting. This is a common problem in learning models, where a model performs well on the training data but poorly on the test data or unseen data (Figures 9 and 10). This means that the CNN model has memorized the specific patterns and noise in the training data, but it cannot adapt to new situations or variations.

Here, overfitting reduces the accuracy and reliability of the model, and it can lead to poor decisions and outcomes. To avoid overfitting, the data needed to be used during regularization techniques that limit the complexity of the model and introduce some randomness or noise to the learning process. Regularization is also a way of adding some constraints or penalties to the model so that it does not overfit the training data. There are also different types of regularization methods, but they all aim to reduce the variance of the model and increase its bias. Variance measures how sensitive the model is to small changes in the data, while bias measures how far the model is from the true relationship. A good model should have low variance and low bias, but there is usually a trade-off between them. Regularization helps find a balance between them by shrinking or pruning the model parameters, adding noise or dropout to the layers, or augmenting the data with transformations.

## 5. Conclusion

This chapter aims to design an intrusion/preventive system using a window-based CNN and an integrated RNN and AutoE to detect and test the performance of the IDS.

The data file reflects the traffic compositions and online intrusions of specific data packets at a particular time which can be reproducible, modified, and extensible. To be able to do this, training a deep learning model to identify anomalies from a given dataset is one of the initial stages of data preprocessing, which was discussed in the paper. Anomaly-based systems are the most effective IDSs, due to the application of machine learning algorithms embedded within the system. The data packets were converted to images where the pixels were used as input. The CNN architecture shows a three-layer model with high predictive performance. The result shows high performance on CNN as compared to both RNN and AutoE and seems to resist overfitting. The future perspective would be to test the model on other standard methods such as support vector machine and control dynamic systems.

## Acknowledgment

The author wishes to acknowledge Nasarawa State University for their support and sponsorship of this paper.

## Conflicts of Interest

The author declares that she has no conflicts of interest to this work.

## Data Availability Statement

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

## References


[1] Baz, M. (2022). SEHIDS: Self evolving host-based intrusion detection system for IoT networks. *Sensors*, *22*(17), 6505. https://doi.org/10.3390/s22176505

[2] Idrissi, I., Azizi, M., & Moussaoui, O. (2022). An unsupervised generative adversarial network-based-host intrusion detection system for internet of things devices. *Indonesian Journal of Electrical Engineering and Computer Science*, *25*(2), 1140–1150. http://doi.org/10.11591/ijeecs.v25.i2.pp1140-1150

[3] Ullah, M. U., Hassan, A., Asif, M., Farooq, M. S., & Saleem, M. (2022). Intelligent intrusion detection system for apache web server empowered with machine learning approaches. *International Journal of Computational and Innovative Sciences*, *1*(1), 21–27. https://ijcis.com/index.php/IJCIS/article/view/13

[4] Agrawal, S., Sarkar, S., Aouedi, O., Yenduri, G., Piamrat, K., Alazab, M., . . . , & Gadekallu, T. R. (2022). Federated learning for intrusion detection system: Concepts, challenges and future directions. *Computer Communications*, *195*, 346–361. https://doi.org/10.1016/j.comcom.2022.09.012

[5] Maesaroh, S., Kusumaningrum, L., Sintawana, N., Lazirkha, D. P., & Dinda, R. (2022). Wireless network security design and analysis using wireless intrusion detection system. *International Journal of Cyber and IT Service Management*, *2*(1), 30–39. https://doi.org/10.34306/ijcitsm.v2i1.74

[6] Vaigandla, K., Azmi, N., & Karne, R. (2022). Investigation on intrusion detection systems (IDSs) in IoT. *International Journal of Emerging Trends in Engineering Research*, *10*(3), 158–166. https://doi.org/10.30534/ijeter/2022/041032022

[7] Alamleh, A., Albahri, O., Zaidan, A., Alamoodi, A., Albahri, A., Zaidan, B., . . . , & Al-Samarraay, M. S. (2023). Multi-attribute decision-making for intrusion detection systems: A systematic review. *International Journal of Information Technology & Decision Making*, *22*(01), 589–636. https://doi.org/10.1142/S021962202230004X

[8] Goyal, A., Han, X., Wang, G., & Bates, A. (2023). Sometimes, you aren't what you do: Mimicry attacks against provenance graph host intrusion detection systems. In *30th Network and Distributed System Security Symposium*.

[9] Gupta, R. K., Chawla, V., Pateriya, R. K., Shukla, P. K., Mahfoudh, S., & Shah, S. B. H. (2023). Improving collaborative intrusion detection system using blockchain and pluggable authentication modules for sustainable smart city. *Sustainability*, *15*(3), 2133. https://doi.org/10.3390/su15032133

[10] Javadpour, A., Pinto, P., Ja'fari, F., & Zhang, W. (2023). DMAIDPS: A distributed multi-agent intrusion detection and prevention system for cloud IoT environments. *Cluster Computing*, *26*(1), 367–384. http://doi.org/10.1007/s10586-022-03621-3

[11] Jeyaselvi, M., Dhanaraj, R. K., Sathya, M., Memon, F. H., Krishnasamy, L., Dev, K., . . . , & Qureshi, N. M. F. (2023). A highly secured intrusion detection system for IoT using EXPSO-STFA feature selection for LAANN to detect attacks. *Cluster Computing*, *26*(1), 559–574. https://doi.org/10.1007/s10586-022-03607-1

[12] Ullah, F., Ullah, S., Srivastava, G., & Lin, J. C. W. (2024). IDS-INT: Intrusion detection system using transformer-based transfer learning for imbalanced network traffic. *Digital Communications and Networks*. https://doi.org/10.1016/j.dcan.2023.03.008

[13] Borkar, A., Donode, A., & Kumari, A. (2017). A survey on intrusion detection system (IDS) and internal intrusion detection and protection system (IIDPS). In *2017 International Conference on Inventive Computing and Informatics*, 949–953. https://doi.org/10.1109/ICICI.2017.8365277

[14] Lee, T. H., Chang, L. H., & Syu, C. W. (2020). Deep learning enabled intrusion detection and prevention system over SDN networks. In *2020 IEEE International Conference on Communications Workshops*, 1–6. https://doi.org/10.1109/ICCWorkshops49005.2020.9145085

[15] Nadeem, M., Arshad, A., Riaz, S., Band, S. S., & Mosavi, A. (2021). Intercept the cloud network from brute force and DDoS attacks via intrusion detection and prevention system. *IEEE Access*, *9*, 152300–152309. https://doi.org/10.1109/ACCESS.2021.3126535

[16] Nayyar, S., Arora, S., & Singh, M. (2020). Recurrent neural network based intrusion detection system. In *2020 International Conference on Communication and Signal Processing*, 0136–0140. https://doi.org/10.1109/ICCSP48568.2020.9182099

[17] Ng, J., Joshi, D., & Banik, S. M. (2015). Applying data mining techniques to intrusion detection. In *2015 12th International Conference on Information Technology-New Generations*, 800–801. https://doi.org/10.1109/ITNG.2015.146

[18] Alabrah, A. (2023). An efficient NIDPS with improved salp swarm feature optimization method. *Applied Sciences*, *13*(12), 7002. https://doi.org/10.3390/app13127002

[19] Bakro, M., Kumar, R. R., Alabrah, A., Ashraf, Z., Ahmed, M. N., Shameem, M., & Abdelsalam, A. (2023). An improved design for a cloud intrusion detection system using hybrid features selection approach with ml classifier. *IEEE Access, 11*, 64228–64247. https://doi.org/10.1109/ACCESS.2023.3289405

[20] Alem, S., Espes, D., Nana, L., Martin, E., & de Lamotte, F. (2023). A novel bi-anomaly-based intrusion detection system approach for industry 4.0. *Future Generation Computer Systems*, *145*, 267–283. https://doi.org/10.1016/j.future.2023.03.024

[21] Logeswari, G., Bose, S., & Anitha, T. (2023). An intrusion detection system for SDN using machine learning. *Intelligent Automation & Soft Computing*, *35*(1), 867–880. https://doi.org/10.32604/iasc.2023.026769

[22] Louk, M. H. L., & Tama, B. A. (2023). Dual-IDS: A bagging-based gradient boosting decision tree model for network anomaly intrusion detection system. *Expert Systems with Applications*, *213*, 119030. https://doi.org/10.1016/j.eswa.2022.119030

[23] Mohamed, S., & Ejbali, R. (2023). Deep SARSA-based reinforcement learning approach for anomaly network intrusion detection system. *International Journal of Information Security, 22*(1), 235–247. https://doi.org/10.1007/s10207-022-00634-2

[24] Ortega-Fernandez, I., Sestelo, M., Burguillo, J. C., & Pinon Blanco, C. (2023). Network intrusion detection system for DDoS attacks in ICS using deep autoencoders. *Wireless Networks*. https://doi.org/10.1007/s11276-022-03214-3

[25] Pham-Quoc, C., Bao, T. H. Q., & Thinh, T. N. (2023). FPGA/AI-powered architecture for anomaly network intrusion detection systems. *Electronics*, *12*(3), 668. https://doi.org/10.3390/electronics12030668

[26] Hariharan, S., Rejimol Robinson, R., Prasad, R. R., Thomas, C., & Balakrishnan, N. (2023). XAI for intrusion detection system: Comparing explanations based on global and local scope. *Journal of Computer Virology and Hacking Techniques*, *19*(2), 217–239. https://doi.org/10.1007/s11416-022-00441-2

[27] Heidari, A., Navimipour, N. J., & Unal, M. (2023). A secure intrusion detection platform using blockchain and radial basis function neural networks for internet of drones. *IEEE*

*Internet of Things Journal*, *10*(10), 8445–8454. https://doi.org/10.1109/JIOT.2023.3237661

[28] Prabhakaran, V., & Kulandasamy, A. (2023). mLBOA-DML: Modified butterfly optimized deep metric learning for enhancing accuracy in intrusion detection system. *Journal of Reliable Intelligent Environments*, *9*(3), 333–347. https://doi.org/10.1007/s40860-022-00197-y

[29] Thakkar, A., & Lohiya, R. (2023). Fusion of statistical importance for feature selection in deep neural network-based intrusion detection system. *Information Fusion*, *90*, 353–363. https://doi.org/10.1016/j.inffus.2022.09.026

**How to Cite:** Isiaka, F. (2024). Performance Metrics of an Intrusion Detection System Through Window-Based Deep Learning Models. *Journal of Data Science and Intelligent Systems*, *2*(3), 174–180. https://doi.org/10.47852/bonviewJDSIS32021485