Deep Learning Based Early Intrusion Detection in IIoT using Honeypot

Abbasgholi Pashaei[1], Mohammad Esmaeil Akbari[2], Mina Zolfy Lighvan[3] , Asghar Charmin[4]
1,2,4- Department of Electrical Engineering, Ahar Branch, Islamic Azad University, Ahar, Iran.
Email: a-pashaei@iau-ahar.ac.ir, m-akbari@iau-ahar.ac.ir (Corresponding author), a_charmin@sut.ac.ir.
3- Department of Electrical and Computer Engineering Faculty, University of Tabriz, Tabriz, Iran.
Email: mzolfy@tabrizu.ac.ir

Received: 27 September 2022 Revised: 19 October 2022 Accepted: 19 November 2022

ABSTRACT:
The increasing number of Industrial Internet of Things (IIoT) devices presents hackers with a huge attack surface from which to conduct possibly more destructive assaults. Numerous of these assaults were successful as a consequence of the hackers' inventive and unique approaches. Due to the unpredictability of network technology and attack attempts, traditional Deep Learning (DL) approaches are made ineffective. The accuracy of DL algorithms has been shown across a range of scientific fields. The Convolutional Neural Network Model (CNN) technique is an ideal alternative for anomaly detection and classification since it can automatically classify incoming data and conduct calculations faster. We introduce Honeypot Early Intrusion Detection System (HEIDS) that detects anomalies and classifies intrusions in IIoT networks using DL methods. The model is designed to detect adversaries attempting to attack IIoT Industrial Control Systems (ICS). The suggested model is implemented using One-dimensional convolutional neural networks (CNN 1D). Due to the importance of industrial services, this system contributes to the enhancement of information security detection in the industrial domain. Finally, this research gives an assessment of the HEIDS datasets of IIoT, utilizing the CNN 1D technique. With this approach, the prediction accuracy of 1.0 was reached.

KEYWORDS: Industrial Internet of Things (IIoT), Honeypot Early Intrusion Detection System (HEIDS), IIoT HEIDS, Network Security, Deep Learning (DL), One-dimensional Convolutional Neural Networks (CNN 1D).

1. INTRODUCTION

Nowadays, an Industrial internet of things (IIoT) infrastructure, has distinct characteristics that influence industrial facilities design. The device, communication channel, protocol, traffic volume, and quality of service requirements for each application may differ. IIoT devices are making use of industrial protocols not found in Information and communications technology (ICT) or Internet of Things (IoT) environments [1]. These industrial devices have a lifetime of decades and must function under strict time constraints. These industrial devices are in charge of vital infrastructures. Unlike IoT, IIoT applications often need continuous monitoring and control of physical processes [2].

IIoT is the usage of the IoT to the automatization of industrial processes. mission-acute and non-exigent applications use these networks of devices to keep track of and control physical processes [3]. Oil and Gas devices such as used Programmable Logic Controllers (PLCs), Remote Terminal Units (RTUs), and etc., are examples of Industrial Control System (ICS) [4]. IIoT links the physical and logical worlds of engineering and information technology. IIoT sensors collect data from physical components and transmit it to logical elements, while actuators respond to logical elements by modifying physical components. IIoT ICS is a closed system that places a premium on information interaction for the purpose of detecting and controlling the physical environment [5].

In contrast to a corporate information technology network, IIoT devices are often used in large-scale Industrial Control Networks. To ensure interoperability and management, industries prefer to use homogeneous devices and protocols (e.g., Message Queue Telemetry Transport (MQTT), Data Distribution Service (DDS), Constrained Application Protocol (CoAP), and etc.) inside a single network. Additionally, industries handle device life cycles in-house (e.g., firmware upgrades or vulnerability patches). Due to the transparency of Industrial Control Networks and the predictability of IIoT devices, basic security requirements may simply prohibit IIoT.

Paper type: Research paper
DOI:10.30486/mjee.2023.1970288.0
How to cite this paper: A. Gh. Pashaei, M. Esmaei Akbari, M. Zolfy Lighvan and A. Charmin, **"Deep Learning Based Early Intrusion Detection in IIoT using Honeypot"**, *Majlesi Journal of Electrical Engineering*, Vol. 17, No. 2, pp. 69-77, 2023.

The IIoT is transforming the twenty-first century into a smart one. Actuators, sensors, and a variety of other industrial devices are being employed in a growing number of facilities worldwide. While this facilitates connection and efficiency, IIoT devices each have their own set of resource limits, network constraints, and so on, all of which impact application security [6].

Due to the limited power, storage, compute, and communication capabilities available to IIoT devices, it is challenging to develop security mechanisms. However, automation devices were not built with safety in mind and were first thought to be secure owing to their isolation. Stuxnet, and Flame, etc., assault all revealed this premise of security obscurity. As more industrial devices connect to the internet, updates and fixes are available over the internet become vital for decades-old industrial equipment [7].

To safeguard IIoT settings, conventional security methods like encryption, unified threat management, antivirus, firewalls, intrusion detection system should be utilized. However, they prevent security specialists from seeing how attackers attack and studying their behavior. Honeypots provide actionable information about the attackers, making them a feasible choice at this point. A honeypot is a device that is intended to be attacked and be compromised. Honeypots deceive intruders into assuming they have physical access to systems [8]. Honeypots should be used in combination with firewalls and intrusion detection systems to improve system security and prevent further attacks [9].

Executing full-scale Artificial Intelligence, DL on tiny devices is regarded as tough and complex. These critical security objectives must be considered during the training and evaluation of the IIoT Honeypot devices' DL models. These dangers need self-tuning DL components and optimizing their hyperparameters in the IIoT network. Due to barriers and practical insights, the development of trustworthy DL techniques for IIoT honeypot networks is still in its infancy. As a result, the HEIDS-based DL model is proposed to be used to evaluate the trustworthiness and reliability of IIoT networks.

While there are a few Honeypot projects focused on the IIoT, there is no research in the literature that evaluates the performance of DL algorithms integrated with Honeypot to expedite and enhance early intrusion detection accuracy in industrial contexts. DL algorithm connected with the Honeypot, analyses its parallels and distinctions, as well as excerpts crucial elements for the design and deployment of IIoT honeypots. To address this critical research need, we present our complete HEIDS model for IIoT contexts. To our knowledge, this is the first research inside the scholarly literature that reviews the current state-of-the-art early intrusion detection system using honeypot based on DL alghorithm for IIoT. To ease IIoT HEIDS development, a new framework is necessary. This work adds to the following contributions in this regard:

• IIoT HEIDS would benefit from DL methodology in order to provide more innovative analysis and classification to Early Detection (EL) systems.

• A novel CNN-based technique will be developed for the EL of recognized attacks in IIoT networks using DL methodologies.

• To identify abnormalities, assaults, and anomalies in the simulated environment at the university laboratory's Honeypot-based IIoT network, a novel approach was employed to gather and construct a new data set called the HEIDS Dataset, which contains information about the sorts of attacks and network traffic.

• The CNN 1D algorithm will be used to analyze the logs and recorded data from IIoT Honeypot systems in order to establish the most optimum and quickest method for detecting network assaults.

• To accomplish EL of HEIDS, multi-classification utilizing CNN1D convolutional neural networks was used.

Because typical cyber security measures for IIoT devices are ineffectual, DL-enabled solutions are the new approach to safeguard IIoT devices. Among the several DL approaches available, this study used the CNN 1D algorithm on the IIoT's HEIDS dataset. This dataset has been made publicly available for research and testing purposes by the Azad University of Ahar.

The following parts have been structured in light of this information: Section 2 discusses the Related Works. Section 3 defines the Proposed HEIDS Methodology. Section 4 presents Results and Discussion. Conclusions are discussed in Section 5.

2. LITERATURE REVIEW

Securing honeypots with DL applications in the IIOT devices is a significant challenge. Numerous solutions have been presented in the literature for resolving this issue. In [10], the author used the adaptive Honeypot alternative to amass information from attackers. RASSH, an adaptive honeypot based on a medium-interaction Kippo honeypot, was presented by the author in [11]. Furthermore, [12] presented a new SSH honeypot called Q Reinforced Adaptive SSH, which makes use of Cowrie and Deep Q-learning.

The authors of [13] presented IRASSH-T, a self-adaptive IoT honeypot that is based on the QRASSH honeypot and focused on SSH/Telnet. In [14], the authors presented a new kind of Honeypot they call intelligent interaction, which mimics the actions of IoT devices without presenting a danger to the Honeypot. The authors of [15] centred on the usage of the Cowrie Honeypot to detect IoT device assaults and built a Cowrie that included ML capabilities. They discovered

that Support Vector Machine (SVM) delivers the most accurate results, at 97.39 percent. The authors of [16] focused on data collection by emulating an IoT botnet system with Cowrie SSH/Telnet honeypots. Their solution optimizes performance by configuring the prefab command that produces correspond to real-world IoT gadgets and by using connections on ports that are sequence-matching. Additionally, they utilized a clustering technique to discover that the most common way of obtaining or producing data is through botnet attacks on Telnet ports and that Mirai is used in a huge number of attacks on IoT devices. The authors of [17] demonstrated how to use a self-adaptive honeypot based on ThingPot, detection of malware, and the identification of unknown malware, such as those employed in DDoS assaults. The proposed approach collects data about ThingPot honeypot attacks and uses it to build machine learning classifiers.

DiPot, a distributed ICS honeypot, was presented by the authors in [18]. They add to the Conpot framework by simulating ICS protocols, collecting data, and doing analysis using K-means clustering. In [19], the authors presented the design and implementation of a NeuralPot strategy for adapting honeypot technology to the requirements of an industrial network. It is an interactive adaptation of the Conpot honeypot that generates network traffic when another network device is detected. The authors of [20] proposed a novel architecture for developing adaptable honeypots with HARM that makes use of SARSA or Q-Learning. Additionally, they demonstrated adaptation and agility when confronted with a honeypot dataset obtained through an SSH assault method. Numerous analyses have identified honeypot technology as an attack vector for malware. The design and operation of honeypots are based on earlier taxonomies designed to produce large datasets over the duration of longitudinal deployments. This framework may be extended to include honeypots that detect various forms of assaults. In [21], the authors advocated combining Conpot and Dionaea of interaction and connection of the honeypots with the ICS network in order to balance detection accuracy and risk, as well as integrating the honeypot detection feeds with an SDN framework to enable autonomic reconfiguration.

Recent research on attack detection has mostly overlooked IIoT's resource-constrained devices. In applications that demand very intricate DL network calculations for recognizing attack events, job allocation and concurrent processing of learning steps are required. Malicious attacks on IIoT systems may result in unmanageable data traffic, power-draining IIoT devices, resource consumption on the network, and data corruption. The IIoT devices were not designed with general-purpose honeypots in mind. We give a real-time technique for early detecting these risks (e.g., system logs, IP addresses, attack kinds and characteristics, instructions performed and commands executed, and behavioral analysis and etc.) since they must be identified in real-time, including processed data. We are the first to analyze the existing IIoT HEIDS based CNN1D model and studies, to suggest an Industrial ICS compatible with a honeypot, and to describe the aforementioned innovative architecture.

As a consequence, it is critical to do research on the effect of combining DL, a CNN1D algorithm with based Honeypot intrusion detection systems. This study contributes to the performance of an IIoT HEIDS system when combined with a DL system in order to increase its accuracy and computational speed.

3. PROPOSED HEIDS METHODOLOGY

Due to the IIoT HEIDS infrastructure, configuring of topological of devices, model training/testing steps of the DL-based Honeypot Early Intrusion Detection Method in IIoT presented in this study are shown in Fig. 1.

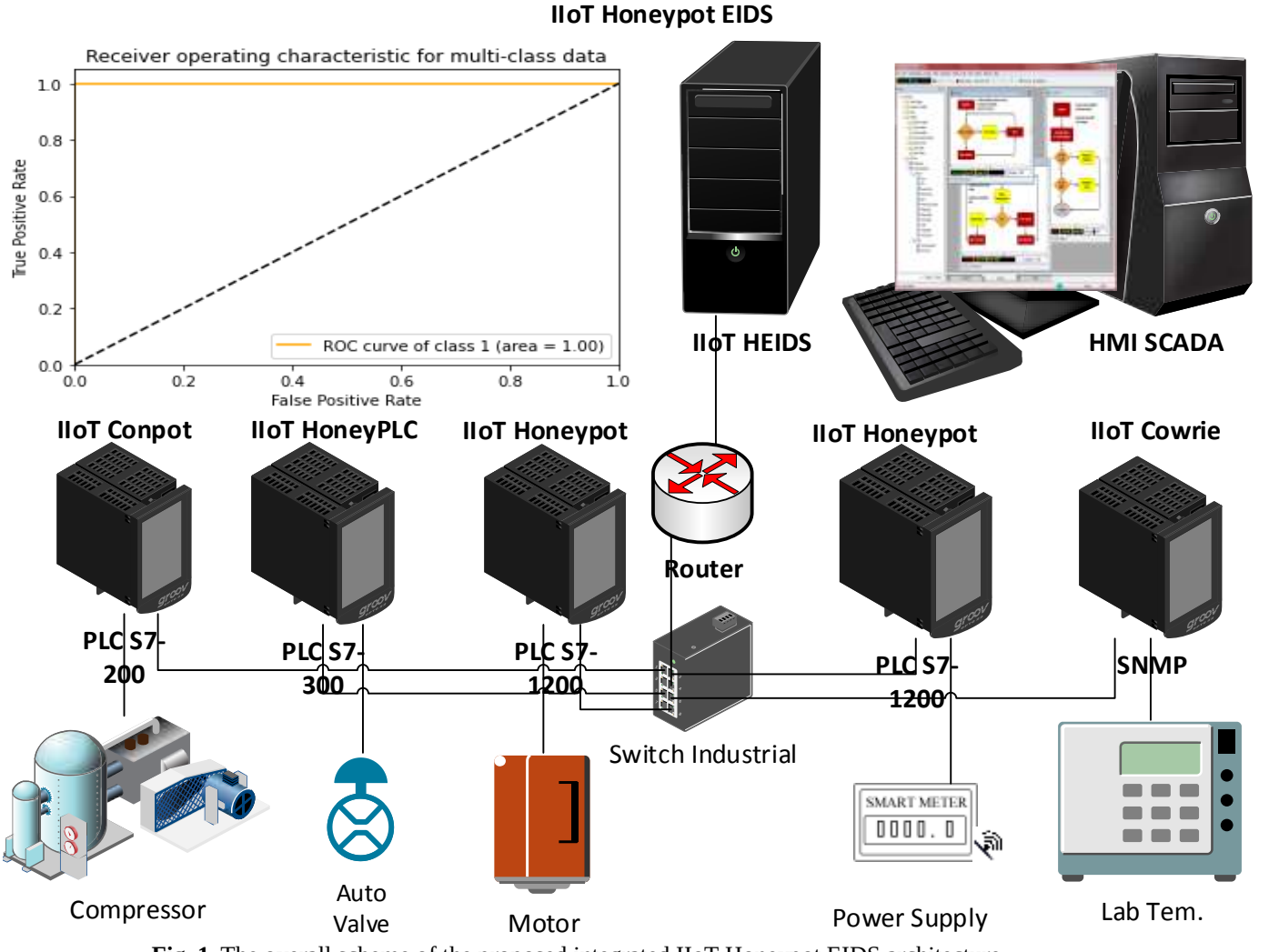


Fig. 1. The overall scheme of the proposed integrated IIoT Honeypot EIDS architecture.

3.1. Packet-Based Detection on EIDS model

Flow-based detection outperforms packet-based detection in experiments. However, flow-based detection has flaws. During data prep, it must pad the short flow or intercept excess bytes in the long flow. Certain flows include bytes in excess of the interception threshold, resulting in severe data loss and reduced detection accuracy. The attacker may also have successfully entered before the forecast is made since many flows include a large number of packets.

The packet-based method distinguishes between fine and coarse-grained packets as data units. These M packets (source/destination IP address, source/destination port, protocol, and etc.) are utilized in the data preparation process. Additionally, the M value has an effect on the accuracy of detection and calculation efficiency. Input units are data packets that contain identical tuples that are received during a waiting time (less than M packets). Apart from the IP address and port, the IP data included in M packets are utilized as input. To maintain the packets' length, fill or intercept the packets' input units. The complement 0 operations are done if the multipacket unit's byte length s is less than or equal to S; otherwise, the last (s-S) bytes are intercepted. Padding or intercepting ensures that all input units have a length of S bytes. Prior to training, the data is compressed to eliminate the computational inefficiencies associated with utilizing raw traffic data as input. In this study, min-max input scaling is utilized to direct networking attention to the component input containing the greatest range. The normalization function's scaler [22] with a minimum-to-maximum range is defined as follows:

$$x_n = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (1)$$

Where x_n is the data vector that has been scaled, $x_{\min}$ and $x_{\max}$ are the data vector that has been supplied, and x is the data vector's various properties.

3.2. DL CNNs model

The term CNN refers to a subset of Feed-Forward Neural Networks (FFNs). This indicates that CNN uses one-dimensional time series data with well-defined time intervals. We use CNN algorithms to describe network traffic events as time series data spanning both benign and malicious connections. The CNN1D method typically consists of five layers, which include convolution, pooling, completely connected, and non-linear activation as ReLU. Each filter is a learnt weighted vector. It starts with convolutional and maximum-pooling layers, continues with sparsely or completely linked layers, and finishes with a decision or classification layer. This deep CNN is capable of managing changes in the input data that are both modest and significant, through the use of supervised learning. A CNN outperforms other neural networks–based feature extraction methods due to the tiny weights of the CNN. A deep CNN is capable of recognizing patterns or objects in one-dimensional (1D) input. In comparison to traditional neural networks, CNNs have a distinct architecture. In a CNN, each layer is composed of a group of neurons that are linked to the preceding layers. By contrast, each layer of a CNN is only partially linked to the neurons of the preceding layer. A hidden layer is constructed by combining one or more CNN layers with an FFN dense layer. CNN gets data in the form 30×1 from an input layer. A CNN generates a Keras of type $30\times3\times64$ (The number 64 refers to the number of filters) and passes it to the max-pooling layer. It decreases the formation of the Keras to $13\times3\times64$. Using the FFN dense layer, this tensor may be used to identify objects or to record temporal patterns.

Where h_i is the feature map at layer i and $h_0 = x$ denotes the input layer, w_i denotes the weight vector of the convolution filter at layer i and b_i and signify the bias vector and activation function, respectively. The rectified linear unit (ReLU) activation function is a frequently utilized non-linear function in CNNs [23]. By sharing the weight and bias vectors, the dominant CNN uses fewer parameters than a standard neural network. Additionally, it does not need hand-crafted feature extraction, as is the case with traditional ML classifiers. The pooling layer down samples the feature map to reduce its dimensionality.

The convolutional 1D layer receives the network traffic data packets as an input vector of size $x = (x_1, x_2, s_3, , , x_{k-1}, x_{29}, cl)$ where x_k signifies features, and cl denotes the class label for the dataset.

Convolution 1D constructs a feature map h_i by convolutioning the input data with a filter w, where f denotes the features in Data packets [24]. As a result of a collection of features f, a new feature map h_i is obtained as

$$kl_i^{hi} = \tanh\left(w^{h_i} x_{i:i+f-1} + b\right) \quad (2)$$

When a bias term is denoted by b. Each set of features f in a data connection is subjected to the filter kl.

The filter kl is applied to each set of features f included in a data connection record $\{x_{1:f}, x_{2:f+1},..., x_{n-f+1}\}$ in order to produce a feature map.

$$kl = \left[kl_1, kl_2, ...kl_{n-f+1}\right] \quad (3)$$

Where kl and the max-pooling operation is applied to each feature map as kl = maxh{kl}. This returns the most important characteristics in which the highest-valued feature is chosen. However, multiple features acquire multiple features, which are then supplied to the fully linked layer. A layer that is completely linked includes the softmax function, which computes the probability distribution over each class [25]. A layer that is completely linked is described mathematically as

$$o_t = SoftMax\left(w_{h_o} kl + b_o\right) \quad (4)$$

Convolution 1D generates a feature map h_i from input data by convolution with a filter w_i. The following equation is used to express the convolution process (5).

$$h_i = f\left(h_{i-1} \otimes w_i = b_i\right) \quad (5)$$

Following the convolution layer, a pooling layer is added to reduce the size of the feature map h_{ij}. The mathematical equation for the pooling layer is written as follows (6):

$$fh_i = pool\left(f\left(fh_{i-1}\right)\right) \quad (6)$$

4. RESULTS AND DISCUSSION

The data required to verify the CNN 1D model should be easily accessible and accurately reflect the behavior of the host or network. Consider how time-consuming and difficult it is to create a dataset. As a consequence, using a benchmark dataset speeds up the diagnostic procedure. Because of the validity of the benchmark data sets, they allow the creation and extraction of more appealing experimental findings in laboratory research, as well as the comparison of the proposed method's outcomes to those from earlier research. To identify the most efficient and best detection model possible for the Honeypot's stored data, HEIDS logs were used in the laboratory to verify the study's results and accuracy. The CIC-IDS 2017 dataset, the NSL-KDD dataset, the Kyoto 2006 dataset, and the CICDoS2019 dataset are all used in this study.

This is a mixed data collection comprised of a huge number of network traffic and system logs containing data, of which the daily data is a portion. The data collection contains a variety of attack types and subtypes, including brute force assaults, distributed denial of service attacks, surveillance network attacks, and penetration attacks. However, there are just a few ways for DL Honeypot intrusion detection in this data set.

4.1. HEIDS Dataset

The intrusion detection system completes the HEIDS collection when industrial network logs are produced. The proposed HEIDS dataset is a tiny but useful tool that allows the system to rapidly suspicious network traffic detection. As a consequence, practically every menace that traverses the network may be detected via the application of adaptable and robust rules. To achieve the aforementioned objectives, a solution is needed for processing the alert data from this enormous dataset. As a consequence, the CSV format is used to analyze alert data since it is the most adaptable and appropriate format for data collection. Table 1, includes a CSV log file named alert.csv in the configuration log's default output, along with 30 features.

Table 1. Feature generation for the IIoT heids dataset.

Feature	Feature	Feature
time	icmpseq	icmpid
icmpcode	date	sig_generator
icmptype	iplen	dgmlen
id	tos	ttl
tcpwindow	tcpln	tcpack
tcpseq	tcpflags	ethlen
ethdst	ethsrc	dstport
dst	srcport	src
proto	msg	sig_rev
sig_id	timestamp	

HEIDS are used to detect ICS attacks. Thus, experiments employing DL techniques on large-scale datasets have been conducted. The CIC-IDS 2017 dataset, the NSL-KDD dataset, the Kyoto 2006 dataset, and the CICDoS2019 dataset are all well-known datasets. The proposed research, however, does not make advantage of existing cyberattack trends. As a result, the HEIDS dataset from the study is enhanced with the most recent Log. Additionally, the new dataset is analyzed using DL methods and compared to previous classification findings.

4.2. Metrics

Methods and criteria for appraising for precision (P), accuracy, recall (R), and F1-score employed by DL in this paper's suggested design to generate the most appropriate model for assessing data attributes are required. As a result, the following criteria are briefly discussed in this article, along with the relevant formulas and equations.

$$\begin{aligned} Accuracy &= \frac{(TP+TN)}{(TP+TN+FP+FN)} \\ R &= \frac{TP}{TP+FN} \\ P &= \frac{TP}{TP+FP} \\ F1 &= 2\times\frac{P\times R}{P+R} \end{aligned} \tag{7}$$

4.3. Experimental Comparison

To begin, the performance of the model for DL detection is examined. Then, using CNN1D as the detection model, the impacts of the four metrics approaches discussed in this research are compared on detection accuracy and computing efficiency. The experiment's hardware environment is comprised of an Intel(R) Core (TM) i5-2450M CPU running at 2.50GHz and 8 GB of RAM. The suggested HEIDS is compared to the model CNN1D with excellent performance in the present research using the common network and data sets CIC-IDS2017, NSL-KDD, Kyoto 2006, and CICDoS2019. The detection precision (P), accuracy, recall (R), and F1-score of the four models are compared and examined via tests. CNN1D employs the same optimum network topology and parameters as the CNN1D model described in this study on the HEIDS dataset. All incoming and outgoing traffic that interacts with the IIoT Honeypot sensors is logged in MySQL, the operating system's dataset management system. The HEIDS dataset was built to ensure the correctness of the operation and its capacity to identify log incursions. To ensure that the results are accurate, it is required to compare the assessment findings for this dataset to those from conventional evaluation techniques.

The proposed IIoT HEIDS is modeled in such a way that it encompasses all available ICSs, including radio frequency systems, PLCs, pressure sensors, position valves, and different actuators, such as control valves and electric motors. In a real-world industrial setting, the suggested system must identify attackers' assaults in real-time. As a result, design and simulation work had to be performed concurrently for multiple distinct regions. As a result, the incoming and outgoing traffic logs for these systems were gathered in existing facilities, which are realistically distributed over many networks using different protocols, and stored in a comprehensive dataset known as the HEIDS system

dataset.

The HEIDS dataset must be properly labeled, to the extent feasible, by comparing simulation findings from other datasets described in this area to those from this research. The IIoT HEIDS research makes available complete data labeling, correctness, and scientific work procedures. Numerous ways and tactics for obtaining accurate data have been examined in this study.

The suggested IIoT HEIDS system's EL performance utilizing stored logs in the dataset was compared to the performance of other large datasets in the globe, and their findings were acquired and studied, processed, and evaluated, and the findings from the DL algorithm for IIoT HEIDS suggest that the technique is successful in detecting early. Additionally, HEIDS was done on all datasets in the same manner as it was on the other datasets.

4.4. Performance Evaluation of DL Detection

As a result, the five datasets were evaluated and processed independently. The findings of individual analyses of each item based on the provided measurements are collected and presented in this part in the form of tables and diagrams with explanations. Explanation of these analyses, procedures, graphs, and findings was accomplished by the use of a specialized application created in the Python programming language for this project.

Table 2 contains information on the datasets, processing, and analysis of the findings acquired from the developed software. Table 2 summarizes the computed results for identifying anomalous traffic using the CNN1D algorithm. Table 2 illustrates the acquired results from the CNN1D algorithm in Table 2 as a bar chart. According to the accuracy criteria, the EIDS Dataset outperforms other datasets.

Table 2. Results for anomalies traffic detection for the cnn1d algorithms used in the research for the five datasets.

Dataset	Accuracy	Recall	Precision	F1
NSL-KDD	0.770	0.652	0.922	0.764
CICIDS2017	0.995	0.863	0.808	0.835
Kyoto2006+	0.9794	0.9514	0.9992	0.9747
CICDoS2019	0.9999	0.9975	0.9809	0.9892
HEIDS	1.000	1.000	1.000	1.000

Another criterion is the F1-Score criterion, which is a mixture of the R and P criteria, and here, as with the R and P criteria, the HEIDS Datasets perform better, as seen in Fig. 2. In the HEIDS dataset, the CNN1D method was used, which has high accuracy and F1-Score. Finally, Table 2 summarizes the results obtained using the model suggested in this study for the EIDS dataset. The output of CNN1D algorithms is seen in Fig. 2. The accuracy requirements suggest that the CNN1D algorithms from EIDS outperform other datasets from DL, as demonstrated by the results of the study.

Additionally, in Table 3 and 4, the improvement rate of the obtained results for detecting traffic anomalies using the proposed HEIDS dataset is expressed in percentage terms for the two essential criteria, accuracy and F1-Score, respectively when compared to the four datasets mentioned in this research. This enhancement is crucial for HEIDS since the approach developed for this research is capable of detecting the incursion of aberrant traffic and demonstrating great accuracy. As a result, this design is very reliable for usage in industrial facilities.

A confusion matrix is a table that is often used to explain the performance of a classification model on a set of test data whose true values are known. A Receiver Operating Characteristic (ROC) curve is a graphical depiction of the diagnostic performance of a binary classifier system when its discrimination threshold is changed. As a result, Fig. 3 illustrates receiver operating characteristic curves in (a, c, e, g, i) and a confusion matrix in (b, d, f, h, k) was utilized to detect traffic anomalies using the Python simulation tool for the NSL-KDD, CIC-IDS2017, Kyoto 2006, CICDoS2019, and IIoT HEIDS dataset.

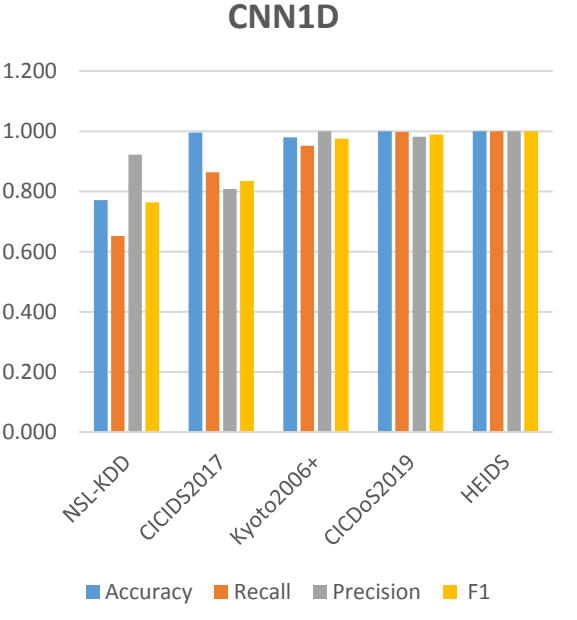


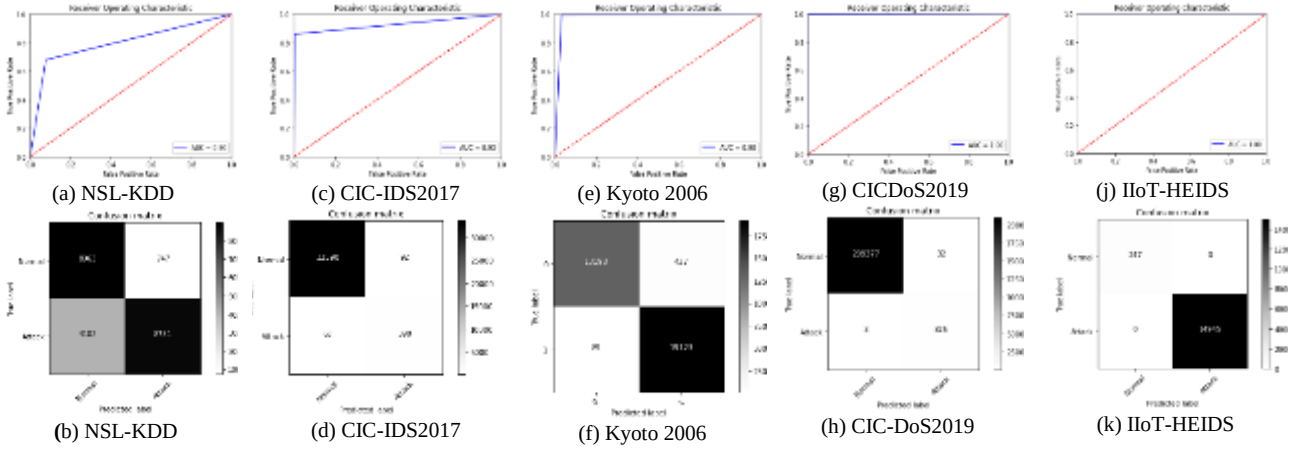
Fig. 2. Using the Python software, we determined the accuracy, R, P, and F1-Score for detecting traffic anomalies in the algorithms used for the CNND1D at NSL-KDD, CIC-IDS2017, Kyoto 2006, cicdos2019, and HEIDS dataset.

Table 3. In comparison to the other four datasets mentioned in this research, the proposed heids dataset shows the greatest improvement in accuracy (percentage) for detecting traffic anomalies.

Dataset Method	NSL-KDD	CIC-IDS2017	Kyoto2006	CICDoS2019
CNN1D	29.80%	0.46%	2.10%	0.07%

Table 4. Comparing the improvement rate of the obtained results for the f1-score to the other four datasets mentioned in this research.

Dataset Method	NSL-KDD	CIC-IDS2017	Kyoto2006	CICDoS2019
CNN1D	30.92%	19.82%	2.59%	1.07%



(a) NSL-KDD (c) CIC-IDS2017 (e) Kyoto 2006 (g) CICDoS2019 (j) IIoT-HEIDS

(b) NSL-KDD (d) CIC-IDS2017 (f) Kyoto 2006 (h) CIC-DoS2019 (k) IIoT-HEIDS

Fig. 3. Confusion matrix and receiver operating characteristic curve measurements were made to identify traffic abnormalities using the Python simulation tool for the methods used in the NSL-KDD, CIC-IDS2017, and Kyoto 2006 datasets, as well as the CICDoS2019 and IIoT HEIDS datasets.

5. CONCLUSION

This paper aimed to investigate the effect of combining the deep learning algorithm CNN1D with the IIoT HEIDS sensors in industrial devices and environments. To build a HEIDS dataset, Port Scanner attacks, DDoS attacks, and etc. were simulated on operating IIoT HEIDS sensors, as well as other critical tools and equipment. It offers a unique solution based on Honeypot and a combination of deep learning algorithm for modeling and forecasting that enables the detection and categorization of characteristics such as normal and abnormal (suspicious) data.

Numerous datasets, including NSL-KDD, CIC-IDS2017, Kyoto 2006, and CIC-DoS2019, were utilized to develop a comprehensive strategy for industrial network categorization, and a dataset was developed using the best characteristics. Finally, the accuracy index was tested using four reference datasets and a dataset in the proposed approach HEIDS in DL-CNN1D in a fully equipped IIoT HEIDS laboratory. HEIDS's accuracy has risen in comparison to the four previously stated datasets. In test data CNN1D, the accuracy of HEIDS on the primary dataset rose by 29.80 percent as compared to NSL-KDD. In test data CNN1D, the accuracy of HEIDS on the primary dataset rose by 0.46 percent when compared to CIC-IDS2017. In comparison to Kyoto 2006, the accuracy of HEIDS on the primary dataset rose by 2.10 percent in test data CNN1D. Finally, the accuracy of HEIDS on the primary dataset rose by 1.07 percent as compared to CIC-DoS2019 in test data CNN1D.

According to the collected findings, when compared to other datasets, the tool built for this study greatly enhanced the analysis of the IIoT HEIDS dataset for early intrusion detection. The completed design, with great accuracy, is capable of detecting abnormal traffic in industrial facilities through its increased sensor network. As a result, it is an efficient and comprehensive cybersecurity system capable of defending against future assaults and zero-day exploits in industrial facilities.

REFERENCES

[1] H. Wang, W. Zhang, H. He, P. Liu, D. X. Luo, Y. Liu, J. Jiang, Y. Li, X. Zhang, and W. Liu, **"An evolutionary study of IoT malware,"** *IEEE Internet of Things Journal*, Vol. 8, No. 20, pp. 15422-15440, 2021.

[2] W. Zhang, B. Zhang, Y. Zhou, H. He, and Z. Ding, **"An IoT honeynet based on multiport honeypots for capturing IoT attacks,"** *IEEE Internet of Things Journal*, Vol. 7, No. 5, pp. 3991-3999, 2019.

[3] J. Franco, A. Aris, B. Canberk, and A. S. Uluagac, **"A survey of honeypots and honeynets for internet of things, industrial internet of things, and cyber-physical systems,"** *IEEE Communications Surveys & Tutorials*, Vol. 23, No. 4, pp. 2351-2383, 2021.

[4] A. Pashaei, M. E. Akbari, M. Z. Lighvan, and Charmin, A, "**Early Intrusion Detection System using honeypot for industrial control networks,**" *Results in Engineering*, 16, 100576, 2022.

[5] W. Tian, M. Du, X. Ji, G. Liu, Y. Dai, and Z. Han, **"Honeypot detection strategy against advanced persistent threats in industrial internet of things: a prospect theoretic game,"** *IEEE Internet of Things Journal*, Vol. 8, No. 24, pp. 17372-17381, 2021.

[6] O. Tsemogne, Y. Hayel, C. Kamhoua, and G. Deugoué, **"Game-Theoretic Modeling of Cyber Deception Against Epidemic Botnets in Internet of Things,"** *IEEE Internet of Things Journal*, Vol. 9, No. 4, pp. 2678-2687, 2021.

[7] Q. Li, X. Feng, H. Wang, and L. Sun, **"Understanding the usage of industrial control system devices on the internet,"** *IEEE Internet of Things Journal*, Vol. 5, No. 3, pp. 2178-2189, 2018.

[8] A. Pashaei, M. E. Akbari, M. Z. Lighvan, and A. Charmin, **"Honeypot Intrusion Detection System using an Adversarial Reinforcement Learning for Industrial Control Networks,"** *Majlesi Journal of Telecommunication Devices*, Vol.12(1), pp. 17-28, 2023.

[9] A. Pashaei, M. E. Akbari, M. Z. Lighvan, and A. Charmin, **"A Honeypot-assisted Industrial Control System to Detect Replication Attacks on Wireless Sensor Networks,"** *Majlesi Journal of Telecommunication Devices*, Vol. 11(3), pp. 155-160, 2022.

[10] G. Wagener, **"Self-adaptive honeypots coercing and assessing attacker behaviour,"** *Institut National Polytechnique de Lorraine-INPL*, 2011.

[11] A. Pauna, and I. Bica, **"RASSH-Reinforced adaptive SSH honeypot."** *In 2014 10th International Conference on Communications (COMM)*, IEEE. Vol., No. Issue, pp. 1-6, 2014.

[12] A. Pauna, A.-C. Iacob, and I. Bica, **"Qrassh-a self-adaptive ssh honeypot driven by q-learning."** *In 2018 international conference on communications (COMM), IEEE*, pp. 441-446, 2018.

[13] A. Pauna, I. Bica, F. Pop, and A. Castiglione, **"On the rewards of self-adaptive IoT honeypots,"** *Annals of Telecommunications*, Vol. 74, No. 7, pp. 501-515, 2019.

[14] T. Luo, Z. Xu, X. Jin, Y. Jia, and X. Ouyang, **"Iotcandyjar: Towards an intelligent-interaction honeypot for iot devices,"** *Black Hat*, Vol. 1, pp. 1-11, 2017.

[15] Y. Zhou, **"Chameleon: Towards adaptive honeypot for internet of things."** *in Proceedings of the ACM Turing Celebration Conference-China*, pp. 1-5, 2019.

[16] B. Lingenfelter, I. Vakilinia, and S. Sengupta, **"Analyzing variation among IoT botnets using medium interaction honeypots,"** in 2020 10th Annual Computing and Communication Workshop and Conference (CCWC): IEEE, pp. 0761-0767, 2020.

[17] M. Wang, J. Santillan, and F. Kuipers, **"Thingpot: an interactive internet-of-things honeypot,"** arXiv preprint arXiv:1807.04114, 2018.

[18] B. Lingenfelter, I. Vakilinia, and S. Sengupta, **"Analyzing variation among IoT botnets using medium interaction honeypots."** in Proceedings of the ACM Turing Celebration Conference-China, pp. 0761-0767, 2019.

[19] I. Siniosoglou, G. Efstathopoulos, D. Pliatsios, I. D. Moscholios, A. Sarigiannidis, G. Sakellari, G. Loukas, and P. Sarigiannidis, **"NeuralPot: An industrial honeypot implementation based on deep neural networks."** *in 2020 IEEE Symposium on Computers and Communications (ISCC):* IEEE, pp. 1-7, 2020.

[20] S. Dowling, M. Schukat, and E. Barrett, **"New framework for adaptive and agile honeypots,"** *ETRI Journal*, Vol. 42, No. 6, pp. 965-975, 2020.

[21] S. Maesschalck, V. Giotsas, B. Green, and N. Race, **"Honeypots for Automatic Network-Level Industrial Control System Security."** *in 14th EuroSys Doctoral Workshop*, 2020.

[22] Y. Wang, Y. Jiang, and J. Lan, **"Fcnn: An efficient intrusion detection method based on raw network traffic,"** *Security and Communication Networks*, Vol. 2021, 2021.

[23] R. Vinayakumar, K. Soman, and P. Poornachandran, **"Applying convolutional neural network for network intrusion detection."** *in 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI): IEEE*, pp. 1222-1228, 2017.

[24] G. Swapna, K. Soman, and R. Vinayakumar, **"Automated detection of cardiac arrhythmia using deep learning techniques,"** *Procedia computer science*, Vol. 132, pp. 1192-1201, 2018.

[25] A. K. Verma, P. Kaushik, and G. Shrivastava, **"A network intrusion detection approach using variant of convolution neural network."** *in 2019 International Conference on Communication and Electronics Systems (ICCES):* IEEE, pp. 409-416, 2019.