A Novel CNN and GA-Based Algorithm for Intrusion Detection in IoT Devices

Ibrahim Darwish[1†] and Samih Montser[2††], Mohamed R,Saadi[3††],

South Valley University, South Valley University, Luxor University

Summary
The Internet of Things (IoT) is the combination of the internet and various sensing devices. IoT security has increasingly attracted extensive attention. However, significant losses appears due to malicious attacks. Therefore, intrusion detection, which detects malicious attacks and their behaviors in IoT devices plays a crucial role in IoT security. The intrusion detection system, namely IDS should be executed efficiently by conducting classification and efficient feature extraction techniques. To effectively perform Intrusion detection in IoT applications, a novel method based on a Conventional Neural Network (CNN) for classification and an improved Genetic Algorithm (GA) for extraction is proposed and implemented. Existing issues like failing to detect the few attacks from smaller samples are focused, and hence the proposed novel CNN is applied to detect almost all attacks from small to large samples. For that purpose, the feature selection is essential. Thus, the genetic algorithm is improved to identify the best fitness values to perform accurate feature selection. To evaluate the performance, the NSL-KDDCUP dataset is used, and two datasets such as KDDTEST^{21} and KDDTEST^{+} are chosen. The performance and results are compared and analyzed with other existing models. The experimental results show that the proposed algorithm has superior intrusion detection rates to existing models, where the accuracy and true positive rate improve and the false positive rate decrease. In addition, the proposed algorithm indicates better performance on KDDTEST^{+} than KDDTEST^{21} because there are few attacks from minor samples in KDDTEST^{+}. Therefore, the results demonstrate that the novel proposed CNN with the improved GA can identify almost every intrusion.

.Keywords:
IoT, Intrusion Detection, CNN, Genetic Algorithm, NSL-KDD CUP dataset.

1. Introduction

Internet of Things (IoT) is considered a new standard that incorporates the internet and physical objects in various domains, such as health, environmental monitoring, industrial process, and home automation. In our day-by-day activities, IoT extends its presence by using internet-connected devices. The security of IoT has caused many associated benefits and challenges. To enhance security, the Intrusion detection System, shortly called IDS, has been studied for more than two decades and is considered the essential tool for information systems and network protection [1]. By analyzing and monitoring malicious traffic in the network, the intrusion detection IDS is defined as the developed system to protect smart IoT devices from various attacks [2]. Due to the architectural constraints, standards, and particular protocol stacks, it is a challenge to implement conventional IDS methods on IoT. Since not all types of attacks can be defended, a novel method should be considered [3]. It includes physical hardware application using network probe to transfer the secure data to a remote server, performs detection types are performed, and requires intensive resources. The IDS plays a vital role in resisting hacker intrusion, thus developing an efficient IDS is necessary. A novel Convolutional Neural Network (CNN) based IDS technique, namely IDS-CNN, is proposed in this paper. This system comprises various open-source tools, such as traffic analysis, packet capture interface, and the machine learning interface TensorFlow, to pre-process the training-based neural network, test the network, and response to the intrusion. In the end, the evaluated experiment and simulation results show precision performance [4]

1.1 Challenges

- The conventional IDS method implementation on IoT is considered a challenge due to the architectural constraints, standards, and particular protocol stacks.
- Computational and resource constraints may lead to detecting intrusion quite difficult.
- It is required to improve the accuracy rate of intrusion detection in IoT devices by detecting all types of attacks.
- A better feature dataset should be used for performance evaluation.

To identify all types of attacks and improve the identification accuracy, a novel CNN and Genetic Algorithm-based method is proposed in this study. The significant contributions are summarized as:

a) To rapid and accurately detect all types of attacks, intrusion detection must be effective. For that purpose, the enhanced CNN is used for

Manuscript received September 5, 2023
Manuscript revised September 20, 2023
https://doi.org/10.22937/IJCSNS.2023.23.9.7

classification, and the improved Genetic Algorithm is utilized for efficient feature extraction.

b) To evaluate the classification performance, two datasets, namely KDDTEST21 and KDDTEST+ are used. The performance has been compared with other existing models to demonstrate effectiveness in malicious attack detection.

c) Both KDDTEST21 and KDDTEST+ datasets are also compared to measure the better dataset.

The rest of the paper is systematized as follows. The relevant literature review is discussed in Section II. The proposed novel CNN and improved GA are described in Section III. The comparison performance results and analysis are conducted and exhibited in Section IV. Finally, the paper concludes in Section V.

2. Related Work

The intrusion detection system based on CNN provides a reasonable detection rate compared with other IDS classifier through the KDDCUP dataset. But the false alarm rate needs to be improved [5]. Eventually, to increase the detection rate accuracy, alarm filtering is considered. The hybrid ant colony optimization algorithm based on unsupervised clustering reduces the false alarm rate and generates a higher detection rate, whose convergence is accelerated by the K-means clustering technique [6]. Further, the deep learning model-based recurrent neural network (RNN) is implemented to design the IDS system, called RNN-IDS. The RNN approach is compared with other machine learning methods for evaluating its performance of intrusion detection on the NSL-KDD dataset. Intrusion detection However, the accuracy is incorrect, and the time taken for training is exceeded [7]. The detection rate of known and unknown attacks is improved using hybrid intrusion detection in the multi-level extreme learning machine (ELM) and the support vector machine (SVM). The overall training time is reduced, and the performance of the intrusion detection system is improved by modified K-means on the KDD 99 dataset with high accuracy [8]. The NSL-KDD dataset is considered an advanced version of the KDD 99 dataset. The intruder's network protocol stack allows anomalies in network traffic and is detected and analysed through this dataset. The network attack and protocol bonding are revealed. The optimization methods have to be proposed to increase the accuracy rate [9].

In IDS, dimensionality reduction plays a significant role because anomalies detection leads to time-consuming in high dimensional network traffic. For feature selection, the firefly algorithm is used. The following features are exposed to KDD CUP 99 dataset, and Bayesian networks are used for classification. Accuracy is improved while detecting intrusion from the features [10]. Additionally, to enhance IDS accuracy, the intrusion detection system is based on text convolutional neural networks, and the random forest is used to extracts the raw network packet features. The random forest exhibits better performance on structured data, and the CNN handles unstructured data. Few attacks are identified by this system. However, high computational complexity occurs at a higher length [11]. The sensors outdoor with IoT devices exhibit computational and resource constraints strictly, and thus the detection of intrusion is quite a challenge. Machine learning techniques like K-NN, neural network, and SVM are used by constructing lightweight access control protocols that lengthen IoT system lifetime and save energy [12]. Using data mining techniques, the accuracy is not sufficient in some cases. Hence, a novel intrusion detection system with linear correlation coefficients based on a conditional random field for feature selection and classification. This system achieves higher accuracy. Optimized standard techniques are required to yield more efficiency [13]. Advanced attack detection methods by traditional IDS are not well efficient. Thus, deep learning-based IDS are used to solve this issue, primarily concentrating on Denial of Service (DoS) attacks.

The KDDCUP 99 intrusion dataset used for evaluation is comprised of DoS attacks, probing attacks, remote to local R2L attacks, and user to root U2R attacks. The recent IDS dataset is also used. The new IDS system compared with RNN shows relatively better efficiency in the detection of intrusion. Multi-class classifications are yet to be focused on [14]. IoT security is enhanced by various machine learning approaches and deep learning methods. The deep learning method used for DoS attack detection also shows better accuracy, and thus IoT system is more efficacious [15]. Another approach to protecting IoT systems is considering the various IoT malware detection methods. Performing quickly and accurately is viewed as a challenge in IoT fields. Using a convolutional neural network, the dynamic analysis of malware detection in IoT can identify various behaviours extracted in associated process, system call, virtual system, and memory. Further, they are converted into IoT malware behaviour images. By analysing and visualizing the malware behavior, the IoT device damages are reduced [16]. For fog security, the full artificial automated IDS is developed. The multi-layer RNN is executed on fog computing and performance evaluation on NSL-KDD dataset. The potentiality stability is resulted after the evaluations in terms of metrics. Besides the R2L, U2R, and probing attacks, the automated IDS system-based RNN focuses on DoS attacks mostly since the extended computational overhead results in other attacks [17]. Extreme learning machine (ELM)-based single hidden layer

neural network is implemented for dimensionality reduction data classification. Both the efficiency of execution and intrusion detection are improved. It recognizes the intrusion quickly when using NSL-KDD dataset. The accuracy yields higher value and thus results better performance [18].

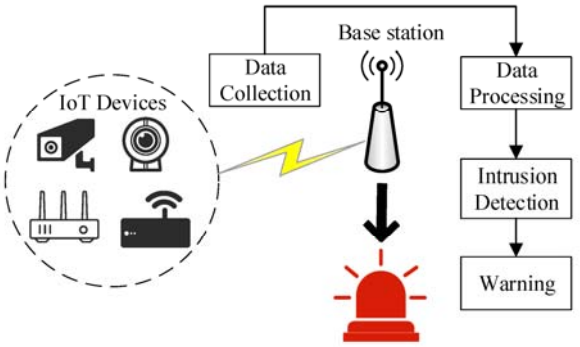

Fig. 1. Intrusion detection in IoT [18]

The above Figure 1 describes that the intrusion detection function is completed at the base stage, and it shows storage and computing ability. The dimensionality of data is reduced after data collection using linear discriminant analysis. The alarm module is presented to notify attacks. At the final stage, the IDS decides about the presence of attacks and their corresponding types from ELM classification.

Each stages for intrusion detection in It is clearly explained below as per figure.1.

IoT devices

It includes software's, wireless sensors, computer devices and actuators. These devices are attached to specific objects that functions via internet connection, permitting the data transfer among people or objects automatically without any human involvement.

Data Collection

It is the stage in which sensors are used to track the physical conditions. The IoT devices and the technology connected through internet can measure and monitor the data in real-time. This data is transferred, stored as well as reclaimed at any time.

Data Processing

Typically, It is comprised of four components. It includes IoT devices (sensors), data collection (connectivity), data processing and finally User Interface (UI). Sensors are IoT devices that gather data and then transfer it via internet. This data could then be sent for accumulating, processing or advance the information distribution. To sense the huge data collected by the IoT sensors, it has to be processed which is defined as the collection and exploitation of data items to generate meaningful information. The motivation behind data processing is the conversion of raw-data into meaningful information.

Intrusion detection

This stage is utilized to protect the IoT devices from different kinds of attacks. It functions in the IoT system's network layer. Thus, Intrusion detection is vital to enhance the network security as it permits to detect and also respond to the malign traffic. The main profit of Intrusion detection is to confirm that the warning is send to the IT personnel during any attack or Intrusion detection.

3. Proposed Methodology

3.1 Novel Convolution Neural Network and Genetic Algorithm based Structure

In the proposed work, the training process of the novel convolutional neural network (CNN) is described, and the number of parameters is learned. The overall novel CNN and the improved GA are shown in Figure 2. Existing issues like failing to detect few attacks from smaller samples are focused, and hence the proposed model is used to detect almost all attacks from minor to larger pieces. For that purpose, best feature selection is essential. Thus, the genetic algorithm is improved to identify the best fitness value and proper feature selection. This section provides further details of novel CNN and improved GA as below.

3.1.1 Improved Genetic Algorithm (GA)

The improved GA solves decrease in accuracy when identifying fitness value issues, initializing the population from lesser pairing sets and enhancing the population again and again by providing new pairings from the remaining search space using improved customize genetic operators. The overall procedure of GA is mentioned in Algorithm 1, and the components are described below. At first, the GA's initial population *ini_pop* is performed, and the further main loop contains selection, crossover, mutation, and other operators imposed in sequential order.

Algorithm 1:

Initialize population ini_pop

Evaluate fitness(ini_pop)

While criterion is not met, do

Selection (parent_pop)

Crossover(parent _1,parent_2)

Mutation(child_pop)

Feasibility heuristic(child_pop)

Repeated paring removal(child_pop)

Evaluate fitness(child_pop)

Population replacement (combined_pop)

Crossover(parent _1,parent_2)

CombinedPairs= combined list of pairing in parent _1,parent_2

For each child chromosome do

For each known-part do

Random selection of solution in combainedpairs

End

For each unknown-part do

Select pairing from combined pairs and their dissimilarity with known part

End

End

Repeated pairing removal(child_pop)

For each chromosome in child_pop do

If bv_i^e *== 1 then*

Set bv_i^e *= 0*

If all values are not covered in chromosome then

Set bv_i^e *= 1*

End

population updation()

For all combination of the parent, list do

X_i = *max_val(Set of parents list)*

If bv_i^e *> 1 then*

For all values in a combination list

Random generation of discrete value between 0 and X_i

$sX_i = X_i$ *– min_val(Set of parents list)*

End

End

End //While

From the above-mentioned algorithm, each chromosomes' length cannot be equal to several pairings, and a chromosome with 2-bits gene encoding is adapted. In the chromosome structure, the bv1 denotes binary variable concerning pairing selected in the second bit bv2. The chromosome structure builds up in two parts: the known part and the unknown part. The known part contains pairings that participate in the quality of solution evaluation. However, the strange part has pairings that are not the part of the solution but are considered for diversity maintenance corresponding to the known part. The fixed-length chromosome is utilized in this work, whereas the known and unknown parts' lengths are allowed to vary dynamically.

Selection

The selection operator is imposed for chromosome selection, which in turn to be parents for child chromosome reproduction. A binary tournament selection operator is assumed in the proposed improved GA, where two chromosome sets, and everyone is formed randomly. Among these sets, the parents with the best fitness value are transferred to the crossover stage.

Crossover

The crossover stage is referred to as the transition stage, in which the parent chromosome's genetic information is transferred to the following generation as same as new child chromosome reproduction. The following crossover operators are discussed:

Crossover 1- The probabilities depend on the parents' fitness to decide the genes which are transferred to the child chromosome.

Crossover 2- To increase convergence rate, greediness is integrated into reproduction operators corresponding to domain knowledge. The child chromosome's known part generates zero solution build-up by randomly selecting pairings from parent chromosome columns. The procedure

is shown in Algorithm 1. By repeating the processes, the crossover operators change to constitute two-child chromosomes from two-parent chromosomes.

Mutation

The mutation operator is imposed on the resulted child chromosomes after the crossover part. Premature convergence is prevented by the mutation part. With the help of some probability, getting stuck at local optima can be avoided by changing child chromosome genes. Compare the two mutation operators in the current work, such as Mutation 1 (bit-flip mutation operator) and Mutation 2 (mutation operator based on population's fitness solution density). In Mutation 1, if the ith gene is selected for mutation, followed by bv1i flipped from 0 to 1 or 1 to 0. However, in Mutation 2, if the ith gene is selected for mutation, followed by bv1i mutated from 0 to 1 with 1s in the fittest individual percentage probability and similarly for 1 to 0.

Feasibility Heuristics

The feasibility of resulted child chromosomes is not definite, and in other words, it may or may not cover all values. The feasibility heuristic is needed to carry out the feasibility in the child chromosome. Simultaneously, the child's chromosome fitness should be maintained. The feasibility heuristic is changed in which a repeated pairing removal is involved at the end. The minimum quality index pairing is selected for every uncovered value. The repeated pairing removal step includes the heuristic, which identifies and deletes the same values exhibited by those pairings shown in Algorithm 1.

Fitness Function Evaluation

The objective function of the problem is the fitness function, and it evaluates the chromosome's fitness value. The important objective is to reduce the total pairing cost and also cover all values at once, and the unique fitness function is exhibited. The performance is measured by the fitness value assigned to the chromosome. Once the population generation is completed, the fitness value of each chromosome can be measured. The better the chromosome, the higher the fitness value. Thus, the chromosomes of individuals corresponding to greater fitness values show a better chance of survival.

Population Updating

The final step of the Genetic Algorithm is the population updating step, in which the parent and child chromosome's survival population is selected and becomes the parent population for following the genetic algorithm sequence. The generational and steady-state approach is usually followed by population updating. In this work, the generational approach was chosen, and the best n chromosomes among the n parents and n children are chosen, which are transferred to the following generation

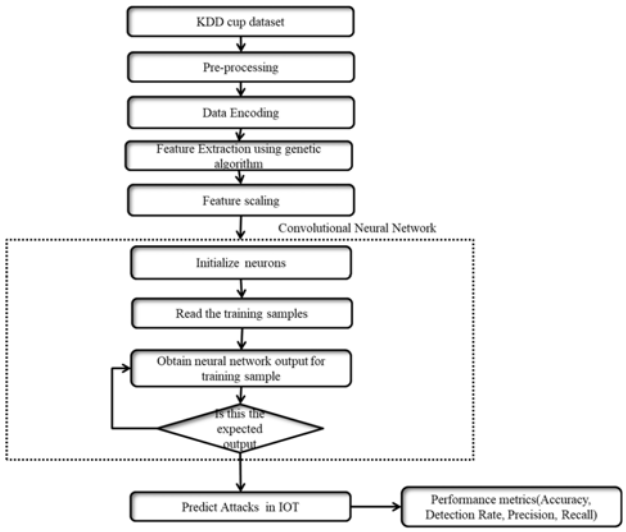

Fig. 2. The flow diagram of the novel CNN and the improved GA

3.2 Novelty in CNN Architecture

The training process of the proposed novel CNN is mathematically explained in this section.

3.2.1 Forward Propagation

The Convolutional Neural Network is comprised of convolutional layers, pooling layers, and fully connected layers. As shown in Figure 3, followed by the pooling layer, the convolutional layer consists of multiple convolutional kernels to generate various feature maps [19]. The ith feature map of the lth convolutional layer is denoted as k_i^l

$$F_j^l = \sum_{i=1}^{N^{l-1}} K_{ij}^l \; X \; k_i^{l-1} + \; b_j^l \, , k_i^l = f\left(F_j^l\right) \qquad (1)$$

where k_i^{l-1} is the ith feature map of the l-1th layer, N^{l-1} is several feature maps, the convolutional kernel is K_{ij}^l concerning the ith map in the lth layer and the ith map in the l-1th layer, the bias term of the mentioned kernel is termed as b_j^l, and f (F_j^l) is determined as an element-wise non-linear activation function that presents non-linearity to multilayer networks. The standard activation function is termed as ReLU, sigmoid, and tanh functions. The integrating features, shift-invariance by feature maps resolution reduction, and parameter decrease are obtained by pooling layers after convolutional layers. The pooling function is denoted as downsample (p_j^l), k_i^l represents each feature map,

$$\left|p_j^l\right| = \text{downsample}(k_i^l) \qquad (2)$$

The standard pooling operations are defined as max pooling and average pooling. The pooling operations take

as the kxk region, and the single value output is shown as mean or max value in the specific region. In some cases like VGG and AlexNet, the fully connected layers after convolutional and pooling layers aim to learn the feature maps at the middle level. For implementing fully connected layers, a large number of weight parameters are needed. The feed-forward procedure is similar to a standard artificial neural network, which is calculated as

$$F_j^l = \sum_{i=1}^{N^{l-1}} k_i^{l-1} \, X \, S_i^l + b_j^l \, , \; k_i^l = f(F_j^l) \tag{3}$$

Where S_i^l and b_j^l are considered as the weight vector and the bias terms of the i^{th} filter and the l^{th} layer correspondingly. In a neural network, the softmax activation is imposed on the last layer, transforming the final layer output to a significant probability distribution.

Let o_i and r_i denote the predicted label and the ground truth label. Thus, the loss function can be calculated as

$$Ls = \frac{1}{2}\sum_{i=1}^{N^l} |r_i - o_i|^2 , o_i = k_i^l \tag{4}$$

Where $N^l$the node number of the l^{th} layer, the last layer is shows the output, and Ls is the total classification error corresponding to output nodes. The loss function in Equation 4 is referred to as Euclidean Loss.

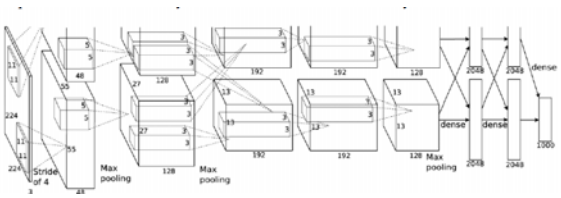

Fig. 3. The architecture of Convolutional Neural Network [19].

3.2.2 Backward Propagation

The errors originated in the output layer are propagated to the input layer in backward propagation from label prediction. The bias term and weight vectors are restructured layer by concerning these errors. The parameters can be calculated as

$$S_{ij}^l = S_{ij}^{l-1} + \gamma \frac{\varphi Ls}{\varphi S_{ij}^{l-1}} , \; \beta_i^l = \beta_i^{l-1} + \gamma \frac{\varphi Ls}{\varphi \beta_i^{l-1}} \tag{5}$$

Where γ is the learning rate, $\frac{\varphi Ls}{\varphi S_{ij}^l}$ and $\frac{\varphi Ls}{\varphi \beta_i^l}$ denoted as loss function partial derivatives corresponding to S_{ij}^l and β_i^l individually and it can be expanded as

$$\frac{\varphi Ls}{\varphi S_{ij}^l} = \frac{\varphi Ls}{\varphi F_i^l} \frac{\varphi F_i^l}{\varphi S_{ij}^l}, \frac{\varphi Ls}{\varphi \beta_i^l} = \frac{\varphi Ls}{\varphi F_i^l} \frac{\varphi F_i^l}{\varphi \beta_{ij}^l} \tag{6}$$

Let δ_i^l defines the right-hand part of Equation 6 as the l^{th} layer error term, and it is mixed the second part result, which can be restructured as

$$\frac{\varphi Ls}{\varphi S_{ij}^l} = \delta_i^{l+1} f'(F_i^l) k_i^l \, , \frac{\varphi Ls}{\varphi \beta_i^l} = \delta_i^{l+1} f'(F_i^l) \tag{7}$$

If the fully connected layer is the lth layer, the output layer is the l+1th layer and the error term is δ_i^l. The activation function's derivative of the lth layer is $f'(x)$. Following the chain rule, the layer l and layer l+1 are convolutional layers and the δ_i^l can be expressed as

$$\delta_i^l = (\sum_{j=1}^{N^{l-1}} S_{ji}^l \, X \, \delta_i^{l+1}) f'(F_i^{l-1}) \tag{8}$$

If the pooling layer is the lth layer and the convolutional layer is the l+1 layer, then the δ_i^l is calculated as

$$\delta_i^l = (\sum_{j=1}^{N^{l-1}} K_{ji}^l \, X \, \delta_i^{l+1}) f'(k_i^l) \tag{9}$$

Where the pooling function derivative is $f'(k_i^l)$ and the linear function is $f(k_i^l)$, which refer to the mean or the max function. And if the convolutional layer is the l^{th} layer and the pooling layer is the l+1th, the error term δ_i^l is measured as

$$\delta_i^l = \text{upsample}\delta_i^{l+1} f'(k_i^l) \tag{10}$$

Where the upsampling operation is denoted as upsample(). If mean pooling is adopted by the pooling layer, then the error is uniformly distributed by upsampling, which is fed into the former layer. The unit selected as the max receives all errors in max pooling. Based on previous changes, the bias term and the weight vector are updated following the up-down direction.

2.3 Dataset Selection

The NSL-KDD CUP dataset is used in this work [20]. This dataset is chosen for training the model and evaluating the performance. A huge amount of data, repeated in the KDDCUP 99 dataset, is removed in the NSL-KDD CUP dataset. Thus, this dataset can run into the requirements of verification experiments because its ratio distribution is quite reasonable, usable, and balanced. The dataset constitutes an intrusion record comprised of 42-dimensional features of attacked type, 3-dimensional symbol structures, and 38-dimensional digital features. The NSL-KDD CUP dataset significantly contains four huge attack type data of R2L, probe, U2R, DoS, and one normal data. Thirty-nine subclasses are categorized into four attack types. Three sub-datasets are also presented, such as KDDTrain as a training set, $KDDTest^{21}$, and $KDDTest^{+}$ as a test set. The distribution of the sample category is shown

below.

Table 1-Distribution for sample category [21]

Category	KDDTraining	KDDTest[21]	KDDTest[+]
Normal	67343	2152	9711
Probe	45927	4342	2421
U2R	52	200	200
R2L	995	2754	2754
DOS	11656	2402	7458
Total	125973	11850	22544

Besides, Table 2 mentions the feature classifications for the sample category [21].

Table 2-Feature classification for sample category

No.	Features	Types	No.	Features	Types
1	Duration	continuous	22	Is_guest_login	discrete
2	Protocol_type	symbolic	23	Count	continuous
3	Service	symbolic	24	Srv_count	continuous
4	Flag	symbolic	25	Serror_rate	continuous
5	Src_bytes	continuous	26	Srv_serror_rate	continuous
6	Dst_bytes	continuous	27	Rerrore_rate	continuous
7	Land	discrete	28	Srv_rerror_rate	continuous
8	Wrong_fragment	continuous	29	Same_srv_rate	continuous
9	Urgent	continuous	30	Diff_srv_rate	continuous
10	Hot	continuous	31	Srv_diff_host_rate	continuous
11	Num_failed_logins	continuous	32	Dst_host_count	continuous
12	Root_shell	discrete	33	Dst_host_srv_count	continuous
13	Su_attempted	discrete	34	Dst_host_same_srv_rate	continuous
14	Num_root	continuous	35	Dst_host_diff_srv_rate	continuous
15	Num_file_creations	continuous	36	Dst_host_same_src_port_rate	continuous
16	Num_shells	continuous	37	Dst_host_srv_diff_port_rate	continuous
17	Num_access_files	continuous	38	Dst_host_serror_rate	continuous
18	Num_outbound_cmds	continuous	39	Dst_host_srv_serror_rate	continuous
19	Root_shell	discrete	40	Dst_host_rerror_rate	continuous
20	Num_compromised	continuous	41	Dst_host_srv_rerror_rate	continuous
21	Is_host_login	discrete			

4. Results and Discussion

4.1 Environmental Configuration

In the proposed study, the comparison and test verification were implemented on the Windows 10 operating system. The novel CNN and GA is executed on python 3.7 with the anaconda spyder IDE environment. The hardware and software configuration is list below.

Table 3-Environmental configuration

Hardware Configuration	Software Configuration
CPU - Intel Core i7 – 7700 @ 2.80 GHz	Windows 11
GTX 1050	Python 3.7
16GB RAM	Anaconda Spyder

4.2 Performance Metrics

4.2.1 True Positive Rate (TPR)

The true positive rate is also known as the sensitivity, which in turn estimates the positive proportions that are identified correctly by itself.

$$TPR = \frac{TP}{TP+FN} \quad (11)$$

Where TP denotes the number of true positives and FN is the number of false negatives.

4.2.2 False Positive Rate (FPR)

It is defined as the proportion of all negatives that, in turn, yield positive outcomes. In other words, the conditional probability of the test results provides an event that is not present. It is also known as the false alarm, which indicates the condition that is to be fulfilled.

$$FPR = \frac{FP}{FP+TN} \text{ and also } FPR = (1 - Specificity) \quad (12)$$

Where FP is the number of false positives and TN is the number of true negatives.

4.2.3 Accuracy

It is defined as the closeness of the measured values to known or standard values. In other words, it is also referred to as the weighted arithmetic means of precision and inverse precision. The weighted arithmetic means of the recall and inverse recall, i.e., weighted by the prevalence.

$$Acc = \frac{TP+TN}{P+N} \text{ or } \frac{TP+TN}{TP+TN+FP+FN} \quad (13)$$

4.2.4 Precision

Precision is defined as the fraction of retrieved instances that are relevant and also called a positive predictive rate. Or it can also be the fraction of retrieved documents that are relevant to the query.

$$\text{Precision}=\{ \frac{relevant\ document \cap retrieved\ document}{retrieved\ document} \} \text{ or } \frac{TP}{TP+FP} \quad (14)$$

4.2.5 Recall

A recall is the proportion of related instances that are recovered. Therefore, both accuracy and recall are based on an understanding of significance and measurement. It is the fraction of documents relevant to the request that is successfully retrieved in the retrieval of data. It is estimated by the formula given below.

$$\text{Recall}= \{ \frac{relevant\ document \cap retrieved\ document}{relevant\ document} \} \text{ or } \frac{TP}{TP+FN} \quad (15)$$

4.3 Comparative Experimental Results

To evaluate the proposed classification method, i.e., the novel CNN, and the feature extraction technique, namely improved GA, the two datasets KDDTest[21] and KDDTest[+] were used. The comparative analysis and its description are shown below.

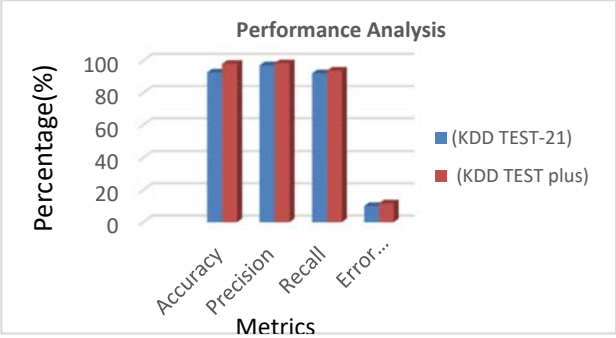

Fig. 4. Comparative performance analysis of KDDTEST[21] and KDDTEST[+]dataset.

Fig. 4 depicts that the KDDTest+ dataset shows higher accuracy, recall, and precision respectively. Compared with KDDTest21, it also displays a greater number of error rates.

Forecasted set of class\Actual set of classes	DoS	Probe	U2R	R2L	Normal
DoS	**1997**	96	0	128	171
Probe	156	**3986**	0	92	108
U2R	0	17	**148**	12	23
R2L	0	77	94	**2219**	364
Normal	104	52	33	183	**1780**

Table 5-The proposed CNN's classification evaluation results for the KDDTest[21] dataset

Forecasted set of classes\Actual set of classes	DoS	Probe	U2R	R2L	Normal
DoS	**2001**	94	0	127	170
Probe	155	**3989**	0	91	107
U2R	0	17	**150**	11	22
R2L	0	76	92	**2222**	364
Normal	104	52	33	183	**1780**

Table 6-Test evaluation and comparison results of the proposed and existing techniques [21]

Method	FPR	Acc	TPR
DBN-KDD TEST-21	4.46	88.69	89.48
IBWNIDM-KDD TEST-21	3.96	90.91	92.42
RNN-KDD TEST-21	1.75	79.47	80.58
Proposed-KDD TEST-21	**1.4**	**92.56**	**93.06**
DBN-KDD TEST plus	0.85	92.45	95.68
IBWNIDM-KDD TEST plus	0.76	95.36	95.55
RNN-KDD TEST plus	0.92	93.08	94.39
Proposed-KDD TEST plus	**0.72**	**97.89**	**96.11**

The proposed novel CNN's evaluation results are compared with DBM, IBWNIDM, and RNN methods for both KDDTest^{21} and KDDTest^{+} datasets, respectively, as shown in Table 6. The proposed KDDTest^{21} and KDDTest^{+} results are shown in bold numbers. In the case of accuracy in detecting intrusion, the proposed method offers higher detection compared to the other three models for both datasets. Especially for the KDDTest^{+} dataset, the accuracy is still higher compared with the KDDTest^{21} dataset. In the case of true positive rate, the proposed novel CNN shows higher values than the rest of the models using both datasets. For false positive rate results, the lower values are shown for both datasets in the proposed method as expected. It results in lower values compared with the rest of the models' FPR values.

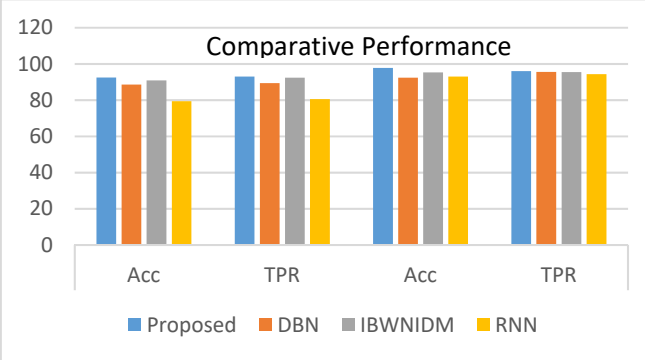

Fig. 5. Comparative performance analysis of the proposed and existing three models [21] in terms of Accuracy and TPR

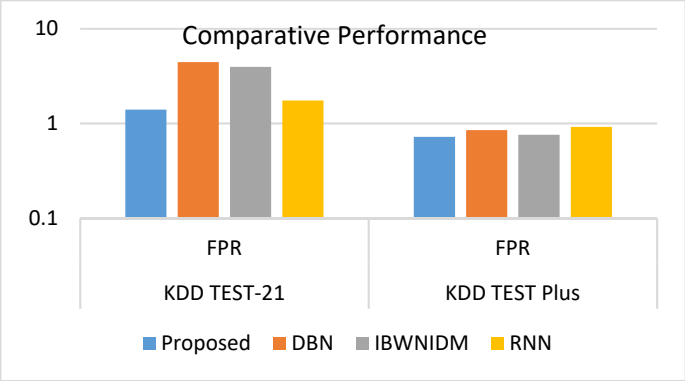

Fig. 6. Comparative performance analysis of the proposed and existing three models [21] in terms of FPR.

Hence, the proposed novel CNN shows its strong ability for intrusion detection identification in both datasets. The proposed method shows better extraction and also a good classification effect using two datasets. The above table is diagrammatically explained in Figs. 5 and 6.

5 Conclusion

Based on the convolutional neural network, a novel CNN classification in intrusion detection is performed in IoT applications. Using the pre-processed dataset of KDDTEST^{21} and KDDTEST^{+}, the classification performances are evaluated. To identify and classify all malicious attacks, advanced feature extraction techniques are required. For that reason, an improved genetic algorithm is used to determine the best fitness value. The results of the proposed work show that the novel CNN-based improved GA has the greater performance to detect intrusions compared with other existing models. The two datasets are also compared, and the KDDTEST+ shows more efficient performance compared with KDDTEST21. Hence, for all IoT applications, the proposed novel CNN-based improved GA can be used to detect all types of attacks in an accurate manner. In future aspects, the proposed work will be evaluated on more datasets, and also it can be compared with more existing models to demonstrate its performance.

Acknowledgment: We would like to thank TopEdit (www.topeditsci.com) for the English language editing of this manuscript.

References

[1] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *Journal of Network and Computer Applications,* vol. 84, pp. 25-37, 2017.

[2] S. Pundir, M. Wazid, D. P. Singh, A. K. Das, J. J. Rodrigues, and Y. Park, "Intrusion detection protocols in wireless sensor networks integrated to the Internet of

Things deployment: Survey and future challenges," *IEEE Access,* vol. 8, pp. 3343-3363, 2019.
[3] E. Benkhelifa, T. Welsh, and W. Hamouda, "A critical review of practices and challenges in intrusion detection systems for IoT: Toward universal and resilient systems," *IEEE Communications Surveys & Tutorials,* vol. 20, no. 4, pp. 3496-3509, 2018.
[4] H. Wang, Z. Cao, and B. Hong, "A network intrusion detection system based on convolutional neural network," *Journal of Intelligent & Fuzzy Systems,* no. Preprint, pp. 1-15, 2019.
[5] Y. Liu, S. Liu, and X. Zhao, "Intrusion detection algorithm based on convolutional neural network," *DEStech Transactions on Engineering and Technology Research,* no. iceta, 2017.
[6] X. Yang and Z. Hui, "Intrusion detection alarm filtering technology based on ant colony clustering algorithm," in *2015 Sixth International Conference on Intelligent Systems Design and Engineering Applications (ISDEA)*, 2015: IEEE, pp. 470-473.
[7] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Ieee Access,* vol. 5, pp. 21954-21961, 2017.
[8] W. L. Al-Yaseen, Z. A. Othman, and M. Z. A. Nazri, "Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system," *Expert Systems with Applications,* vol. 67, pp. 296-303, 2017.
[9] H. Ji, D. Kim, D. Shin, and D. Shin, "A Study on comparison of KDD CUP 99 and NSL-KDD using artificial neural network," in *Advances in computer science and ubiquitous computing*: Springer, 2017, pp. 452-457.
[10] B. Selvakumar and K. Muneeswaran, "Firefly algorithm based feature selection for network intrusion detection," *Computers & Security,* vol. 81, pp. 148-155, 2019.
[11] E. Min, J. Long, Q. Liu, J. Cui, and W. Chen, "TR-IDS: Anomaly-based intrusion detection through a text-convolutional neural network and random forest," *Security and Communication Networks,* vol. 2018, 2018.
[12] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning: How do IoT devices use AI to enhance security?," *IEEE Signal Processing Magazine,* vol. 35, no. 5, pp. 41-49, 2018.
[13] B. Riyaz and S. Ganapathy, "A deep learning approach for effective intrusion detection in wireless networks using CNN," *Soft Computing,* pp. 1-14, 2020.
[14] J. Kim, J. Kim, H. Kim, M. Shim, and E. Choi, "CNN-Based Network Intrusion detection against Denial-of-Service Attacks," *Electronics,* vol. 9, no. 6, p. 916, 2020.
[15] B. Susilo and R. F. Sari, "Intrusion detection in IoT Networks Using Deep Learning Algorithm," *Information,* vol. 11, no. 5, p. 279, 2020.
[16] J. Jeon, J. H. Park, and Y.-S. Jeong, "Dynamic Analysis for IoT Malware Detection with Convolution Neural Network model," *IEEE Access,* 2020.
[17] M. Almiani, A. AbuGhazleh, A. Al-Rahayfeh, S. Atiewi, and A. Razaque, "Deep recurrent neural network for IoT intrusion detection system," *Simulation Modelling Practice and Theory,* vol. 101, p. 102031, 2020.
[18] D. Zheng, Z. Hong, N. Wang, and P. Chen, "An improved LDA-based ELM classification for intrusion detection algorithm in IoT application," *Sensors,* vol. 20, no. 6, p. 1706, 2020.
[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097-1105.
[20] L. Dhanabal and S. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms," *International Journal of Advanced Research in Computer and Communication Engineering,* vol. 4, no. 6, pp. 446-452, 2015.
[21] H. Yang and F. Wang, "Wireless network intrusion detection based on improved convolutional neural network," *IEEE Access,* vol. 7, pp. 64366-64374, 2019.

Ibrahim Darwish received his BSc (1996) Alexandria University, in Computer Science. He worked as an assistant teaching at the King Khaled University-College of Computer Science from 2001 to 2015.His received his Master Degree from Institute of postgraduate studies and researches, Alexandria University. His interests in-routing distributed-algorithm for wireless sensor networks-Cloud Computing, Load Balancing, Parallel Processing, Big Data