

Intrusion prevention system using convolutional neural network for wireless sensor network

Pankaj Chandre¹, Parikshit Mahalle², Gitanjali Shinde³

¹Department of Computer Engineering, Smt. Kashibai Navale College of Engineering, Savitribai Phule Pune University and MITSoE, MIT ADT University, Pune, India

²Department of Artificial Intelligence and Data Science, Vishwakarma Institute of Information Technology, Pune, India

³Department of Computer Engineering, Vishwakarma Institute of Information Technology, Pune, India

Article Info

Article history:

Received Apr 2, 2020

Revised Feb 15, 2022

Accepted Mar 1, 2022

Keywords:

Deep learning

Intrusion detection

Intrusion prevention

Wireless sensor network

ABSTRACT

Now-a-days, there is exponential growth in the field of wireless sensor network. In wireless sensor networks (WSN's), most of communication happen through wireless media hence probability of attacks increases drastically. With the help of intrusion prevention system, we can classify user activities into two categories, normal and suspicious activity. There is need to design effective intrusion prevention system by exploring deep learning for WSN. This research aims to deal with proposing algorithms and techniques for intrusion prevention system using deep packet inspection based on deep learning. In this, we have proposed deep learning model using convolutional neural network. The proposed model includes two steps, intrusion detection and intrusion prevention. The proposed model learns useful feature representations from large amount of labeled data and then classifies them. In this work, convolutional neural network is used to prevent intrusion for WSN. To evaluate and check the effectiveness of the proposed system, the wireless sensor network dataset (WSNDS) dataset is used and the tests are performed. The test results show that proposed system has an accuracy of 97% and works better than existing system. The proposed work can be used as future benchmark for the deep learning and intrusion prevention research communities.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Pankaj Chandre

Research Scholar at Department of Computer Engineering, Smt Kashibai Navale College of Engineering, Savitribai Phule Pune University

Pune, Maharashtra, India

Email: pankajchandre30@gmail.com

1. INTRODUCTION

Now a day, due to its large deployment in many military and civil services, growing attention is given to the wireless sensor network (WSN). These services include environmental sensing, healthcare, smart cities, and area monitoring. Such essential services would increase security attackers' interest in such networks, and involve the development of ongoing security solutions. These solutions can avoid, track and restrict potential attacks that could seriously affect the services provided by these networks [1]. With internet technology's rapid growth, problems with organized security have ended up as extraordinary with each passing day. Researchers have been continuously working on intrusion detection system (IDS), to protect the network against malware and attacks [2].

An IDS can be categorized into two groups on the basis of detection objects: Host-based IDS and network-based IDS. Host-based IDS tracks the actions or state of the host system service, such as whether

there is unauthorized installation and access in system events or whether memory status of file system status anticipatory data are present. Host-based IDS is based on the system of event log. But the disadvantage of HIDS, is that it has a low false alarm rate and again, it is not able to analyze internet related behavior. Network-based IDS works differently than the host-based IDS [3]. An IDS is a set of hardware and/or software that detects unwanted attempts to gain access to, control, or deactivate computer systems through a network. In order to safeguard a system from network threats, a network-based IDS monitors and examines network traffic. All inbound packets are examined by a network-based IDS for suspicious patterns. Depending on the seriousness of the threat, the device may take action, such as notifying administrators or blocking the originating IP address from accessing the network. Therefore, a more automated and smarter solution needs to be studied to construct an intrusion detection system focused on self-adaptive behavior detection and dynamic expansibility.

Day by day, approaches to network attacks are being changed with the approach of huge information. A pattern of intelligentization and complication has been shown by new network intrusions. Traditional anomaly detection technologies have a hard time, having an effect on new network intrusions and delivering a satisfactory result. The established attack database is used in detection by virtually all commercialized intrusion detection systems. The subject of research has been on how to train a high-efficiency model and reduce training costs at the same time. Many new smart intrusion detection algorithm implementations have emerged in the face of different new network intrusions, as the times require [4]. If we expand this research method to the convolution neural network (CNN) classifier, we can find a way to achieve higher detection accuracy than the previous approaches. With complex structures or combinations of nonlinear transformations, a CNN can realize high-grade data abstraction and can therefore, achieve a higher detection rate. We suggest a new approach to malware detection through the use of a CNN that doesn't want superior know-how of malware operations or extracting capabilities of community visitors as it routinely learns capabilities. The suggested solution is not only a cost effective solution to intrusion detection but also, it is more accurate in recognizing current malware that is altered to prevent detection. In this section, similar work related to intrusion detection and prevention are presented. Below is another discussion of the topic's extant literature.

Mohammadpour *et al.* have suggested a deep learning based system for intrusion detection [5]. In the suggested work, an author has used CNN with the NSL-KDD dataset. The appropriate separation of the training and testing dataset is used in all the experimentation. For implementation, CNN NIDS is used with two class classification, the proposed model is implemented using the python programming language and Keras library. In the proposed system, authors have used layers such as convolutional, fully connected layer, and pooling and the suggested system is able to detect anomaly-based intrusion with greater efficiency.

Sharma *et al.* have presented a machine learning (ML) based model, and the suggested model is capable to detect intrusion like web-based attacks [6]. In the proposed work, authors have tried to analyze and then address the root cause of the false negative and false positive. The appropriate separation of the training and testing dataset is used for all the experimentation. And the dataset used in this work, is CSIC 2010 HTTP. The performance of the suggested model is assessed by considering the parameters such as accuracy, precision and recall. The suggested model is tested by using ML classifiers like J48, one rule and Naïve Bayes. All the implemented techniques are compared with each other and the J48 classifier gives the highest intrusion detection accuracy, which is 94.5%.

Pandey and Singh proposed a ML based model to detect a black hole type of attack [7]. In the suggested work, an author has used two ML based approaches such as artificial neural network (ANN) and support vector machine (SVM). The two approaches are capable to discover a blackhole attack in the network. The suggested model is tested multiple times with different sets of nodes, and the obtained result is also different. Implementation is carried out by using an NS2 simulator. The proposed model gives better throughput for black hole attack detection, and it is equal to 88.68%, and the PDR is 92.91%.

Tan *et al.* have suggested a ML based model to detect intrusion [8]. In the proposed work, authors have used synthetic minority oversampling techniques (SMOTE) to manage the dataset, and a random forest (RF) classifier for training purposes. Implementations are conducted, and the results of the random forest algorithm give better intrusion detection accuracy with 92.39% and with the SMOTE dataset it gives 92.57% accuracy, which is a little bit improved. To improve the intrusion detection accuracy, both SMOTE and the random forest algorithm are combined.

Kumar *et al.* have suggested network intrusion detection system (NIDS) by using a supervised machine learning technique [9]. In the proposed work, the authors have supervised the machine learning technique with feature selection. The authors have presented a unique model by using a supervised ML technique and the presented system is able to check whether the network traffic is malicious or not. To improve the detection selection rate, the authors have combined various supervised learning algorithms such as ANN and SVM. The use of ANN is to wrap the feature selection and the role of SVM is to classify the network traffic. For the evaluation of the presented system, the NSL-KDD dataset is used. On the basis of

experimentation, the performance of the suggested model is superior as compared to the current system in respect to an intrusion detection success rate. The proposed model using ANN gives an intrusion detection accuracy of 94.02%.

Al-issa *et al.* have suggested a machine learning based IDS and the suggested model is capable to detect a denial of service (DoS) attack in WSN [10]. In the suggested work, an author has used the decision tree and the support vector machine classifier. The suggested system is based on a signature-based attack. The appropriate separation of the training and the testing dataset is used for all the experimentation. In the proposed work, authors have a WSN-DS dataset, which is a standardized dataset for a WSN. The used dataset consists of four types of attacks like black hole, gray hole, time division multiple access (TDMA) and flooding. After implementation of both the classifiers, the findings are compared with each other and the decision tree gives better results in respect of true positive rate (TPR) and false positive rate (FPR).

Shenfield *et al.* have presented a novel approach to detect malicious traffic by the use of ANN [11]. In the proposed work, authors have used the concept of deep packet inspection. For the experimentation, authors have used normal and abnormal traffic data such as images, word documents, and dynamic links. The performance of the suggested model is evaluated by considering the parameters namely, false positive rate and large dataset. An author has considered maximum 1,000 epochs to train the proposed system. The proposed system can distinguish between normal data and malicious data with better detection accuracy in a repeated 10-fold cross validation.

Sohi *et al.* have proposed an enhanced network IDS using deep learning [12]. In this proposed work, to handle the unknown attacks, authors have introduced recurrent neural network (RNN) based IDS. The proposed RNNIDS is able to find out the complex patterns in attacks and after that, one similar pattern is generated. The proposed model, is able to generate new as well as unseen attacks with the improvement in intrusion detection rate. Experiments are performed by using the standard datasets, which are publicly available and the proposed system, gives an IDR improvement of up to 16.67%.

Ugochukwu *et al.* have suggested a ML based IDS [13]. In the suggested work, an author has used machine learning algorithms like RF, random tree, J48 and Bayes Net. The KDDCup99 dataset is used and the appropriate separation of the training and testing dataset is used for all the experimentation. For experimentation, the waikato environment for knowledge analysis (WEKA) tool is used. The results of RF and random tree perform good in respect of precision, recall and F1 score.

Elmasry *et al.* have suggested a framework for intrusion detection [14]. In this suggested work, an author has suggested a deep learning framework and double particle swarm optimization (PSO) metaheuristic are used. In the proposed work, PSO is used to perform two tasks like feature selection and hyper parameters in one step. The proposed model is pre trained to select optimized features and hyper parameters automatically. In this work, to evaluate the performance, three deep learning classifiers are used like deep neural network (DNN), LSTM-RNN and deep belief network (DBN). NSL KDD was the dataset used in this study, and the evaluation is performed for binary as well as multiclass classification. In the proposed work, intrusion detection rate is improved and the FAR is reduced. The proposed model, works better for network intrusion detection.

Hoyos *et al.* have suggested a ML based approach to detect intrusion like DoS attacks [15]. In the suggested model, an author has used the supervised learning-based classifier such as SVM. In the proposed work, the SVM classifiers are used, and it captures network data, filters that data and then, normalizes the data. And then, this information is transmitted to evaluate the suggested system. The suggested work is capable to detect distributed DoS attacks. The intrusion detection accuracy of the suggested system is high and it is equal to 99%. Again, the proposed model is capable to decrease the false-positive rate (FPR) and false-negative rate (FNR) as compared to the existing models.

The most essential reasons for the requirement for IPS are protection from DoS attacks and protection from a variety of critical exposures. More attention has been paid to intrusion detection systems; however, they are ineffective since they only identify intrusions that have already occurred. When we require more and more security, we need a powerful model, which we can get by utilising the deep learning classifier. As a deep learning classifier, we use a convolutional neural network in our study. Our suggested system will be capable of detecting and preventing intrusions such as DoS, grayhole and blackhole attacks.

2. METHOD

In this work we are going to use deep learning classifier. We can detect as well as prevent intrusion by using performing a deep packet inspection. In existing work only header part of a packet is analyzed. In this work, we are going to analyze header as well as data and on the basis of that we can take some actions in terms of normal or abnormal. The proposed algorithm as well as the main notions of the proposed system are discussed in this part. The layers of CNN such as convolution layer, flatten layer, dense layer and max

pooling layer are employed as depicted in Figure 1. The design of a CNN-based intrusion prevention system is shown in Figure 1. The following are the stages of the planned architecture:

- Raw data: As we know that, data preparation in deep learning is an ongoing process of transforming raw data, so that we can apply the proposed model to predict insight based current statistics. In this step, input is provided as raw data.
- Preprocessing: Large-scale datasets generally contain noisy, redundant and diverse data types that pose crucial challenges to the exploration of information and data modeling. Intrusion detection algorithms typically deal with one or more raw input data types, such as the CNN algorithm, which deals with image data. That's why we prepare data and then convert it into numerical data.
- Test data: The test dataset is used to provide an impartial assessment of the final model that fits into the training dataset. Sometimes, the test dataset is referred as a validation dataset. The test dataset may be referred to as the validation dataset if the initial dataset was partitioned into just two sub sets. In this step, the appropriate selection of the dataset is done for experimentation.
- Convolution layer: The potential for CNN to automatically learn in parallel a large number of filters, each unique to a training dataset, under the constraints of a particular predictive model problem such as classification is called CNN innovation. CNN uses a filter on the input to create feature maps that highlight the existence of observed features in the data.
- Max pooling layer: Pooling layers can be used to select the greatest values on the function maps and use them as inputs to following levels. In theory, any operation can be carried out in pooling layers, but in reality, only maximum pooling is used because we want to detect outliers.
- Fully connected layer: Before the CNN classification output, fully connected layers are mounted and utilised to flatten the data before classification. It's used to compile data from final feature maps before generating final categorization. The number of nodes and the activation function are among the factors that are used.
- Flatten layer: Flatten layer is used to convert the data into 1-dimensional array, and then it is given as a input to the next layer. Performing the flatten operation, can be considered as a key step in the convolutional neural network.
- Filtered data: Lastly, the filtered data is considered as an output.

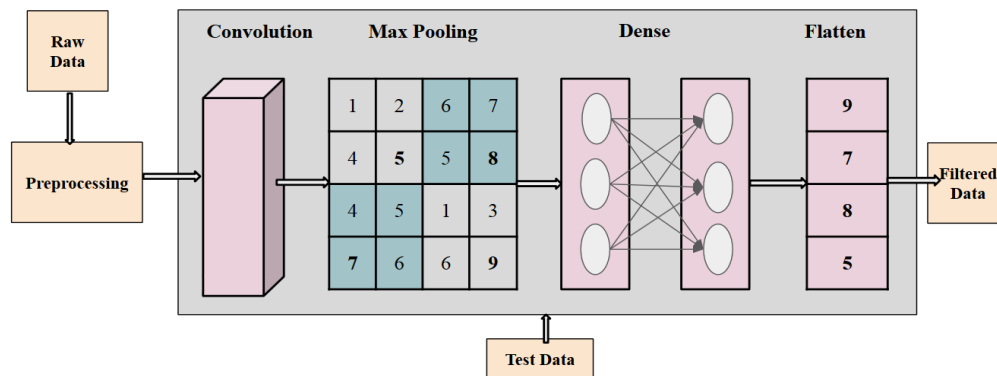


Figure 1. CNN based intrusion prevention system architecture

2.1. Algorithm

The proposed system's algorithm for intrusion prevention using CNN is presented and discussed in this section. We started our work by giving a dataset to a system as an input. The acceptable input data was then preprocessed in order to improve the model's outputs. After that, we fed the model with the processed data in order to train it. The CNN model was used in this work. First and foremost, we looked at the model's convolution layer. Conv1D was used in this example, and it primarily works with sequential datasets. The kernel size for the convolution layer is $[4 \times 4]$. Second, we employed the max pooling layer, which has a size of $[2 \times 2]$ and is largely used to lower the size of the representation and the network's computation. Third, we used the dense layer, which basically connects every neuron in one layer to every neuron in another layer. The dense layer has a size of 50 pixels, and the dropout value of 0.2 is used to prevent overflow. The number of epochs is fixed to 10 and an adaptive moment estimation (adam) is utilised for optimization. The rectified linear unit (ReLU) is utilised as an activation function again, this time with the softmax activation function. Finally, the Flatten layer has been applied. The proposed model was then tested, with test data as an input for

testing. We started with a null value for attack type and a zero value for status, which means that all attack types are treated as normal data. The status was then checked using the detect function. When the status value is 0, the data is considered normal and the message “Normal” is printed. If the value of the state is not 0, the data is considered anomalous and we check the attack pattern of this data, which is printed. Finally, we evaluate the accuracy and print the confusion matrix.

<i>Input-</i>	<i>Dataset (A WSN-DS)</i>
<i>Output-</i>	<i>ATTACK TYPE(Normal, Grayhole, Blackhole, Flooding, TDMA)</i>
<i>Steps-</i>	
<i>Step-1</i>	<i>Start</i>
<i>Step-2</i>	<i>Input- Labelled Training Data as</i> $X = \{X^{(1)}, X^{(2)}, X^{(3)}, \dots, X^{(c)}\}$, <i>Where, c is the total number of classes</i>
<i>Step-3</i>	<i>CNN ← X,</i> <i>Here, the training data are provided as an input to CNN for the feature extraction</i>
<i>Step-4</i>	$F = \{F^{(1)}, F^{(2)}, F^{(3)}, \dots, F^{(c)}\}$, <i>Here, F is the extracted feature vector</i>
<i>Step-5</i>	<i>Apply Conv1D (4, 3, input_shape=(3,3), activation='relu',padding='same')</i> <i>Apply Max Pooling (pool_size=2, stride=2)</i> <i>Apply Dense (50, input_dim=num_features, activation='relu')</i> <i>Apply Flatten</i>
<i>Step-6</i>	<i>Test the Model,</i> <i>Input=Y and show the appropriate attack</i>
<i>Step-7</i>	<i>Matrix of Accuracy Evaluation (Y test, Y pred)</i>
<i>Step-8</i>	<i>Confusion Matrix (Y test, Y pred) to print</i>
<i>Step-9</i>	<i>Stop</i>

For experimentation, the programming language python is utilised. Python codes and the function used like MaxPooling(), Flatten(), Softmax() are also given. Figure 2 gives details about python code.

```

Out [28]: (299729, 3, 6)

In [29]: import tensorflow as tf
import matplotlib.pyplot as plt
import numpy as np

In [30]: from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv1D
from keras.layers.convolutional import MaxPooling1D
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from keras import backend as K

num_features = X_train.shape[1]

model = Sequential()
model.add(Conv1D(4, 3, input_shape=(3,6), padding="same"))
model.add(MaxPooling1D(pool_size=2))
#model.add(Convolution1D(15, 3, activation= 'relu' ))
#model.add(MaxPooling1D(pool_size=(2, 2)))
#model.add(Dropout(0.2))
#model.add(Dense(128, activation= 'relu' ))
model.add(Flatten())
model.add(Dense(50, input_dim = num_features, activation= 'relu' ))
model.add(Dense(5, activation= 'softmax' ))
# Compile model
model.compile(loss= 'categorical_crossentropy', optimizer= 'adam', metrics= [ 'accuracy' ])

Using TensorFlow backend.

In [31]: model.fit(X_train, y_train,)
score = model.evaluate(X_test, y_test)

Epoch 1/1

```

Figure 2. Details of python code

3. RESULTS AND DISCUSSION

This section gives a high-level overview of the research. This section implements the CNN classifier. Python is used as a scripting language, and WSN-DS is used as an implementation dataset [16]. The CNN is employed for classification in this experiment. The model was built with Python and

TensorFlow on a system with 16 GB of RAM, an Intel Corei7 processor, and a Windows 10 operating system. The complete entries in a WSN-DS dataset are classified into several labels such as normal, blackhole, flooding, grayhole and TDMA, and the LEACH approach is used to collect data. In this the parameters used, number of nodes used are 100, number of clusters used are 5, network area used are 100*100 m, size of packet considered 500 bytes and the size of packet header consider 25 bytes. The proposed system can prevent attacks like a black hole, gray hole, flooding, and TDMA [17]. To determine whether the data is normal or malicious some features are selected such as is that node is cluster head or not (IS_CH), advertising messages sent and received (A_SNT and A_RCD), join request messages sent and received (J_SNT and J_RCD), scheduled messages sent and received (SC_ST and SC_RD), data sent and received (D_SNT and D_RCD) and lastly the send code [18]. Table 1 shows WSN-DS dataset.

Table 1. A sample dataset

ID	IS_CH	WHO CH	A_SNT	A_RCD	J_SNT	J_RCD	SC_ST	SC_RD	D_SNT	D_RCD	SEND_C ODE	ATTACK TYPE
117099	0	117078	0	2	1	0	0	1	22	0	1	Normal
118000	0	118058	0	7	1	0	0	1	30	0	3	Normal
118040	1	118040	1	6	0	23	1	0	0	1173	0	Grayhole
118005	1	118005	1	6	0	12	1	0	0	1020	0	Grayhole
114083	1	114083	1	1	0	13	13	0	0	0	0	TDMA
115016	1	115016	1	2	0	49	49	0	0	0	0	TDMA
610091	1	610100	1	27	0	0	0	0	0	0	0	Blackhole
610092	1	610100	1	27	0	0	0	0	0	0	0	Blackhole
102001	1	102001	6	14	0	51	1	0	0	150	0	Flooding
102034	1	102034	4	16	0	8	1	0	0	888	0	Flooding

The most important thing in deep learning is the data. By using that data, you are going to build a model. If your data is incomplete or irrelevant, then you will build an incorrect model. And if your data is clean, then you can build a correct model. If you are expecting better result with your model, then before using you need to clean data. So, data cleaning is nothing but the process of identifying missing values, irrelevant, incomplete and incorrect data and then deleting, replacing or modifying them according to the necessity. For example- if you have a dataset with five columns and the third columns data is irrelevant for your work, then simply you can drop those particular columns and can focus on remaining four columns. After performing data cleaning, you can say that your data is clean and you can use it to build correct model.

The attributes of a used dataset are as-ID, this is used to differentiate all nodes from each other. IS_CH, this is used to check whether the node is cluster head or not. WHO CH, this uses the id of the current node. A_SNT, this is the count of advertising messages sent to the normal node from the cluster head. A_RCD, this is the count of advertising messages received by cluster head from the normal node. D_SNT, this is the number of data packets sent from a normal node to its head in the cluster. J_SNT, this is the count of the join request messages sent by the normal node to the cluster node. D_RCD is the number of packets received since the start of the cluster. J_RCD is the number of merge request messages received by the cluster head from a normal node. SC_ST is the number of ad schedule messages sent to the nodes. SC_RD, this is the number of ad schedule messages received from the cluster head. Send code, it is used as cluster sending code. Attack type, used as a label to distinguish between attack types.

3.1. Normal

By examining attributes such as the number of advertising broadcast messages sent and received, the number of join request messages sent and received, and the number of data packets received, we may determine whether the packet delivered is normal or anomalous. The following two scenarios are reviewed to see if the data transmitted is normal.

– Rule-I

If the node is the cluster head, the value IS_CH=1 is assigned. This node then plays an advertisement with A_SNT=1 and A_RCD values of 0 at the same time. This node receives join requests from other nodes, such as J_RCD, after sending the ad, and then sends the SC_ST schedule message. Finally, some values of D_RCD and D_SNT were modified and the sending code received null values.

– Rule-II

We have set the variable IS_CH=0 if the node is not the cluster head. This node will then receive an ad with value A_RCD=and value A_SNT zero at the same time. This node sends J_SNT requests to other nodes after receiving an advertisement, and then sends an SC_RD schedule message. Finally, we looked at

some values for D_SNT and sent some code with D_SNT set to null. It can be said that the data sent is of typical type if the system can satisfy the above two scenarios. Table 2 depicts both normal cases.

Table 2. Rules for normal attack type

RULES	ID	IS_CH	WHO_CH	A_SNT	A_RCD	J_SNT	J_RCD	SC_ST	SC_RD	D_SNT	D_RCD	SEND_CODE	ATTACK_TYPE
Rule-I	101010	1	101010	1	0	0	30	1	0	0	1230	0	Normal
Rule-II	101025	0	101031	0	4	1	0	0	1	111	0	2	Normal

3.2. Blackhole

The attacker node gets data from the source node and does not forward it to the target node in the black hole [19]. All packets are simply dropped by the attacking node [20]. The attacker will reveal his or her identity as a cluster leader at the start of the first round in this type of attack (i.e CH). The attacker's node (i.e. CH) wishes to send data packets to the base station [21]. The attacker, on the other hand, is playing the part of the CH, and instead of delivering these data packets to the base station, the attacker will drop them [22]. The black hole attack can be identified by looking at the values of a few metrics, such as the number of advertising broadcast messages sent and received, the number of join request messages sent and received, and the number of data packets received [23]. The following two scenarios are reviewed to determine whether the data transmitted is normal or not.

– Rule-I

If we set IS_CH=1 for a node, it will broadcast advertising messages to other nodes as A_SNT=1 while also receiving advertising requests as A_RCD=3. We have assumed that the value of J_SNT is null in this case, thus J_SNT=0. J_RCD=93 is non-null, and its value is non-null. The node will then transmit SC_ST=1 scheduling messages and receive D_RCD=1302 data messages. Last but not least, the transmit code value of 0.

– Rule-II

In this case, the node is not the group leader; however, the same node claims to be the group leader. As a result, we have set this node's value to IS_CH=1. Then, like A_SNT=1, this node will broadcast advertising messages to another node while also receiving ad requests from other nodes. another node like A_RCD=3. We have taken into account the values of J_SNT and J_RCD, both of which are zero because J_SNT=0 and J_RCD=0, respectively. D_RCD and send code are both considered null in this case, because D_RCD=0 and send code=0. Table 3 depicts both examples for the blackhole category.

Table 3. Rules for blackhole attack type

RULES	ID	IS_CH	WHO_CH	A_SNT	A_RCD	J_SNT	J_RCD	SC_ST	SC_RD	D_SNT	D_RCD	SEND_CODE	ATTACK_TYPE
Rule-I	102041	1	102041	1	3	0	93	1	0	0	1302	0	Blackhole
Rule-II	109029	1	109100	1	3	0	0	0	0	0	0	0	Blackhole

3.3. Gray hole

The attacker node in the gray hole receives data from the source node, but it is not sent to the target node [16]. The attacker node just drops some packets [24]. The following two scenarios are reviewed to determine whether the data transmitted is normal or not.

– Rule-I

If we set IS_CH=1 for a node, it will broadcast advertising messages to other nodes as A_SNT=1 while also receiving advertising requests as A_RCD=5. We have assumed that the value of J_SNT is null in this case, thus J_SNT=0. J_RCD=0, and its value is null. Then, as SC_ST=0, all other values are treated as scheduling messages, and D_RCD=0 is used to receive data messages. Finally, the transmit code value is 0.

– Rule-II

In this case, the node is not the cluster head, but it reports itself as such. As a result, we set this node's value to IS_CH=1. Then, similar to A_SNT=1, this node will broadcast advertising messages to other nodes while also receiving ad requests from other nodes. another node with the value A_RCD=26 We considered J_SNT and J_RCD values, both of which are zero because J_SNT=0 and J_RCD=0. Because D_RCD=0 and send code=0, D_RCD and send code are both considered null in this case. The gray hole category is represented in both cases in Table 4.

Table 4. Rules for gray hole attack type

RULES	ID	IS_C H	WHO CH	A_SNT	A_RCD	J_SNT	J_RCD	SC_ST	SC_RD	D_SNT	D_RCD	SEND_ CODE	ATTACK TYPE
Rule-I	301032	1	301100	1	5	0	0	0	0	0	0	0	Gray hole
Rule-II	107029	1	107029	1	6	0	25	1	0	0	1200	0	Gray hole

3.4. Flooding

Basically, a fraudulent message is generated in a flooding attack to increase network traffic in order to consume server or network resources [25]. The following two scenarios are used to determine if the data transmitted is normal or not.

– Rule-I

We have defined the variable IS_CH=1 if the node is the cluster head. Then, with A_SNT=60 and A_RCD null, the node will broadcast an announcement message. This node receives join requests from other nodes such as J_RCD=90 after sending the ad and then sends the SC_ST schedule message. Finally, different values for D_SNT=1350 and D_RCD=15 were considered and the passcode was given null.

– Rule-II

We have defined the variable IS_CH=1 if the node is the cluster head. Then, with A_SNT=37 and A_SNT not zero, the button will broadcast an announcement message. This node receives join requests from other nodes such as J_RCD after broadcasting advertisement messages and then sends SC_ST scheduling message. Finally, we looked at some values for D_SNT=238, D_SNT=238 and submitted the code. Table 5 depicts both flooding cases.

Table 5. Rules for flooding attack type

RULES	ID	IS_C H	WHO CH	A_SNT	A_RCD	J_SNT	J_RCD	SC_ST	SC_RD	D_SNT	D_RCD	SEND_ CODE	ATTACK TYPE
Rule-I	102009	1	102009	60	0	0	90	0	1	1350	15	0	Flooding
Rule-II	102034	1	102034	37	28	0	0	0	1	238	238	0	Flooding

3.5. Time division multiple access (TDMA)

For shared medium networks, one form of channel access strategy is time division multiple access [26]. By separating the signals into distinct time slots, TDMA allows multiple users to share the same frequency channel [27]. Table 6 depicts both examples for the TDMA category. Table 7 depicts the details of a packet that was sent normally utilising our proposed technique. The details of the attacks avoided by our proposed algorithm are shown in Table 8. All of the mentioned types of attacks are blocked by applying our proposed technique, as shown in Table 8.

Table 6. Rules for TDMA attack type

RULES	ID	IS_C H	WHO CH	A_SNT	A_RCD	J_SNT	J_RCD	SC_ST	SC_RD	D_SNT	D_RCD	SEND_ CODE	ATTACK TYPE
Rule-I	101075	1	101075	1	0	0	46	46	0	0	0	0	TDMA
Rule-II	102039	1	102039	1	6	0	14	14	0	0	0	0	TDMA

Table 7. Samples of attack type(normal)

ID	IS_C H	WHO CH	A_SNT	A_RCD	J_SNT	J_RCD	SC_ST	SC_RD	D_SNT	D_RCD	D_SNT ENT- TO-BS	SEND_ CODE	ATTACK TYPE
101001	0	101044	0	4	1	0	0	1	38	0	0	4	Normal
101002	0	101010	0	4	1	0	0	1	41	0	0	3	Normal
101003	0	101044	0	4	1	0	0	1	38	0	0	4	Normal
101004	0	101010	0	4	1	0	0	1	41	0	0	3	Normal
101005	0	101010	0	4	1	0	0	1	41	0	0	3	Normal
101006	0	101044	0	4	1	0	0	1	38	0	0	4	Normal
101007	0	101010	0	4	1	0	0	1	41	0	0	3	Normal
101008	0	101044	0	4	1	0	0	1	38	0	0	4	Normal
101009	0	101000	0	4	1	0	0	1	48	0	0	1	Normal
101010	1	101010	1	0	0	30	1	0	0	1230	41	0	Normal

Table 8. CNN based attack prevention

ID	IS_C H	A_SNT	A_RCD	J_SNT	J_RCD	SC_ST	SC_RD	D_SNT	D_RCD	SEND_C ODE	ATTACK TYPE
1702058	1	1	9	0	0	0	0	0	0	0	Normal
211069	1	1	11	0	0	0	0	0	0	0	Gray hole
405018	1	1	10	0	4	4	0	0	0	0	TDMA
602009	1	1	26	0	0	0	0	0	0	0	Blac khole
602094	1	1	26	0	0	0	0	0	0	0	Grayhole
604002	1	1	27	0	0	0	0	0	0	0	Gray hole
404077	1	1	26	0	0	0	0	0	0	0	Gray hole
906045	1	1	23	0	0	0	0	0	0	0	Gray hole
213071	1	1	5	0	4	4	0	0	0	0	TDMA
206021	1	1	2	0	5	1	0	0	676	0	Blackhole
103058	1	1	5	0	32	1	0	0	1248	0	Blackhole
213086	1	1	7	0	15	15	0	0	0	0	TDMA

To evaluate the performance of the proposed model, the confusion matrix is shown in Table 9. Certain metrics like accuracy, precision, recall, and F1 score can be used to compare the performance of the classifiers. Different parameters can be calculated using a confusion matrix as a starting point. A confusion matrix can be used to track the number of cases that are correctly or incorrectly predicted by a classification model. In the confusion matrix, all the labels are considered namely, normal, blackhole, flooding, grayhole and TDMA. The parameters considered are precision (P), recall (R) and F1-score (F1). The average accuracy for the proposed model is 97%.

Table 9. Confusion matrix

Attack Type	Precision (P)	Recall (R)	F1-Score (F1)
Normal	99	99	99
Flooding	73	50	59
Gray Hole	89	81	85
Tdma	63	93	75
Blackhole	94	90	92
Accuracy	97	97	97

The Figure 3 depicts the visual illustration of attacks that CNN was able to prevent. In Figure 3 the precision (P), recall (R), and f1-score (F1) for attacks prevented using CNN are presented and the accuracy of prevention is also mentioned such as TDMA (63%), flooding (73%), gray hole (89%), black hole (94%) and lastly, normal (99%).

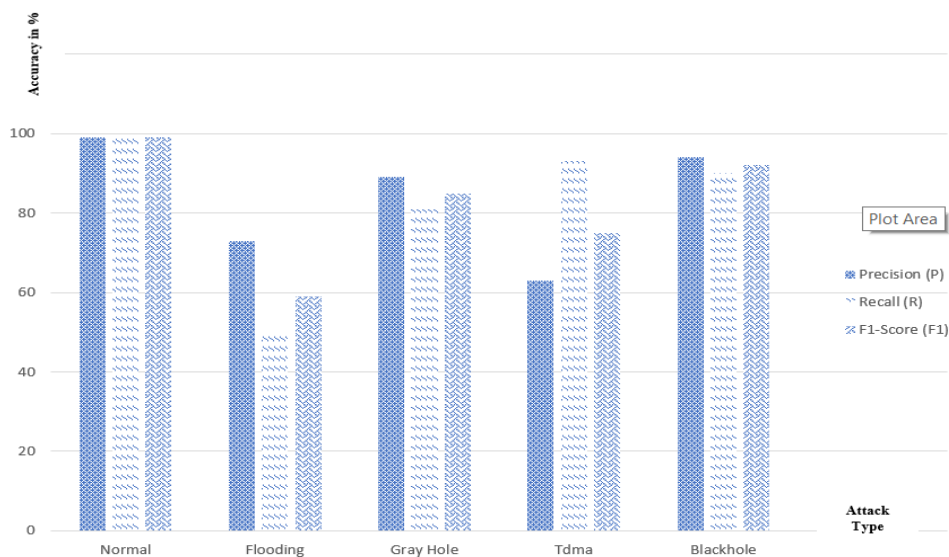


Figure 3. Graphical representation of CNN based attack prevention

Figure 4 depicts the results of attacks that were avoided by CNN with varying data sizes. The results of accuracy, recall, and F1 scores for varying data sizes are shown in Figure 4. The recommender model is evaluated on the same data set each time, despite the size of the set data is different. Based on the results, we can infer that the accuracy is constant regardless of the size of the data set, which suggests that the proposed model is not affected by the size of the data.

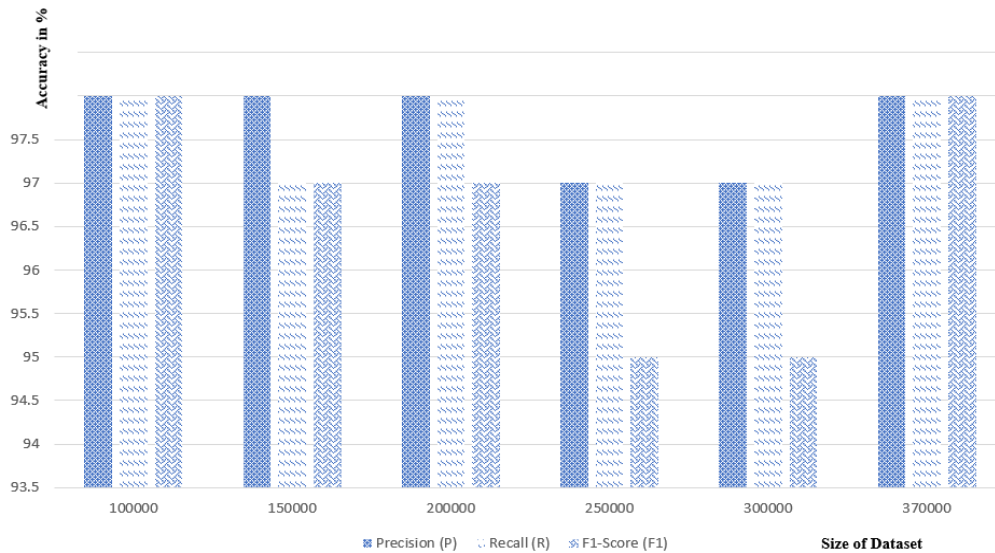


Figure 4. Attacks prevented using CNN with varying data size

4. CONCLUSION

In this paper, CNN algorithm is implemented to detect and prevent intrusion for WSN. This study employs the CNN algorithm to identify and prevent intrusion in WSNs. The suggested system includes two parts, intrusion detection and prevention. A WSN-DS dataset is fed into our system for implementation purposes. At the time of experimentation, a dataset of the same size as well as a dataset of varying sizes is used, and the results of both experiments are presented, indicating that the intrusion detection accuracy is improved. Based on the findings, we can conclude that changing the size of a dataset has no effect on the accuracy of our algorithm. As a result, CNN algorithms are more effective at detecting and preventing intrusion in WSN. The system is designed to use the CNN classifier for a WSN to detect and prevent attacks and the accuracy of our proposed IPS is 97% which is better than the existing system.





REFERENCES

- [1] P. R. Chandre, P. N. Mahalle, and G. R. Shinde, "Machine learning based novel approach for intrusion detection and prevention system: a tool based verification," in *2018 IEEE Global Conference on Wireless Computing and Networking (GCWCN)*, Nov. 2018, pp. 135–140, doi: 10.1109/GCWCN.2018.8668618.
- [2] P. R. Chandre, P. N. Mahalle, and G. R. Shinde, "Deep learning and machine learning techniques for intrusion detection and prevention in wireless sensor networks: comparative study and performance analysis," in *Lecture Notes in Networks and Systems*, Springer Singapore, 2020, pp. 95–120.
- [3] P. N. Mahalle, N. Rashmi Prasad, and R. Prasad, "Object classification based context management for identity management in internet of things," *International Journal of Computer Applications*, vol. 63, no. 12, pp. 1–6, Feb. 2013, doi: 10.5120/10515-5486.
- [4] N. Dey, S. Wagh, P. N. Mahalle, and M. S. Pathan, Eds., *Applied machine learning for smart data analysis*. First edition. | New York, NY : CRC Press/Taylor & Francis Group, 2019. | Series: Computational Intelligence in Engineering Problem Solving: CRC Press, 2019.
- [5] L. Mohammadpour, T. C. Ling, C. S. Liew, and C. Y. Chong, "A convolutional neural network for network intrusion detection system," *Proceedings of the APAN-Research Workshop 2018*, 2018.
- [6] S. Sharma, P. Zavarisky, and S. Butakov, "Machine learning based intrusion detection system for web-based attacks," in *2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*, May 2020, pp. 227–230, doi: 10.1109/BigDataSecurity-HPSC-IDS49724.2020.00048.
- [7] S. Pandey and V. Singh, "Blackhole attack detection using machine learning approach on MANET," in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Jul. 2020, pp. 797–802, doi: 10.1109/ICESC48915.2020.9155770.
- [8] X. Tan *et al.*, "Wireless sensor networks intrusion detection based on SMOTE and the random forest algorithm," *Sensors*, vol. 19, no. 1, p. 203, Jan. 2019, doi: 10.3390/s19010203.
- [9] S. Kumar, A. Viinikainen, and T. Hamalainen, "Machine learning classification model for network based intrusion detection





- system,” in *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*, Dec. 2016, pp. 242–249, doi: 10.1109/ICITST.2016.7856705.
- [10] A. I. Al-issa, M. Al-Akhras, M. S. ALSahli, and M. Alawairdhi, “Using machine learning to detect DoS attacks in wireless sensor networks,” in *2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)*, Apr. 2019, pp. 107–112, doi: 10.1109/JEEIT.2019.8717400.
- [11] A. Shenfield, D. Day, and A. Ayes, “Intelligent intrusion detection systems using artificial neural networks,” *ICT Express*, vol. 4, no. 2, pp. 95–99, Jun. 2018, doi: 10.1016/j.ict.2018.04.003.
- [12] S. M. Sohi, J.-P. Seifert, and F. Ganji, “RNNIDS: Enhancing network intrusion detection systems through deep learning,” *Computers and Security*, vol. 102, p. 102151, Mar. 2021, doi: 10.1016/j.cose.2020.102151.
- [13] C. J. Ugochukwu, E. O. Bennett, and P. Harcourt, “An intrusion detection system using machine learning algorithm,” vol. 4, no. 1, pp. 39–47, 2018.
- [14] W. Elmasry, A. Akbulut, and A. H. Zaim, “Evolving deep learning architectures for network intrusion detection using a double PSO metaheuristic,” *Computer Networks*, vol. 168, p. 107042, Feb. 2020, doi: 10.1016/j.comnet.2019.107042.
- [15] M. S. Hoyos LI, G. A. Isaza E, J. I. Vélez, and L. Castillo O, “Distributed denial of service (DDoS) attacks detection using machine learning prototype,” in *Distributed Computing and Artificial Intelligence, 13th International Conference*, Springer International Publishing, 2016, pp. 33–41.
- [16] I. Almomani, B. Al-Kasasbeh, and M. AL-Akhras, “WSN-DS: a dataset for intrusion detection systems in wireless sensor networks,” *Journal of Sensors*, vol. 2016, pp. 1–16, 2016, doi: 10.1155/2016/4731953.
- [17] S. Babar, P. Mahalle, A. Stango, N. Prasad, and R. Prasad, “Proposed security model and threat taxonomy for the internet of things (IoT),” in *Recent Trends in Network Security and Applications*, Springer Berlin Heidelberg, 2010, pp. 420–429.
- [18] T. Liu, S. Fang, Y. Zhao, P. Wang, and J. Zhang, “Implementation of training convolutional neural networks,” Jun. 2015, [Online]. Available: <http://arxiv.org/abs/1512.07108>.
- [19] J. Gu *et al.*, “Recent advances in convolutional neural networks,” Dec. 2015, [Online]. Available: <http://arxiv.org/abs/1512.07108>.
- [20] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” Aug. 2016, [Online]. Available: <http://arxiv.org/abs/1608.06993>.
- [21] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” 2010.
- [22] A. A. Alurkar *et al.*, “A comparative analysis and discussion of email spam classification methods using machine learning techniques,” in *Applied Machine Learning for Smart Data Analysis*, First edit., New York, NY :CRC Press/Taylor and Francis Group, 2019. | Series: Computational Intelligence in Engineering Problem Solving: CRC Press, 2019, pp. 185–206.
- [23] P. N. Mahalle, B. Anggorojati, N. R. Prasad, and R. Prasad, “Identity authentication and capability based access control (IACAC) for the internet of things,” *Journal of Cyber Security and Mobility*, Feb. 2013, doi: 10.13052/jcsm2245-1439.142.
- [24] U. Ghugar and J. Pradhan, “A study on black hole attack in wireless sensor networks,” 2016.
- [25] A. Dhaka, A. Nandal, and R. S. Dhaka, “Gray and black hole attack identification using control packets in MANETs,” *Procedia Computer Science*, vol. 54, pp. 83–91, 2015, doi: 10.1016/j.procs.2015.06.010.
- [26] N. Patani and R. Patel, “A mechanism for prevention of flooding based DDoS attack,” *International Journal of Computational Intelligence Research*, vol. 13, no. 1, pp. 101–111, 2017.
- [27] M. Manzo, T. Roosta, and S. Sastry, “Time synchronization attacks in sensor networks,” in *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks-SASN '05*, 2005, p. 107, doi: 10.1145/1102219.1102238.

BIOGRAPHIES OF AUTHORS







Pankaj R. Chandre     has obtained his B.E degree in Information Technology from Sant Gadge Baba Amravati University, Amravati, India and M.E. degree in Computer Engineering from from Mumbai University Maharashtra, India in the year 2011. Currently he is pursuing his PhD in Computer Engineering from Savitribai Phule Pune University, Pune, India. He is currently working as an Assistant Professor in Department of Computer Science and Engineering, MIT School of Engineering, MIT ADT, Pune, India. He has published 60 plus papers at international journals and conferences. He has guided more than 30 plus under-graduate students and 20 plus postgraduate students for projects. His research interests are Network Security and Information Security. He can be contacted at email: pankajchandre30@gmail.com.



Dr. Parikshit N. Mahalle     has obtained his B.E degree in Computer Science and Engineering from Sant Gadge Baba Amravati University, Amravati, India and M.E. degree in Computer Engineering from Savitribai Phule Pune University, Pune, India. He completed his Ph. D in Computer Science and Engineering specialization in Wireless Communication from Aalborg University, Aalborg, Denmark. He has more than 18 years of teaching and research experience. He has been a member board of studies in computer engineering, Savitribai Phule Pune University (SPPU), Pune, India. He has been a member-Board of studies in computer engineering, SPPU. He is member-BoS coordination committee in computer engineering, SPPU. He is also serving as member- Technical committee, SPPU. He is IEEE member, ACM member, Life member CSI and Life member ISTE. He can be contacted at email: aalborg.pnm@gmail.com.



Dr. Gitanjali Shinde     has obtained her B.E. degree in Computer Engineering from Pune University, Maharashtra, India in 2006 and received Master's degree in computer Network from Pune University in 2012. In 2018 awarded PhD in Wireless Communication, Department of Electronic Systems, Center for Communication, Media and Information Technologies Copenhagen, Aalborg University Denmark. She is working as an Assistant Professor in Department of Computer Engineering, BRAC's Vishwakarma Institute of Information Technology, Pune, India. She has published 25+ papers at National and International level. She received research funding from Board of College and University Development, SPPU, Pune India. She can be contacted at email: gr83gita@gmail.com.