Wireless Personal Communications (2022) 126:911–931
https://doi.org/10.1007/s11277-022-09776-x

Intrusion Detection System in Wireless Sensor Network Using Conditional Generative Adversarial Network

Tanya Sood[1] · Satyartha Prakash[2] · Sandeep Sharma[3] · Abhilash Singh[4] · Hemant Choubey[3]

Accepted: 7 May 2022 / Published online: 31 May 2022

Abstract
Wireless communication networks have much data to sense, process, and transmit. It tends to develop a security mechanism to care for these needs for such modern-day systems. An intrusion detection system (IDS) is a solution that has recently gained the researcher's attention with the application of deep learning techniques in IDS. In this paper, we propose an IDS model that uses a deep learning algorithm, conditional generative adversarial network (CGAN), enabling unsupervised learning in the model and adding an eXtreme gradient boosting (XGBoost) classifier for faster comparison and visualization of results. The proposed method can reduce the need to deploy extra sensors to generate fake data to fool the intruder 1.2–2.6%, as the proposed system generates this fake data. The parameters were selected to give optimal results to our model without significant alterations and complications. The model learns from its dataset samples with the multiple-layer network for a refined training process. We aimed that the proposed model could improve the accuracy and thus, decrease the false detection rate and obtain good precision in the cases of both the datasets, NSL-KDD and the CICIDS2017, which can be used as a detector for cyber intrusions. The false alarm rate of the proposed model decreases by about 1.827%.

Keywords Wireless sensor networks · Deep learning · GANs · XGBoost · Security · IDS · Confusion matrix

1 Introduction

Over the years, wireless networks have been used in a wide array of applications like—outdoor and indoor monitoring applications, communication, and internet services. There is a wide variety of data being transmitted in these deployed networks. This variety of networks in today's scenario comes with a dire need for security enhancements because of the several different types of attacks and security breaches possible in today's world. With the continuously increasing data transmission between various entities and networks, we also need suitable network protection strategies. There have been various security methods

✉ Sandeep Sharma
sandeepsvce@gmail.com

Extended author information available on the last page of the article

implemented and adapted in the networks until now, like cryptographic systems [1], base system protection [2], location verification [3], intrusion verification and detection [4], doubly near-far problem [5], and so on. The networks these days have large amounts of data and information to be processed; this leads to the drainage of their energy and memory capacities. Wireless information and data are simultaneously transferred in wireless networks to decrease the battery drain problem [6]. The security mechanism chosen for these networks may lead to more energy and memory consumption, leading to the failure of sensors, failure of a network entity, loss of data, and thus, a network failure altogether. A solution to these limitations may lie in Intrusion Detection Systems.

An intrusion detection system (IDS) is a security application that recognizes the security breaches by outsiders and insiders in a system. An IDS can thus be understood as a wholesome entity that monitors the behavior of a system and responds to the abnormalities in its functioning. Modern-day systems deal with a vast amount of data, and thus, developing more intelligent and platform-friendly solutions is necessary. The security systems should be flexible in their operations across various platforms and provide maximum security. Thus, a viable solution to develop such an entity that tackles these present-day problems is the integration of machine learning to ease the process of monitoring and securing the wireless networks and take care of its limitations.

Wireless Sensor Networks (WSNs) consist of low-power sensors capable of sensing and transmitting information through wireless channels. These are battery constraint devices, and one must use them efficiently; otherwise, if the battery is drained, the node is dead, and a dark zone with no network coverage occurs. The WSNs find applications in various filed like data aggregation, information sensing, environment monitoring, digital agriculture, smart agriculture, remote sensing, healthcare, and military applications like border surveillance [7, 8]. There is always no man's land between the line of control (LOC) between any two nations. There are army personnel and troops deployed to patrol their areas, but monitoring humans is not always feasible. Any intruder crossing these crucial areas can harm the army deployment and arrangements, causing great harm to the nation's security. Wireless sensor networks are deployed to have the best possible surveillance against intrusion to strengthen detection in vulnerable border areas [9].

For securing such fields, a typical communication system is to be established. Such a scenario requires adding newer entities to the networks and hence generates a massive amount of data in those networks with the communicating entities. Monitoring in these areas is continuously done without any interruption, and hence there exists a fast-growing data exchange scenario. The challenge here is to develop a solution to increase data transmission speeds and modernize how the entities communicate. *Security* is an equally important aspect other than the communication system and needs to be addressed. Network security is an aspect that takes care of the network and its services altogether. It protects against unauthorized access to unapproved network modifications and maintains the fundamental integrity and confidentiality between the users to protect the data exchanged within the network. Various security measures have been adopted to secure the wireless sensor networks from unwanted threats like cryptography, code-testing techniques, secure location, secure routing techniques, etc. [2]. However, measures like a key exchange, firewalls, antivirus, network analyzers, etc. [10] have worked well in the past, but these models prove incompetent in more extensive networks.

Moreover, the previous models, like cryptographic key exchange systems, can be cumbersome when applied to a more extensive sensor network. They would exploit the sensor's limited memory and energy capacity, and key management can be a problematic task [11]. Thus, we need to use a solution to monitor a scalable network of any size at a wholesome

level [12]. One viable solution for such problems is the Intrusion Detection System (IDS). It is a network security measure that takes over the security of the hardware and the software involved in a network [13]. It aids in the overall protection of the network and provides us with a much easier solution. The IDS can monitor the total traffic transmitted to the network and keep track of the system logs. Whenever it observes an abnormality or vulnerability in the system, it alerts the administrator. It eases the complexity of the network and thus is more comfortable in monitoring and maintenance. The use of AI technologies [14] and fuzzy-based decision tree mechanism [15, 16] in building such IDSs is a growing area of research, as it can ease human intervention in network monitoring tasks [17].

Recently, drones have been deployed to secure premises of high-security zones where the open challenge is to predict the drone's flight's direction of arrival (DoA) [18]. However, when we talk about a sensor-based system, the sensors are deployed randomly in clusters which are formed based on any fuzzy-based [19], nature-inspired [20] or any other conventional algorithm [21]. The node deployment plays a critical role in the performance of the system. Node localized at the right place and with minimum localization error [22] substantially boosts the system's performance. If a node is not appropriately placed, potential intruders can enter the system and alleviate the attacks in the sensor network deployed system. It is countered by choosing a trust-based system [23, 24]. Any IDS detects a suspicious activity with the help of any technique such as machine learning [25] and deep learning, which trains the system against various scenarios. Then with the help of the trained scenarios, any intrusion can be detected [26]. The deployed sensor nodes require a mobility model for random movement in the region of interest. The mobility model is selected in such a way as to decrease energy consumption and enhance the QoS of the system [27].

Author's Contribution: In this paper, we propose one such IDS, based on Deep Learning methods, which can train itself to distinguish between normal and attack class data. The proposed method prevents the administrator from supervising network traffic and save energy and time. The system would detect the fraud data itself and alert any attacks occurring in the network when combined with an alerting system. This model can reduce the number of sensor nodes in the sensor network. Moreover, it can reduce the need to deploy extra sensor nodes to create adversary data to confuse the attacker. Our proposed system is also capable of generating more fake data and thus, can help reduce the size of the sensor network.

Paper Organization Section 2 talks and sums up the referred literature regarding the various previous developments that have taken place in this area of research. Section 3 deals with the proposed system model, the IDS algorithm, and the system's basic entities. We summarize the simulation parameters and discuss the inferences drawn from the obtained results in Sect. 4. The paper is concluded in Sect. 5 with its future scope.

2 Related Works

The security of wireless networks has been an open challenge for researchers over the years. Technological advancements have aided developers and researchers in finding and implementing newer security solutions. However, it has also led the attackers to crack the traditional techniques more easily. For example, ransomware attacks affect a vast number of systems worldwide. Therefore, it created a necessity to develop newer and more technologically advanced models to tackle such attacks.

Previously, many methods have been used to develop the IDSs. Although the various deep learning methods introduced have an excellent accuracy measure, the methods used have their drawbacks, which may prove to be constraints in the application phase. The authors [28] introduced an IDS using the Support Vector Machine (SVM) algorithm and Deep Learning on the Port Scan data, similar to one part of our experiment. However, the deep learning model depicts good accuracy over the SVM model. The SVM method generally faces disadvantages in dealing with more massive datasets, which might pose an issue in applying more extensive networks and more enormous datasets. In [29], the author talks about how the deep learning techniques have established their efficacy in ML and intrusion detection. In the image classification tasks, the systems might have discrepancies that benefit the attackers. It might lead to having doubts about deploying the deep learning networks in the intrusion detection field. The author has researched these deep learning algorithms against excellent attack algorithms like JSMA, DeepFool, FGSM, etc., on the NSL-KDD dataset. The author did not work upon transferability with different inputs in the same neural network, between the different NNs, or with well-known ML techniques like SVM. In [30], the authors propose SVM and Autoencoders to introduce a Network Intrusion Detection System (NIDS) based on Self Taught Learning (STL). They manage to obtain good classification accuracy and compare it to various other methods. However, the autoencoders are not very competent in data regeneration; their performance is not very satisfactory in this respect. The GANs can perform better when compared to autoencoders. Deep neural networks (DNN) study various datasets [31]. The framework is a multilayered hybrid DNN that is to be used as an IDS. The results are reasonably accurate in all the datasets. However, the stability of the results is not constant in the datasets, which can be used to further improve the results by using other deep learning methodologies.

This paper is similar to [32] in using a GAN model for developing a security system. The authors presented a sensor network security technique, which was accurate. However, our work is different as we have developed a detection system based on the CGAN model, which is a better way to learn from classed data. Also, our model uses XGBoost for visualization of results, unlike [32], which required a cross-platform working effort.

Further, our work shows an increase in the accuracy and precision of the output and works on a faster and lesser complex model. It can work on a single platform for all the tasks, from preprocessing the data to visualization results. The results thus obtained are much faster and decrease the computational time. The proposed IDS has shown a significantly lower false detection rate, which has been one of the constraints of the Intrusion Detection Systems in general. The following section discusses the algorithm involved in the proposed work and the concepts.

3 System Model

Before we discuss the system model, we need to know about the basics of our model—GAN, CGAN, and XGBoost. The data is first taken in and trained by the CGAN model and then passed on to the XGBoost classifier to validate the obtained results. Therefore, we first shed light on the problem we aim to fix, explain these basic concepts, and discuss our proposed model.

3.1 Problem Statement

This work focuses on a new unsupervised learning algorithm and its application to secure IDS. This framework will produce fake data to confuse the attacker. It can secure the network and transmit data between the sender and the receiver, compared to the other approaches of IDS developed using Deep Learning. This technique eliminates the need for fake sensor nodes or fake data-producing entities, increasing the network's energy and memory consumption. The two frameworks discussed in the following subsections and considered after the literature survey have been found to have limited research on the possible applications of the CGAN and XGBoost. Thus, we take on the task of presenting an amalgamated framework of the two algorithms and studying their results.

3.2 Generative Adversarial Networks (GAN)

A 'GAN' is a neural network that can create new data without any prerequisites or from scratch. In the original formulation of GANs, proposed by Ian Goodfellow et al. in [33], the discriminator network model generates an estimated probability that if a given image/data was real or generated. The discriminator would then have access to both the generated and actual data, which would lead it to generate the estimate about both types of inputs. The discrepancy between the discriminator's results and the actual values is calculated as the loss. The G network is intended to get familiar with the organization of the training data. In contrast, the D network is intended to find the similarity of the data coming from the training (real) data instead of the generator data (attack/fake).

Figure 1 is a representation of the GAN's working. As we can see from the figure, the noise input (z) is taken and mapped into a latent space, usually real data size. Then, this noise is input into the Generator (G) and processed according to the network conditions defined. This output 'test' data becomes the input to the discriminator (D), which compares the real data (x) and the Generator's test data, and then gives the output. The output of D is in the form of weights assigned to the results, like 0 for 'fake'data and 1 for 'real'data. This output is then fed back to the G network and D network. G adheres to $D's$ feedback to improve the generated data, and this process goes on for the defined

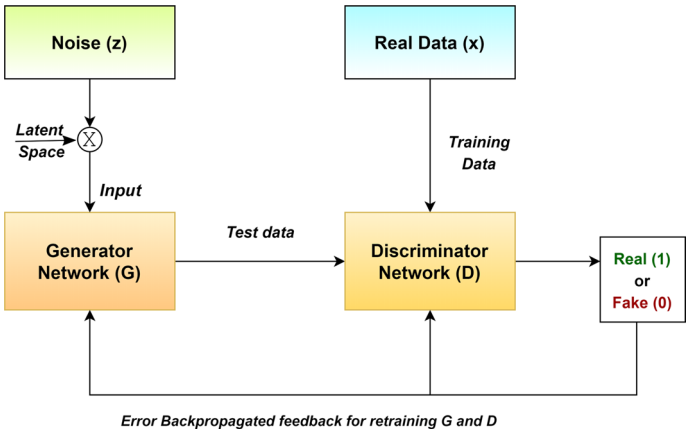

Fig. 1 Two model training in GAN

number of steps. These *G* and *D* networks work in union to learn from the feedback and thus, be able to generate new data [34].

The generator's capacity to create new data that resembles genuine examples is improved. The perception is to bewilder the intruder/ attacker and keep them from differentiating between the data from the generator and the valid data from the datasets. The Discriminator differentiates between real and fake data [32] by enhancing the probability of the valid data to '1'and limiting the probability of fake samples being '0'.

- Conditional Generative Adversarial Network (CGAN)

  Conditional GAN is a variant of GAN, a Machine Learning structure for the training of the generative models. The authors introduced this concept in [35]. The CGAN architecture allows the generator and discriminator to train themselves with supplementary information, like class labels or other data. There should be some extra data or information over which the generator and discriminator should be fed to train the model better. The conditional training can be done in a CGAN if we provide this extra data to the generator and the discriminator. These work well for the data where we have conditional requirements, and it works well with data like text and images [36]. This model thus helped us in developing our IDS. The functioning of the CGAN is not different from the underlying GAN and differs only in the selection of data to be input. The basic structure of the CGAN, as proposed in [35], is shown with some modifications in Fig. 2.

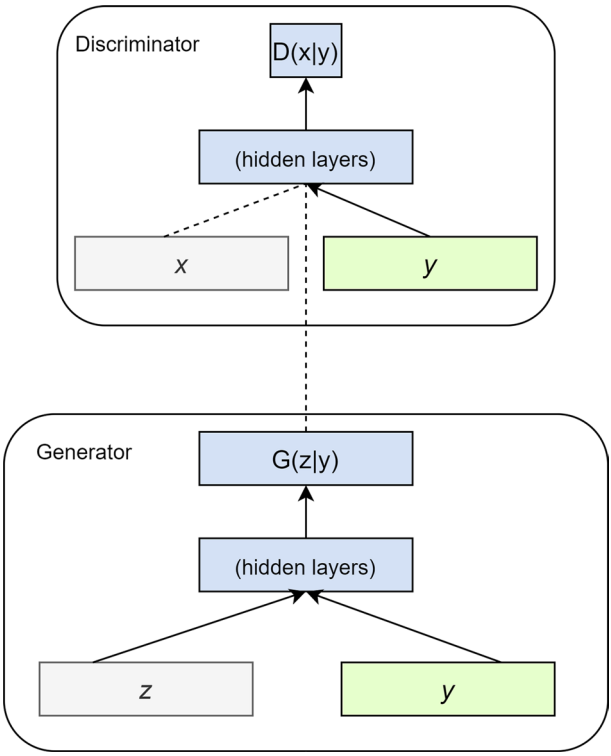

Fig. 2 CGAN model

3.3 eXtreme Gradient Bossting Algorithm (XGBoost)

XGBoost is a decision-tree-based Machine Learning (ML) algorithm that uses a gradient-boosting algorithm structure. In prediction problems or challenges, including unstructured information, Artificial Neural Systems will, in general, beat every other algorithm or structure [37]. Nonetheless, about little to medium organized/tabular data, decision-tree-based calculations are viewed as top-tier. XGBoost is a sparsity-aware algorithm for irregular/sparse data for an estimated tree-based learning algorithm [38]. It is a scalable framework that supports the majority of scenarios.

It is also a preferred classification model due to the following advantages that it offers [39].

- It is about ten times faster than the previous classification methods.
- It enables parallel processing by using various cores for the processing.
- Cross-validation is a parameter that already exists in the framework, and there is no need to install any external package for it.
- It can deal with missing values, unlike Logical Regression.
- XGBoost can deal with over-fitting problems and also flooding of data as happens in case of DDoS attacks.
- Tree pruning, i.e., reducing the size of a tree by removing the parts that offer none or significantly fewer instances for classification, is done till the maximum depth of the tree and does not only terminate after achieving a negative gradient.
- The user can define various evaluation metrics and an objective function of choice.

3.4 Datasets

The two datasets studied, chosen, and considered for experimenting with to study the functioning of our IDS are mentioned below:

- **NSL-KDD dataset:** The NSL-KDD dataset [40] is one of the many datasets available for cybersecurity study and experimentation and has a variable ratio of attack and standard class data. This dataset is a successor of the KDD99 dataset, which had problems like redundant and duplicate data. The NSL-KDD dataset has 42 attributes, which contain labels such as protocol type, service, and label (attacks/normal). We have selected this dataset to be one of the two datasets for obtaining our IDS results, as this dataset is often used as a benchmark in various studies of IDSs. The training set consists of 67,343 standard class data and 58,630 attack class data.
- **CICIDS2017 dataset:** The CICIDS2017 dataset [41] is another such dataset present for the studying of cybersecurity and developing newer network protection systems. This dataset is up-to-date with the latest attack information. The information labels include source and destination IP addresses and attack labels like DDoS, PortScan, Infiltration, etc. A dataset is a benchmark dataset if it covers 11 criteria, for example, complete traffic, labelled dataset, attack diversity, etc. This dataset has many different datasets labelled with different attacks that have been monitored over various systems. It consists of 78 attribute labels. We have only used one of these datasets to study in our model. We have considered the 'Friday-WorkingHours-Afternoon-PortScan' dataset to study our second batch of data to obtain the system's results. The PortScan attack sends

the client request to several server port addresses that might be present on a host [42] to find a vulnerable active port and then exploit the services it provides. This dataset has a total of 1,27,537 'BENIGN' class data and about 1,58,930 'PortScan' attack class data.

3.5 Work Flow

The system workflow (Fig. 3) depicts the constituent blocks that form the main foundation of our work. First, the dataset is input into the network; we take the NSL-KDD dataset and the CICIDS2017 dataset to our framework. The data is then passed onto pre-processing. At this stage, data is usually unclean i.e. data might have missing values; the data may have classes that the other program may not support. We pre-process the data and convert it into a binary form using data science operations. We check for missing values and even them out using suitable methods. Binarization is the labeling of string data about our data.

After the pre-processing, data is sent to the CGAN model, which uses a Generator and a Discriminator. The Generator is used to generate data samples, while the Discriminator network can be understood as a critic network used to correct the generator outputs.

We then obtain the output of the CGANs' training, compare the generated fraud data with the actual fraud data, and obtain the accuracy measure of this generated data; we move on to the next step in our framework. It is the step where we utilize the XGB Classifier for the data classification and results from visualization. This classifier recognizes the different classes of data present in the dataset, defines them, and provides the visualization results of the same-this classification and visualization of the data aid in understanding the patterns in the data under consideration. Here, the XGBoost Classifier helps us rapidly while not requiring many program run times. This data classification also helps to recognize how efficiently the system generated the fraud data to detect the actual fraud data.

We use some samples of the original fraud data and some of the generated fraud data and use them as inputs to the classifier to obtain accuracy. The data accuracy measured by the classifier is the accuracy of our proposed IDS. We also check the classifier's accuracy, which tells us if the classifier could recognize the data correctly and to what measure.

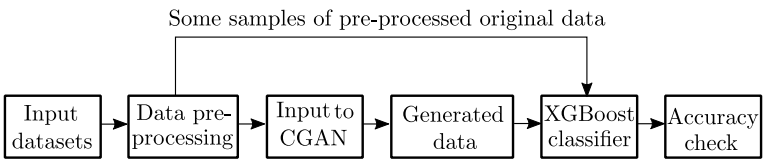

Fig. 3 System workflow

3.6 Proposed IDS Algorithm

The proposed algorithm aims to create an IDS that works faster and better than the previously developed or introduced models.

Algorithm 1: Proposed IDS Working Algorithm

Input: X= Training Data; N_g = noise; M= Mini-batch size; i= Number of iterations
Output: T_s= Testing Data; T_g= Generated Data; F= Fake samples (from generator); A= Accuracy

```
Randomly select samples from X of size M.
In GAN,
for i iterations do
    Take n samples of noise, N_g = n1, n2, ... (for generator)
    Take n samples from real data, X= x1, x2, ... (for discriminator)
    Generator tries to shape n samples of N_g like n samples of X
    Discriminator gets data from generator after completing generator step
    Updates weights of results as 0 or 1 to give feedback to generator
    i ← i-1
Obtain T_g Take m samples to form T_s from X and m samples from T_g
Input T_g and T_s to classifier
Calculate accuracy of T_g based on T_s
return Accuracy 'A'
```

The algorithm begins by introducing the inputs and outputs for the system's working under consideration. These are the fundamental entities that will help build the system as it is supposed to be to achieve our research objectives. The datasets taken in as the inputs to the system are NSL-KDD and CICIDS2017 datasets. The specifications of both datasets, their inclusive entities, etc., have been discussed later. Now, as mentioned in the algorithm, the inputs are taken according to the specifications of the system we want to administrate. The working of the two models, as in the algorithm, is explained below. In the algorithm, input samples from dataset *X* are fed to the CGAN network. These samples are taken according to a mini-batch size, i.e., some samples would be randomly selected according to the number defined for the mini-batch. Then, we define how many steps we need to complete the training for the functioning to start, and this can be tailored according to one's need for refining the results. The number of iterations mentioned is the number of steps the model requires to complete training. The generator network then starts to take noise samples, *n*, in the defined number, and tries to mould them into the form of the real data. After this first step, it passes the data to the discriminator to compare and check with the real data. The discriminator checks this data by the generator and compares it with the real data samples *x*, which are equal to the noise samples. The discriminator gives feedback to the generator about how real or fake this data was, by assigning weights to the results as 0 for fake and 1 for real. The generator takes this input from the discriminator and then repeats the process until the number of steps is completed. The next step is the classifier. The classifier has a certain number of samples (m) from the actual data in the dataset and similar data from the generated output by the CGAN. Then, the classifier compares the two types of data given to it and then provides us with the accuracy of the data generated by the CGAN.

The next step is the classifier. The classifier has a certain number of samples (m) from the actual data in the dataset and similar data from the generated output by the CGAN.

Then, the classifier compares the two types of data given to it and then provides us with the accuracy of the data generated by the CGAN.

The classifier can produce a confusion matrix to tell us about the CGAN's performance and how accurately and precisely it has generated the data. It tells us about the class-wise distribution of data in the created set, i.e., it may reveal whether the samples created that are said to be 'Normal'data belong to the 'Normal'data class, and so on. We will explain the confusion matrix by referring to a dummy matrix in the next section.

4 Simulation Results and Discussions

The proposed model has been implemented using Python, and the experimentation has been carried out using Keras 2.3.1 and Tensorflow 2.1.0. We have supported the scikit-learn, Seaborn, and XGBoost features for evaluation and result extraction. The work has been carried out on a system equipped with Intel(R) Core(TM) i5 CPU@2.5 GHz processor. We used a Python 3 Jupyter Notebook 6.0.3 Platform for the coding and implementation.

We discuss the simulation settings, evaluation parameters, and the results obtained in the upcoming sections.

4.1 Generator Network Parameters

The Generator's function generates fake samples from the dataset samples fed to it. *G* will improve its results based on the feedback received from *D*. We created the Generator with four fully connected layers. Reshaping of the data was done according to the binary class data. We took a mini-batch size of 128 and employed an Adam Optimizer on the model, with a 0.0001 learning rate momentum of 0.9 for 5000 steps. Refer to Table 1 for further details regarding the simulation parameters.

4.2 Discriminator Network Parameters

Discriminator receives inputs from both G and real samples of the dataset and aims to differentiate between them. *G* and *D* are trained while contesting with each other. The learning rate was specified as 0.0001, using the Adam Optimizer, the activation function as *tanh*, mini-batch size was specified as 128, and the values of $\beta 1 = 0.5$ and $\beta 2 = 0.9$. Here,

Table 1 Simulation parameters of generator network

S. no.	Parameter	Description	Value
1.	Mini-batch size	Creates small batches of the entire data of the size defined here	128
2.	Optimizer	Attribute to reduce losses	Adam
3.	Learning rate	Enables to traverse the data slope while being able to cover the data points	0.0002
4.	Activation function	Adds non-linearity to function	tanh
5.	Momentum	Helps in faster convergence of vectors	0.9
6.	Layers in the network	Receiving and processing inputs happens here	4

$\beta 1$ and $\beta 2$ are decay rates of the moving averages of the gradients. $\beta 1$ controls the exponential decay of the moving average for the first moment, whereas $\beta 2$ does the same for the second moment.

4.3 XGBoost Parameters

The XGBoost Classifier is ten times faster than the other decision-tree-based algorithms. It is also accurate and provides excellent results on sparse or tabular data such as ours. In the XGBoost Classifier, various parameters can be used and tuned according to one's own needs and the tasks desired to be performed using the classifier. The parameters we used for our system are listed below, with a short description of their purpose and the value we set for the same.

- *objective*: It is used in defining the loss function that needs to be minimized.
- *max_depth*: It defines the tree-depth. The model gets more complicated with the increase in the tree depth. Bigger models require more considerable tree depth.
- *eval_metric*: It is used to calculate the model's accuracy on the testing data.

The values of these parameters are tabulated in Table 2 for ready reference.

4.4 Confusion Matrix

The confusion matrix is a vital source of information about network performance in any Artificial Intelligence application/algorithm. The basic structure of a binary-class confusion matrix is shown in Fig. 4. There are values distributed over two main classes of data that are actual and predicted values. These values then get divided up to form the four leading constituent labels of the confusion matrix, namely—True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). The 'positive' value in the case of our data would be the 'Normal' class data, and the 'negative' value would be that of the 'Attack' class data. The significance of the four labels is, as explained: True Positive value is assigned to a sample if the sample classified by the model as a positive sample is positive in the dataset, i.e., its actual value is also positive. True Negative is assigned to a sample if the model has classified the sample as a negative sample. It does so due to the negative class in the dataset. A False Positive value occurs when the predicted results differ from the actual results. When the expected value infers a positive value, the real value is a negative class value. The False Negative result is the opposite of the False Positive label. The outcome predicted as unfavourable by the model is the actual class data in the original dataset.

The confusion matrix is responsible for visualizing the output quality of the model. The classification outputs of a dataset shown in a confusion matrix are the vital components of

Table 2 XGBoost Parameters

Parameter	Value
Objective	Binary:logistic
Max_depth	4
Eval_metric	Auc

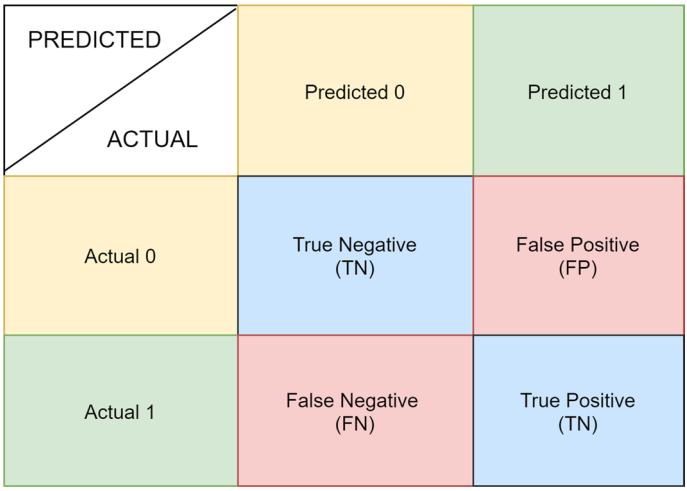

Fig. 4 Confusion matrix

measuring the correctness of the model's predictions. Refer to Fig. 4, which represents the confusion matrix, when studied, shows that the elements in the blue diagonal boxes represent the correct/truly classified labels. The items in the pink boxes represent the incorrectly classified labels by the model. A higher count of values in the diagonal (blue) boxes results in a better confusion matrix of the generation/classification/prediction model, resulting in better model accuracy.

Using the different parameters—TP, TN, FP, and FN, of the confusion matrix, one can calculate various measures related to finding how good the model is a prediction/classification performance was. Several steps help to do so, for instance, Recall, Sensitivity, etc. We are only concerned with calculating the Accuracy and Precision of the model, which are formulated as follows:

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{1}$$

$$Precision = \frac{TP}{(TP + FP)} \tag{2}$$

It measures the accuracy and precision of the model, which we intend to find out to check the correctness of our proposed IDS.

4.5 Results

The model proposes to build an intrusion detection system. The IDS we propose is based on the generative adversarial network algorithm and uses the XGBoost algorithm. We have carried out the experiments for 5000 samples of data from both datasets. We varied the dataset feature sizes to obtain results on the different values of the dataset. The variations in the dataset result in limiting the amount of available data for training and testing, which further affects the training and testing of the model, as it has lesser

Table 3 Confusion matrix for 40 features

	Pred 0	Pred 1
True 0	1176	0
True 1	1	1175

Table 4 Confusion matrix for 30 features

	Pred 0	Pred 1
True 0	353	0
True 1	2	351

Table 5 Confusion matrix for 20 features

	Pred 0	Pred 1
True 0	242	0
True 1	2	240

Table 6 Accuracy and precisions for NSL-KDD data set

Table number	Title	Accuracy (%)	Precision (%)
Table 3	For 40 features	99.95	100
Table 4	For 30 features	99.71	100
Table 5	For 20 features	99.58	100

instances to learn from and lesser instances to compare. Therefore, we change the feature size of the datasets to see the effect of this change and check our system's response to these conditions. We establish whether or not the accuracy and precision of our system are affected by this change and whether the system is stable for these changes. The accuracy is calculated according to Eq. 1 and the precision according to Eq. 2. These results are depicted and discussed below.

- *For NSL-KDD dataset*

  These results are obtained while experimenting with the feature size of the NSL-KDD dataset. We experimented with the feature sizes at the original 40 feature size, 30 feature size, and 20 feature size. The plots for the losses are shown in Fig. 5. These losses depict that our model has converged the losses during the learning process and has reached an optimum value where learning is completed. Therefore, it depicts the stability of the training of our model. The results for the confusion matrix obtained for the original feature size, i.e., 40, can be seen in Table 3. The confusion matrices obtained for the varied feature sizes 30 and 20 can be seen in the Tables 4 and 5, respectively. We have summed up their accuracies and the precision measures in Table 6. We observed minute changes in the accuracies across all the feature sizes and obtained almost accurate predictions in the case of the varied feature sizes. The precision measure, which tells the correctness of the obtained positive classes, is constant. Thus, we could reduce the false detection and have obtained

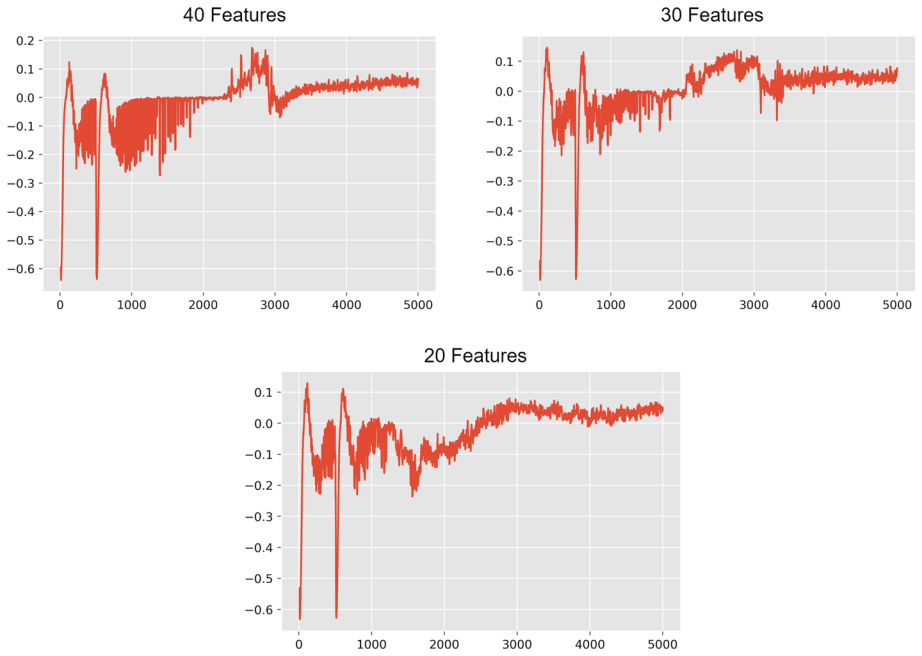

Fig. 5 Losses for NSL-KDD dataset

good accuracy and precision measures for the same. We might, therefore, say that the system performs well and is stable across varied conditions.

- *For CICIDS2017 dataset*

  We particularly considered the 'Friday-Working Hours-Afternoon-PortScan' dataset from the many datasets available in CICIDS2017. We considered the different feature sizes, and as this is a wider and much bigger dataset than NSL-KDD, we took the liberty to experiment with more changes in the feature sizes. Hence, we found the results for feature size 78, feature size 68, feature size 58, and feature size 48. We also depict the losses, and we may observe that the losses have converged to similar values throughout the 5000 step training, and the system has reached an optimum stage. The losses are shown in Fig. 6. The results obtained for the original feature size, i.e., 78, can be seen in Table 7. The confusion matrices obtained from varying the feature sizes to 68, 58 and 48 can be seen in the Tables 8, 9 and 10, respectively. We have mentioned all their accuracies and precision in Table 11. We could observe the changes in the accuracies for this dataset as well. Across all the feature sizes, the accuracy measure was almost accurate, and the false detection was minimal. We could thus, reduce false detection by 1.827% and have obtained good accuracies for the same. We might, therefore, say that the system performs well and is stable across varied conditions.

We can draw upon the results obtained through both the datasets and infer that the decrease in dataset features is a parameter that can create a difference in the model's overall accuracy. Thus, we can observe changes in our accuracy and precision measures and all the used values. The losses converged in both cases, thus, implying that the system has been optimally trained. We might also observe that the data sets have played a role in the

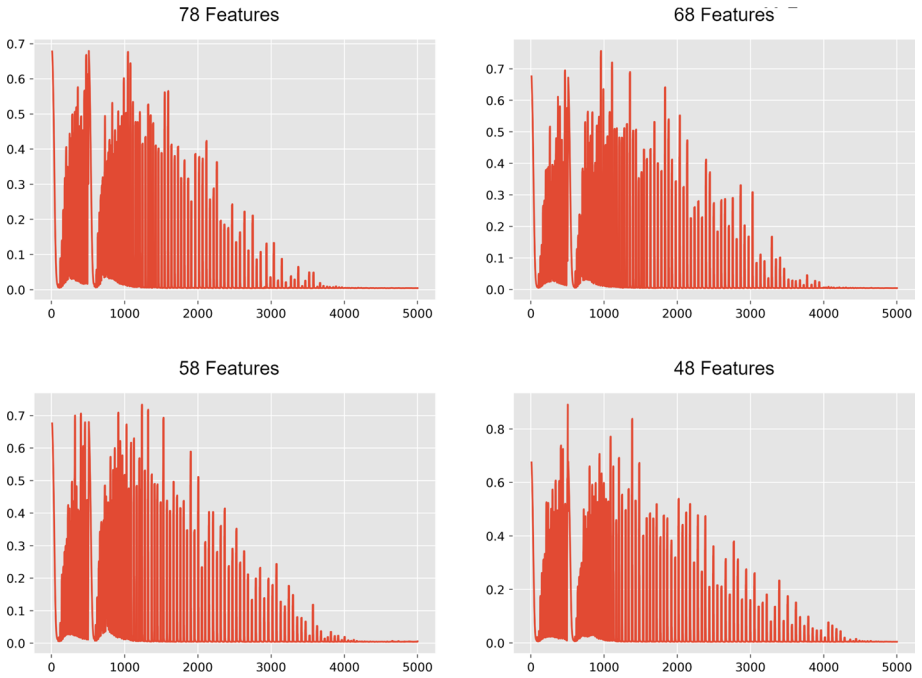

Fig. 6 Losses for CICIDS2017 dataset

Table 7 Confusion matrix for 78 features

	Pred 0	Pred 1
True 0	1234	1
True 1	0	1235

Table 8 Confusion matrix for 68 features

	Pred 0	Pred 1
True 0	740	1
True 1	0	741

Table 9 Confusion matrix for 58 features

	Pred 0	Pred 1
True 0	600	0
True 1	1	599

Table 10 Confusion matrix for 48 features

	Pred 0	Pred 1
True 0	493	1
True 1	0	494

Table 11 Accuracy and precisions for CICIDS dataset

Table number	Title	Accuracy (%)	Precision (%)
Table 7	For 78 features	99.95	99.91
Table 8	For 68 features	99.93	99.93
Table 9	For 58 features	99.91	100
Table 10	For 48 features	99.89	99.92

model's changing accuracy and precision. The precision measures are better in the case of NSL-KDD. However, we get better accuracy using the CICIDS2017 dataset than the NSL-KDD dataset. It may be the case because the NSL-KDD has much lesser attribute classes of data than the CICIDS2017 dataset, which leads to better data availability for the training and, thus, provides better accuracy. We compiled all the accuracy and precision measures that we obtained from the experiments over different values and have presented them in an orderly fashion in Tables 6 and 11.

5 Conclusion

This work aims to propose an Intrusion Detection System based on CGAN and the eXtreme Gradient Boosting Algorithm. Our proposed framework is a robust IDS based on Deep Learning to give it an advantage over other previously introduced IDSs (as referred to in Sect. 2), which may not be competent in today's rapidly changing and growing environment. A self-learning model proves to be more accurate when provided with more samples. The quality of samples in the running batch may also affect the accuracy and precision of the system. The proposed IDS provides excellent accuracy and precision under given constraints; besides, the model was applied with different datasets and dimensions. We show the model's efficacy in the 'Accuracy' and 'Precision' measure of the system. We get a decent value for both these parameters with the model that we have developed and altered for the said conditions. The proposed model can reduce the average number of sensors deployed by about 1.2–2.6% for our selected features, deployment strategy, and distribution, along with the false alarm rate that shows a reduction of 1.827%. The limitation of the proposed work is that due to the non-availability of a higher-end computing system, we could not study the model for a more extensive dataset with more samples. However, the samples chosen offer a considerable range of fraud and standard data to the training model. The future scope of the research may involve considering the usage of the complete datasets and keeping the features constant. One might also check the model's work by subjecting it to changes in the CGAN model. Also, changing the layer density of the neural networks can be studied for further work and betterment of the IDS.

Acknowledgements We want to acknowledge Madhav Institute of Technology & Science Gwalior, Gautam Buddha University Greater Noida, and IISER Bhopal for providing institutional support. We thank the editor and the anonymous reviewers for providing us with their helpful comments and suggestions to finally improve the manuscript in the current form.

Funding This research received no external funding.

Data Availability Data will be made available on reasonable request to the corresponding author.

Declarations

Conflict of interest We also declare that we do not have conflicts of interest with any person or agency.

References

1. Aysal, T. C., & Barner, K. E. (2008). Sensor data cryptography in wireless sensor networks. *IEEE Transactions on Information Forensics and Security, 3*(2), 273–289.
2. Chen, X., Makki, K., Yen, K., & Pissinou, N. (2009). Sensor network security: A survey. *IEEE Communications Surveys & Tutorials, 11*(2), 52–73.
3. Kotiyal, V., Singh, A., Sharma, S., Nagar, J., & Lee, C.-C. (2021). Ecs-nl: An enhanced cuckoo search algorithm for node localisation in wireless sensor networks. *Sensors, 21*(11), 3576.
4. Singh, A., Amutha, J., Nagar, J., Sharma, S., & Lee, C.-C. (2022). Lt-fs-id: Log-transformed feature learning and feature-scaling-based machine learning algorithms to predict the k-barriers for intrusion detection using wireless sensor network. *Sensors, 22*(03), 1070.
5. Singh, J., Chaturvedi, A., Sharma, S., & Singh, A. (2021). A novel model to eliminate the doubly near-far problem in wireless powered communication network. *IET Communications, 15,* 1539–1547.
6. Sharma, S., Kumar, R., Singh, A., & Singh, J. (2020). Wireless information and power transfer using single and multiple path relays. *International Journal of Communication Systems, 33*(14), e4464.
7. Amutha, J., Sharma, S., & Nagar, J. (2020). WSN strategies based on sensors, deployment, sensing models, coverage and energy efficiency: Review, approaches and open issues. *Wireless Personal Communications, 111*(2), 1089–1115.
8. Amutha, J., Nagar, J., & Sharma, S. (2021). A distributed border surveillance (DBS) system for rectangular and circular region of interest with wireless sensor networks in shadowed environments. *Wireless Personal Communications, 117*(3), 2135–2155.
9. Sharma, S., & Nagar, J. (2020). Intrusion detection in mobile sensor networks: A case study for different intrusion paths. *Wireless Personal Communications, 115,* 2569–2589.
10. Pandey, S. (2011). Modern network security: Issues and challenges. *IJEST, 3*, 4351–4356.
11. Roy, A. S., Maitra, B. N., Nath, C. J., Agarwal, D. S., & Nath, E. A. (2012) Ultra encryption standard (ues) version-ii: Symmetric key cryptosystem using generalized modified Vernam cipher method, permutation method, columnar transposition method and ttjsa method. In *Proceedings of the international conference on foundations of computer science (FCS)* (p. 1). The Steering Committee of The World Congress in Computer Science, Computer 2012.
12. Zhang, Y., Meratnia, N., & Havinga, P. (2010). Outlier detection techniques for wireless sensor networks: A survey. *IEEE Communications Surveys & Tutorials, 12*(2), 159–170.
13. Liu, H., & Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. *Applied Sciences, 9*(20), 4396.
14. Alsheikh, M. A., Lin, S., Niyato, D., & Tan, H.-P. (2014). Machine learning in wireless sensor networks: Algorithms, strategies, and applications. *IEEE Communications Surveys & Tutorials, 16*(4), 1996–2018.
15. Nancy, P., Muthurajkumar, S., Ganapathy, S., Kumar, S. S., Selvi, M., & Arputharaj, K. (2020). Intrusion detection using dynamic feature selection and fuzzy temporal decision tree classification for wireless sensor networks. *IET Communications, 14*(5), 888–895.
16. Ganapathy, S., Kulothungan, K., Muthurajkumar, S., Vijayalakshmi, M., Yogesh, P., & Kannan, A. (2013). Intelligent feature selection and classification techniques for intrusion detection in networks: A survey. *EURASIP Journal on Wireless Communications and Networking, 2013*(1), 1–16.
17. Depren, O., Topallar, M., Anarim, E., & Ciliz, M. K. (2005). An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. *Expert Systems with Applications, 29*(4), 713–722.
18. Balamurugan, N., Mohan, S., Adimoolam, M., John, A., Wang, W., et al. (2022). DOA tracking for seamless connectivity in beamformed IoT-based drones. *Computer Standards & Interfaces, 79,* 103564.
19. Kumar, S. S., Palanichamy, Y., Selvi, M., Ganapathy, S., Kannan, A., & Perumal, S. P. (2021). Energy efficient secured k means based unequal fuzzy clustering algorithm for efficient reprogramming in wireless sensor networks. *Wireless Networks, 27,* 3873–3894.
20. Singh, A., Sharma, S., & Singh, J. (2021). Nature-inspired algorithms for wireless sensor networks: A comprehensive survey. *Computer Science Review, 39*, 100342.

21. Amutha, J., Sharma, S., & Sharma, S. K. (2021). Strategies based on various aspects of clustering in wireless sensor networks using classical, optimization and machine learning techniques: Review, taxonomy, research findings, challenges and future directions. *Computer Science Review, 40*, 100376.
22. Singh, A., Kotiyal, V., Sharma, S., Nagar, J., & Lee, C.-C. (2020). A machine learning approach to predict the average localization error with applications to wireless sensor networks. *IEEE Access, 8*, 208253–208263.
23. Khan, T., Singh, K., Hasan, M. H., Ahmad, K., Reddy, G. T., Mohan, S., & Ahmadian, A. (2021). Eters: A comprehensive energy aware trust-based efficient routing scheme for adversarial WSNs. *Future Generation Computer Systems, 125,* 921–943.
24. Selvi, M., Thangaramya, K., Ganapathy, S., Kulothungan, K., Nehemiah, H. K., & Kannan, A. (2019). An energy aware trust based secure routing algorithm for effective communication in wireless sensor networks. *Wireless Personal Communications, 105*(4), 1475–1490.
25. Singh, A., Nagar, J., Sharma, S., & Kotiyal, V. (2021). A gaussian process regression approach to predict the k-barrier coverage probability for intrusion detection in wireless sensor networks. *Expert Systems With Applications, 172*, 114603.
26. Vallathan, G., John, A., Thirumalai, C., Mohan, S., Srivastava, G., & Lin, J.C.-W. (2021). Suspicious activity detection using deep learning in secure assisted living IoT environments. *The Journal of Supercomputing, 77*(4), 3242–3260.
27. Yadav, A. K., Singh, K., Ahmadian, A., Mohan, S., Shah, S. B. H., & Alnumay, W. S. (2021). Emmm: Energy-efficient mobility management model for context-aware transactions over mobile communication. *Sustainable Computing: Informatics and Systems, 30*, 100499.
28. Aksu, D., & Aydin, M. A. (2018). Detecting port scan attempts with comparative analysis of deep learning and support vector machine algorithms. In *2018 International congress on big data, deep learning and fighting cyber terrorism (IBIGDELFT)* (pp. 77–80). IEEE.
29. Wang, Z. (2018). Deep learning-based intrusion detection with adversaries. *IEEE Access, 6*, 38367–38384.
30. Al-Qatf, M., Lasheng, Y., Al-Habib, M., & Al-Sabahi, K. (2018). Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. *IEEE Access, 6,* 52843–52856.
31. Vinayakumar, R., Alazab, M., Soman, K., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. *IEEE Access, 7*, 41525–41550.
32. Alshinina, R. A., & Elleithy, K. M. (2018). A highly accurate deep learning based approach for developing wireless sensor network middleware. *IEEE Access, 6*, 29885–29898.
33. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672–2680).
34. Odena, A. (2016) Semi-supervised learning with generative adversarial networks. arXiv preprint arXiv:1606.01583, 2016.
35. Mirza, M., & Osindero, S. (2014) Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.
36. Sricharan, K., Bala, R., Shreve, M., Ding, H., Saketh, K., & Sun, J. (2017). Semi-supervised conditional gans. arXiv preprint arXiv:1708.05789, 2017.
37. Chen, Z., Jiang, F., Cheng, Y., Gu, X., Liu, W., & Peng, J. (2018). Xgboost classifier for DDOS attack detection and analysis in SDN-based cloud. In *2018 IEEE international conference on big data and smart computing (bigcomp)* (pp. 251–256). IEEE.
38. Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 785–794).
39. Dhaliwal, S. S., Nahid, A.-A., & Abbas, R. (2018). Effective intrusion detection system using xgboost. *Information, 9*(7), 149.
40. University of New Brunswick, NSL-KDD. (2009). http://nsl.cs.unb.ca/nsl-kdd/.
41. University of New Brunswick, CICIDS2017. (2017). https://www.unb.ca/cic/datasets/ids-2017.html.
42. Gadge, J., & Patil, A. A. (2008) Port scan detection. In *2008 16th IEEE international conference on networks* (pp. 1–6). IEEE.

Tanya Sood received her Integrated Dual Degree (B.Tech. and M.Tech.) in Electronics and Communication Engineering with specialization in Wireless Communication and Networks in 2020 from Gautam Buddha University, Greater Noida, India. She is currently working as an analyst at IQVIA. Her research interest includes WSNs and their applications, data analysis, deep learning, and machine learning. She is currently working with IoT applications in healthcare.

Satryartha Prakash received his Integrated Dual Degree (B.Tech. and M.Tech.) in Electronics and Communication Engineering with specialization in Wireless Communication and Networks from Gautam Buddha University, Greater Noida, India. He is currently working as a Senior Project Fellow at CSIR-Institute of Genomics and Integrative Biology, New Delhi, India.

Sandeep Sharma received a B.Tech. Degree in Electronics Engineering from RGPV, Bhopal, India, in 2001 and an M.Tech. Degree in Digital Communication from Devi Ahilya University, Indore, India, in 2005. He is a Ph.D. in Electronics and Communication Engineering from Gautam Buddha University, Greater Noida, India, in 2016. Currently, he is an Assistant Professor in the Department of Electronics Engineering at Madhav Institute of Technology and Science, Gwalior (M.P.), India. Before joining MITS Gwalior, he was with Gautam Buddha University, Greater Noida. His research interest includes wireless sensor networks, wireless network security, physical layer authentication, intrusion detection in wireless networks, cross-layer design, and machine learning applications in WSNs. He has published 40 research papers in reputed international journals and more than 43 papers published in international conferences. He also authored 6 book chapters which were published with reputed publishers like Springer, Taylor & Francis CRC Press. Dr. Sharma is a recipient of the Best Conference Paper in the international conference ICCCS, 2016, and the Young Scientist Award in 2019 for his research work. He is an active reviewer of IET Communications, IEEE Wireless Comm. Letters, Journal of Information Technology (Springer), Personal and Ubiquitous Computing (Springer), Multimedia Tools and Applications (Springer), International Journal of Computer Applications in Technology (Inderscience), International Journal of Communication Systems (Wiley), Journal of The Institution of Engineers (India): Series B, Journal of Intelligent and Fuzzy Systems, Neural Networks (Elsevier), Expert System with Applications (Elsevier), Soft Computing (Elsevier), Artificial Intelligence Review (Springer), Sensor (MDPI), Electronics (MDPI), Designs (MDPI).

Abhilash Singh received his Integrated Dual Degree (B.Tech. and M.Tech.) in Electronics and Communication Engineering with specialization in Wireless Communication and Networks in 2017 from Gautam Buddha University, Greater Noida, India. He is currently working on his Ph.D. Degree in Remote Sensing from the Indian Institute of Science Education and Research Bhopal, Bhopal, India. Since 2018, he has been working on a NASA-ISRO Synthetic Aperture Radar (NISAR) project at IISER Bhopal, Bhopal, India. He has been publishing research articles in peer-reviewed conferences and internationally reputed journals. His current research interest includes microwave remote sensing, machine learning, bio-inspired algorithms, wireless sensor network, and wireless communication. Mr. Abhilash was a recipient of gold medal awards from the University for being a first rank holder in his UG and PG. He received the prestigious "DST-INSPIRE" Fellowship to carry out his doctoral degree from the Department of Science and Technology (DST), India's Ministry of Science and Technology. He also received the DAAD fellowship, travel grant from American Geophysical Union (AGU), and AGU Ecohydrology Early Career Tiny Grant. Recently, he got featured as a 'leaf' in the meet a leaf series of AGU ecohydrology section. He is an active reviewer of Remote Sensing of Environment, Artificial Intelligence Review, Complex & Intelligent Systems, Journal of Intelligent and Fuzzy Systems, Advances in Space Research, IEEE Access, The Journal of Open Source Software, Wireless Personal Communications, Journal of The Institution of Engineers (India) Series B, and Journal of Ambient Intelligence and Humanized Computing. He is a member of IEEE (student), European Geophysical Union (EGU), American Geophysical Union (AGU), ISPRS, and the Indian Radio Science Society (InRaSS).

Hemant Choubey received a B.Tech. Degree in Electronics and Communication Engineering from RGPV, Bhopal, India, in 2009 and an M.Tech. Degree in Digital Communication from University Institute of Technology, RGPV, Bhopal, India, in 2013. He is a Ph.D. in Electronics and Communication Engineering from MANIT, Bhopal, India, in 2020. Currently he is working as an Assistant Professor in the Department of Electronics Engineering, MITS, Gwalior, India. Before MITS, he was with TIT&S Bhopal, India as a Head of Electronics and Communication Engineering Department during 2016–2021. His research interests include Digital Communication, Biomedical Signal Processing, and Chaotic Communication. Dr. Choubey has published many papers in reputed international journals and conferences. He also on the board of Editor and reviewer of several reputed international journals. He is also a member of many international bodies.

Authors and Affiliations

Tanya Sood[1] · Satyartha Prakash[2] · Sandeep Sharma[3] · Abhilash Singh[4] · Hemant Choubey[3]

Tanya Sood
tanya27alt@gmail.com

Satyartha Prakash
satyartha.p@igib.in

Abhilash Singh
sabhilash@iiserb.ac.in

Hemant Choubey
hemantchoubey271986@gmail.com

[1] School of Information and Communication Technology, Gautam Buddha University (GBU), Greater Noida, Uttar Pradesh 201312, India

[2] CSIR- Institute for Genomics and Integrative Biology (IGIB), New Delhi 110025, India

[3] Department of Electronics Engineering, Madhav Institute of Technology and Science, Gwalior 474005, India

[4] Fluvial Geomorphology and Remote Sensing Laboratory, Indian Institute of Science Education and Research Bhopal, Bhopal 462066, India