I. J. Computer Network and Information Security, **2023, 1, 26-38**
Published Online on February 8, 2023 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2023.01.03

Outlier Detection Technique for Wireless Sensor Network Using GAN with Autoencoder to Increase the Network Lifetime

Biswaranjan Sarangi*
Biju Patnaik University of Technology (BPUT), Department of CSE, Rourkela, 769015, Odisha, India
E-mail: biswaranjan.sarangi@gmail.com
ORCID iD: http://orcid.org/0000-0002-8999-2194
*Corresponding Author

Biswajit Tripathy
GITA Autonomous College, Department of CST, Bhubaneswar, 752054, Odisha, India
E-mail: biswajit69@gmail.com
ORCID iD: http://orcid.org/0000-0001-9707-9170

Received: 17 July 2022; Revised: 06 October 2022; Accepted: 14 November 2022; Published: 08 February 2023

Abstract: In wireless sensor networks (WSN), sensor nodes are expected to operate autonomously in a human inaccessible and the hostile environment for which the sensor nodes and communication links are therefore, prone to faults and potential malicious attacks. Sensor readings that differ significantly from the usual pattern of sensed data due to faults in sensor nodes, unreliable communication links, and physical and logical malicious attacks are considered as outliers. This paper presents an outlier detection technique based on deep learning namely, generative adversarial networks (GANs) with autoencoder neural network. The two-level architecture proposed for WSN makes the proposed technique robust. The simulation result indicates improvement in detection accuracy compared to existing state-of-the-art techniques applied for WSNs and increase of the network lifetime. Robustness of outlier detection algorithm with respect to channel fault and robustness concerning different types of distribution of faulty communication channel is analyzed.

Index Terms: WSN, Outlier, GAN, Autoencoder, Network-lifetime.

1. Introduction

The rapid development of smart sensors, allowing independent sensor nodes to be linked up to establish a WSN to track, identifies and report time-critical issues. Different types of sensor nodes are deployed in hostile environments in large numbers with both short and long operation time. As a result, unexpected and frequent faults occur in WSNs, where the nodes behave arbitrarily or maliciously. Sensor nodes suffer from many resource constraints [1]. In a resource-constrained WSN, the low computing power limits the adoption of high-end security solutions and thus renders the nodes more susceptible to security threats. The outlier sources are faults in sensor nodes, adverse channel conditions, and malicious attacks. Permanent, intermittent, and transient type of faults are considered both in hardware and software [2]. The outliers may lead to misinterpretation or unwanted alarms resulting in life-threatening incidents particularly in safety-critical applications. For efficient band width and energy usage, outliers do need to be prohibited from entering the network to increase the life time of the network [3]. To the best of our understanding, most of the current methods of outlier detection do not detect outliers in real-time. Rajasegarar et al. [4] suggested one-class quarter SVM technique which reduced computational complexity and identified outliers locally by considering the sensor data outside of the quarter sphere. Abid et al. [5] suggested un-supervised outlier detection using data nearest for outlier detection (DNOD) technique. Chen and Lee [6] suggested spatio-temporal and attribute correlated support vector data description (STASVDD) approach that consider that outliers can independently occur in each attribute when the collected data vectors are independent and identically distributed. Luo and Nagarajan [7] suggested a two-part algorithm by using autoencoder, which resides respectively on sensor nodes or the cloud. Offline outlier detection may have a severe impact on the accuracy of real-time decision making.

This paper presents an unsupervised learning technique called GAN for outlier detection. The generator (G) network and the discriminator (D) network of GAN are implemented using autoencoder and each autoencoder is trained with Adam optimizer. A two-layer WSN architecture for the detection of outlier is suggested where the cluster-heads use the proposed detection algorithm to detect outlier locally.

2. Related Works

GAN is the method for assessing generative models through an adversarial mechanism in which two models, the discriminator (D) and the generator (G) are trained simultaneously. The main objective of G is to capture the distribution of data, while D estimates probability that a sample originated from training data rather than G[8]. As shown in Fig.1., autoencoders are used to implement G and D. An autoencoder aimed at reconstructing inputs, rather than predicting certain target variables. An autoencoder learn non-linear transformations with a non-linear activation function and multiple layers which helps to reduce the noise in data as well as dimensionality[9]. As illustrated in Fig.2., an autoencoder consists of an input layer, an output layer and one or more hidden layers.

- Input layer is an M-dimension vector representing the time series of sensor readings represented by $x = (x_1, x_2, \dots \dots, x_m)$.
- Output layer is represented as $x' = (x'_1, x'_2, \dots, x'_m)$.
- Hidden layer(s) is placed between the input and output layer.

This work considers a single hidden layer to avoid computational complexity. An autoencoder consists of activation function and the hyper parameters. The activation function used in this work is a hyperbolic tangent function and is given by Eq. (1).

$$f(z) = \tanh(z) = \frac{2}{1+e^{2z-1}}. \quad (1)$$

Let the weights W and biases b is the hyper-parameters of an autoencoder, where $W_{ij}^{(l)}$is the weight related with the connection from node j in layer l to layer l+1 and $b^{(l)}$is the bias related with node i in layer l+1. Therefore the activation $a_i^{(l)}$ is given by Eq. (2).

$$a^{(l)} = f\left(W^{(l-1)}a^{(l-1)} + b^{(l-1)}\right) \quad (2)$$

The intercept term $b^{(l-1)}$ in the above articulation compares to the node (in $n^{(l)}$duplicates) at layer *l*-1, which is known as bias unit. The result x' is then $a^{(L)}$ which is indicated as $h_{(w,b)}(x)$ where L is the total number of layers.{x[1],x[2],….,x[T]} are training samples (T), where each x[i] = { x_1[i],x_2[i],….,x_M[i]}. The target of an autoencoder is to limit the cost function as in Eq. (3).

$$J(W,b) = \frac{1}{T}\sum_{i=1}^{T}\frac{1}{2}\left\|h_{(W,b)}x[i] - x[i]\right\|^2 + \frac{\lambda}{2}\sum_{l=1}^{L-1}\sum_{j=1}^{n^{(l)}}\sum_{i=1}^{n^{(l+1)}}\left(W_{ij}^{(l)}\right)^2 \quad (3)$$

The initial term portrays the reconstruction error comparable to the first data sources and the subsequent term for regularization to forestall over fitting. In this work, ADAM optimizer is utilized to limit the objective function [10].

D and G performs the following two player minimax game with value function V (G, D) given by Eq. (4).

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim Pdata(x)}[logD(x)] + \mathbb{E}_{z\sim Pz(z)}[log(1 - D(G(z))] \quad (4)$$

For the generated samples$G_{auto}(z_i)$, where z is a distribution from the random latent space, the generator G, an autoencoder model, characterizes a probability distribution. Another autoencoder, the discriminator, is then trained to limit the average negative cross-entropy between its expectations and succession labels. Thus the discriminator loss is given by Eq. (5).

$$D_{loss} = \frac{1}{M}\sum_{i=1}^{M}\left[logD_{auto}(x_i) + \log\left(1 - D_{auto}\left(G_{auto}(z_i)\right)\right)\right] \quad (5)$$

The discriminator loss should be minimized to recognize x_i as real and $G_{auto}(z_i)$ as false. The generator is trained to confuse the discriminator such that the discriminator would recognize as much as generated samples as real as possible. The generator loss is given by Eq. (6).

$$G_{loss} = \frac{1}{M}\sum_{i=1}^{M}\left[\log\left(1 - D_{auto}\left(G_{auto}(z_i)\right)\right)\right] \tag{6}$$

A GAN based outlier detection scheme is presented in Algorithm 1. As described in the algorithm upon completion of the training of the module, the threshold has been evaluated using precision and recall. The trained module is then deployed in all the cluster heads. The updated W, b and Threshold are scheduled to be sent to the cluster heads periodically by the sink or cloud.

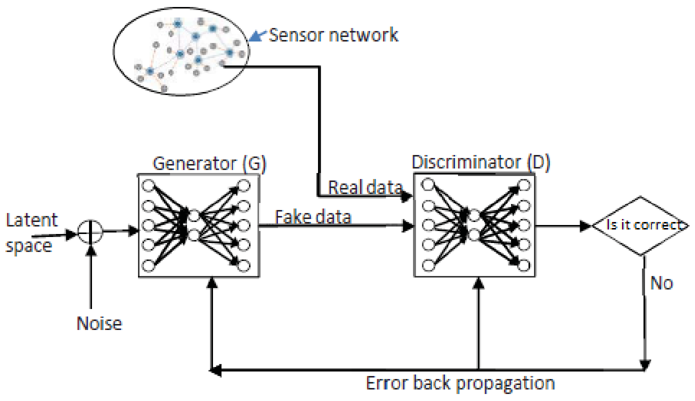

Fig.1. GANs Framework.

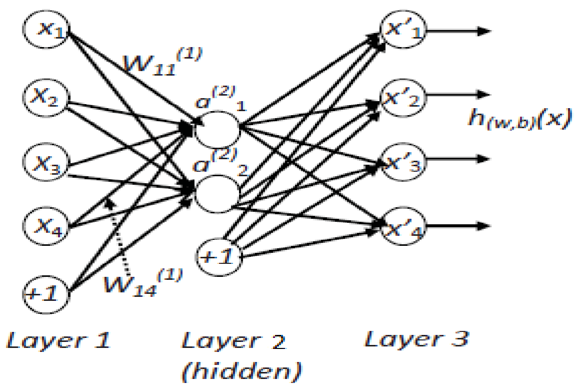

Fig.2. Structure of an Autoencoder [7].

Algorithm 1 Autoencoder-GAN-based Outlier Detection

```
1.  Input: Dataset D
2.  for j=1 to training iterations do
3.  for k steps do
4.      Sample minibatch of m noise samples {z1,z2,…zm} from noise with distribution $p_g$(z).
5.      Sample minibatch of m original data samples {x-1,x2,…,xm} from D.
6.      Update discriminator parameters by minimizing $D_{loss}$
7.      Minimize $D_{loss} = \frac{1}{M}\sum_{i=1}^{M}\left[logD_{auto}(x_i) + \log\left(1 - D_{auto}\left(G_{auto}(z_i)\right)\right)\right]$
8.  end for
9.     Sample minibatch of m noise samples {z1,z2,…,zm} from noise with distribution$p_g$(z).
10. Update generator parameters by minimizing $G_{loss}$
Minimize $G_{loss} = \frac{1}{M}\sum_{i=1}^{M}\left[\log\left(1 - D_{auto}\left(G_{auto}(z_i)\right)\right)\right]$
11.  end for
12.  Evaluate threshold using Precision and Recall.
13.  Send updated W, b and Threshold to the cluster heads.
```

In the proposed system architecture, each cluster head runs one copy of GANs as shown in Fig.3. Apart from sensing, it performs following assignments: (i) collects sensor readings from its affiliating sensor nodes, (ii) relays the sensor readings received from neighboring cluster heads,(iii)receives the updated hyper-parameters (W,b) for both autoencoder-based generator and discriminator of the GANs from sink node or cloud, (iv) perform outlier detection locally using the received threshold from sink node or cloud, and(v) sends the outlier report to the sink node through the spanning tree of cluster heads as shown in Fig.4. All sensor readings are obtained from individual cluster heads in the cloud. For each cluster head in the network, the sink node or the cloud generates a copy of GANs i.e., n copies of GANs assuming n cluster heads in the network. Each copy of GANs representing one cluster is periodically trained in the cloud by using the sensor data received from the corresponding cluster head. The inter training interval is application specific, and it varies a few minutes for applications with short operation time to a few hours or even once in a day for applications with long operation time. It should be noted that if the sink node is sufficiently competent to train the GANs, then the cloud will not be included within the architecture. The cloud transmits the latest hyper-parameters (W, b) and threshold value to the respective cluster heads through sink node to update their own generator and discriminator.

Numerous researchers have focused on WSN clustering [11]. However, the unequal cluster-based routing (UCR) protocol is considered to mitigate the hot spot problem. With an increase in the distance from the sink node, the cluster size increases[12]. In UCR protocol, the cluster head status is revolved among sensors at each turn to allocate the equal load through network.

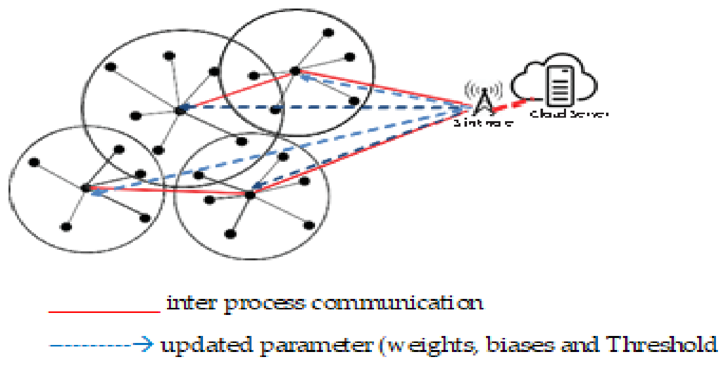
Sink node
Cloud Server
inter process communication
updated parameter (weights, biases and Threshold

Fig.3. System Architecture.

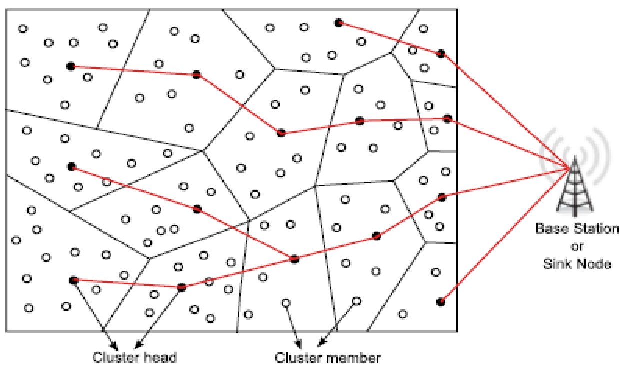

Fig.4. Clusters and Spanning Tree.

3. Proposed Method Logy

A two-layer outlier detection technique (Algorithm 2) is proposed where the detection is performed at two-layers in a WSN where sensors are organized into n numbers of clusters. At each cluster, the outlier is identified locally by the cluster heads. First, the cluster head checks the readings received from its member nodes. If an outlier is detected for node v_i, the cluster head checks the reading obtained from the respective friend node v_i, i.e., $x_{i\ \mathrm{Friend}}$. If this reading is detected as an outlier, the cluster head drops the reading and marks the node as a misbehaving node. A two-time checking is suggested for the reading identified as an outlier because it may be possible that the cluster head receives a reading (say x_i from v_i) corrupted by the noisy channel. Dropping of outlier data by the cluster heads prevents the outlier data from reaching the network and increases the efficient use of band width and energy. The cluster head decisions are then communicated to the sink node using the spanning tree. Because the behavior of wireless communication channels is dynamic, and in particular, more complex in indoor environments, sensor readings can also be affected during inter cluster communication. Thus, a final detection of the outliers is performed at the sink node.

A node misbehaves if it is either faulty or attacked by malicious attacks or data from that node may be delivered incorrectly to the cluster head due to faulty channel. The sink node instructs the cluster heads to isolate the misbehaving nodes and to keep these misbehaving nodes under observation for a random number of data-gathering rounds. The cluster head continues to track these misbehaving nodes and allows them to engage in network operations after the aforementioned rounds of data collection rounds if they behave correctly; otherwise, they are permanently removed from the network. This observation period allows the cluster head to decide whether a node suffers from transient fault or intermittent and permanent fault. Further, it will enable the cluster head to determine if the data is corrupted due to faults in the communication channel or the node is misbehaving.

Algorithm 2. Outlier Detection (Cluster head)

```
Input: {$W_{CH_i}$, $b_{CH_i}$}and Threshold$_{CH_i}$from the sink node or cloud.
1.  Sensor node $v_i$selects randomly a Friend node $v_j$such that $v_j €$ N($v_i$)and $v_j$and $v_i$are members of cluster
    head $CH_i$.
2.  Sensor node $v_i$sends friend request to $v_j$.
3.  Sensor node $v_i$broadcast the sensor reading $x_i$.
4.  Sensor node $v_j$over hears $x_i$ and marks it as $x_{iFriend}$.
5.  The friend node $v_j$sends $x_{iFriend}$ to cluster head $CH_i$by embedding the node ID of $x_i$.
6.  Cluster head $CH_i$ obtains sensor readings from its member nodes.
7.  Cluster head $CH_i$ obtains the second copy of sensor readings from its member nodes declared as Friend
    nodes in its cluster.
8.  Cluster head runs the outlier detection algorithm on the sensor reading $x_i$.
9.  Compute reconstruction error for $x_i$.
10. if Reconstruction error is greater than Threshold$_{CH_i}$ then
11.     Cluster head runs the outlier detection algorithm on the sensor reading $x_{iFriend}$.
12.     Compute reconstruction error for $x_{iFriend}$.
13. if Reconstruction error is greater than Threshold$_{CH_i}$then
14.         Declare the data sample as outlier.
15.         Drop the reading $x_i$.
16. end if
17.   else
18. Cluster head relays reading $x_i$.
19. end if
20. Upon receiving all the sensor readings through the cluster head backbone, the sink node executes the outlier
    detection algorithm on $x_i$and takes the final decision.
21. Node identified with outlier is kept under observation for random data interval to take care of intermittent
    and transient fault.
```

3.1. Data Set

To validate the proposed model, some extracted data samples from the sensor scope dataset [13] have been utilized. Considering the integrity and continuity of data, only measurements of surface temperature and ambient temperature from the sensor node with ID 10 and 12 are included for this work. Some synthetic data are embedded into the data set. Each fault has a shortcoming start time (f_{st}) and shortcoming end time (f_{et}). This work considers spike and shock fault models to generate outliers where the error process $e(t)$ is a function of the fault type $f(t)$ as given in Eq. (7) and Eq. (8).

Spike fault:

$$e(t) = \begin{cases} x_i(t) + \Delta if f_{st} \leq f_{et} \\ x_i(t) otherwise \end{cases} \tag{7}$$

Where Δ is the spike magnitude and ($f_{st} - f_{et}$) →0.

Shock:

$$e(t) = \begin{cases} x_i(t) + \Delta if f_{st} \leq f_{et} \\ x_i(t) otherwise \end{cases} \tag{8}$$

Where Δ is the spike magnitude and ($f_{st} - f_{et}$) = some random finite value.

3.2. Evaluation Metric

The matrices considered for evaluation of the proposed method are given in Eq. (9) to Eq. (13) [14].

$$Accuracy\, rate = \frac{TP+TN}{TP+TN+FP+FN} \tag{9}$$

$$\text{Precision (P)} = \frac{TP}{TP+FP} \tag{10}$$

$$\text{True Positive Rate (Recall) TPR} = \frac{TP}{TP+FN} \tag{11}$$

$$\text{False Positive Rate FPR} = \frac{FP}{FP+TN} \tag{12}$$

$$\text{F1} = \frac{2(Precision \times Recall)}{Precision+Recall} \tag{13}$$

3.3. *Experimental Setup*

For event-based simulation of distributed algorithms, Python library Pymote 2.0 is used [15]. Table1. tabulates the radio model parameters as in UCR for the simulation. In this simulation, at any time a sensor transmits or receives data, energy is consumed. The channel error rate Pcerr at each node is 1 X 10^{-3}. The UCR protocol parameters are set as T=0.2, R0 =80m, c =0.3, TD MAX =100m, and k =2.

Table 1. Simulation Parameters.

Parameter	Value
Number of sensors	920
Network Grid	From (0,0) to (600, 400) m
Sink	At (620, 200) m
Initial energy	1 J
E_{elec}	50 nJ/bit
ϵ_{fs}	10pJ/bit/m2
ϵ_{amp}	0.0013pJ/bit/m4
d_0	87m
E_{DA}	5 nJ/bit/signal

The data from nodes 10 and 12 are used for testing the performance of the projected GANs based outlier detection algorithm. For both the generator and the discriminator, the autoencoder neural network has input and output layers each with nodes of n (1) = M = 32. There are n (2) = 16 number of hidden neurons corresponding to a 50% compression ratio.

4. Results and Discussion

The experiment is performed in two stages: (i) the classifier in view of GANs is first trained using training dataset and then tried with the test dataset of Sensor Scope. The performance of the suggested classifier is contrasted with the state-of-art classifiers. (ii) A WSN with 920 nodes is created and the test dataset of Sensor Scope's 92 nodes is replicated and randomly deployed in the 920 nodes. Experiments are performed to determine the performance of the proposed approach.

4.1. *Experiment 1*

Both the discriminator and the generator are trained in this experiment, and the threshold is obtained experimentally. For training 80% of data from the dataset is used while the 20% data is used for testing. Further, 20% of the training data is used for validating the training model. As shown in Fig.5., the proposed model is assessed on the training dataset and validation dataset after each update during training and plots the performance. The training accuracy is depicted in Fig.6.

In data science, where the negative class substantially out-numbers the positive class, accuracy is not a reasonable metric for model performance evaluation [16]. Therefore, both precision and recall should be evaluated to determine the efficacy of the model. It is known that precision and recall are conflicting objectives, and thus trade-offs are considered to determine a threshold. Fig.7. shows the different values of both precision and recall for various threshold values. The threshold at which the precision curve intersects the recall curve is known to be the optimum threshold, and its value is found to be 0.9. Reconstruction error for different data points of the test data is shown in Fig.8. The outliers are the data points above the threshold line.

Fig.9. demonstrates the confusion matrix which shows performance on a set of test data for which the true values are known. It provides visualization of an algorithm's performance for the proposed approach and compared with state-of-the-art solutions as shown in Table 2.

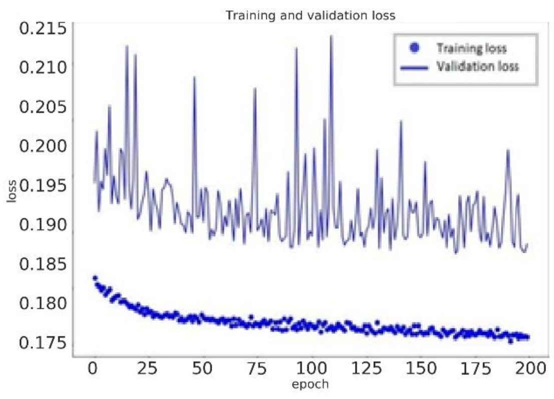

Fig.5. Training Learning Curve.

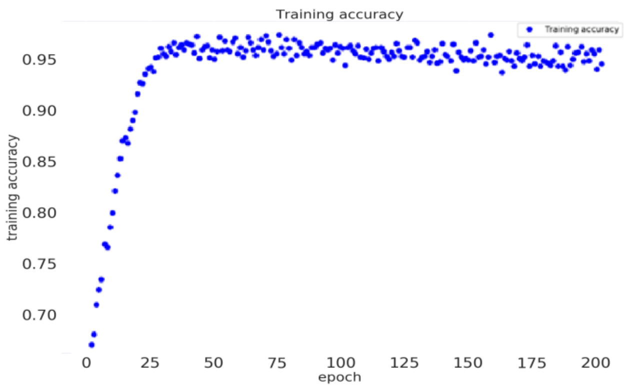

Fig.6. Training Accuracy.

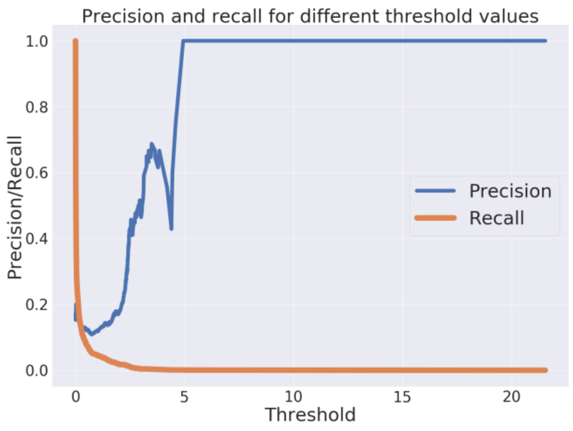

Fig.7. Precision and Recall for Different Threshold Values.

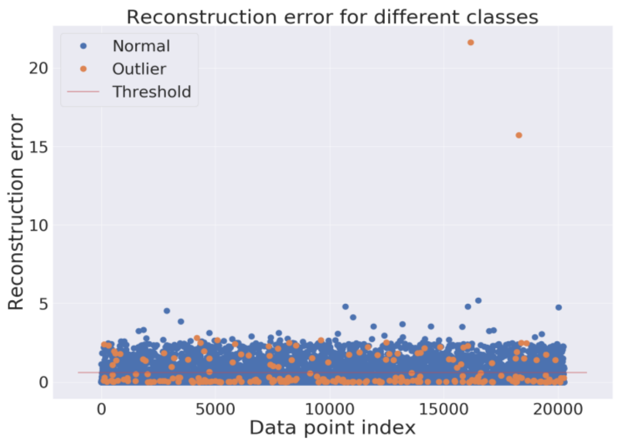

Fig.8. Detection Using the Threshold.

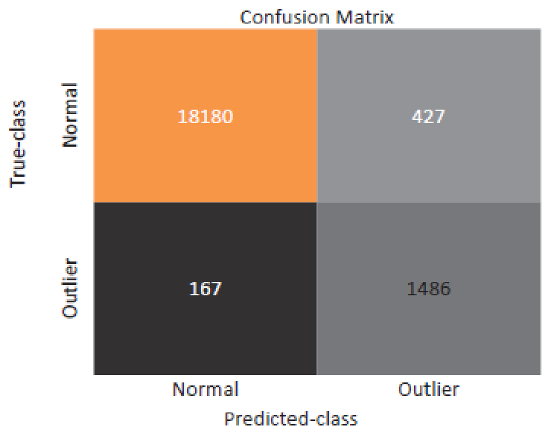

Fig.9. Confusion Matrix.

Table 2. Comparison with State-of-art Solutions.

Method	AR	TPR	FPR	P	F1
SVM [4]	82.29	84.86	42.11	95.05	89.06
DNDO [5]	84.51	87.04	39.83	95.44	91.05
N-STASVDDc [6]	90.65	92.78	29.74	96.76	94.72
DADA [7]	86.94	89.45	37.11	95.85	92.54
Proposed	94.11	95.81	22.32	97.62	96.7

4.2. *Experiment 2*

Outlier detection algorithm concerning TPR and FPR is assessed and matched with DADA [7]. In this simulation, sensor nodes are assumed to have outliers with probabilities of 0.05, 0.10, 0.15, 0.20, 0, 25, and 0.30 respectively. In this experiment the positioning of both nodes with outlier and nodes without outlier follow the uniform distribution. As expected and shown in Fig.10. and Fig.11., the TPR and FPR of the proposed approach outperform both DADA.

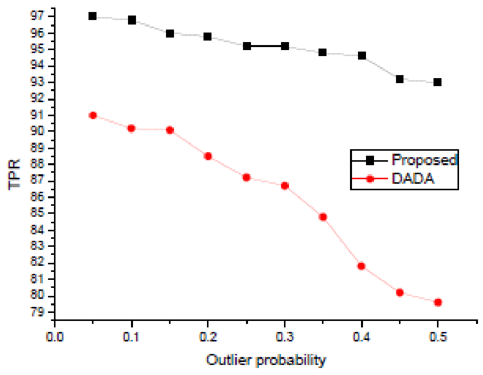

Fig.10. True Positive Rate.

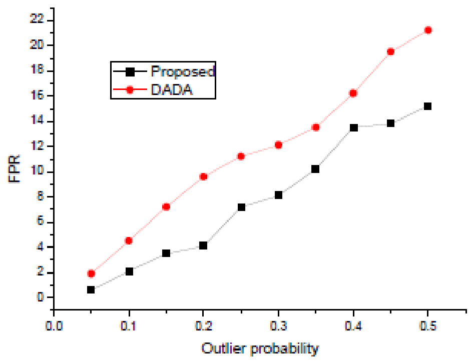

Fig.11. False Positive Rate.

A. *Robustness with Respect to Channel Fault*

A two-state Markov channel model is considered having two states: G (good) state and B (bad) state. The errors on the bit level is considered where Pgood is the probability that the bits are received incorrectly in good channel condition, and Pbad is the probability that the bits are received incorrectly in poor channel condition. It is assumed that $\mathrm{Pgood} \ll \mathrm{Pbad}$. The probability in G and B is high and transition probability $\mathrm{T_{GB}} = \mathrm{P(G \rightarrow B)}$ and $\mathrm{T_{BG}} = \mathrm{P(B \rightarrow G)}$ is small. In the bad state, the steady-state probability of a channel is $\mathrm{P_B} = \mathrm{T_{GB}}/(\mathrm{T_{GB}} + \mathrm{T_{BG}})$. Therefore, the channel's average bit error probability is $\mathrm{Pcerr} = \mathrm{PbadP_B} + \mathrm{Pgood}(1 - \mathrm{P_B})$.The strength of the outlier identification calculation concerning channel faults is assessed in this experiment by estimating FPR for different probabilities of channel errors. Pgood is taken as 0 for simplicity in the simulation, and Pbad as 1. To acquire different channel error probabilities, Pcerrr, PBG is set to 1/8 and PGB is wide-ranging. The previously generated network is used with an outlier probability of 0.2. The channel error rate is enlarged in steps from 10^{-5} to 10^{-1}. Communication channel faults could cause the cluster head to fail in receiving a correct reading from a healthy node. In this case, we refer to the healthy node as the non-misbehaving node. However, in the proposed approach, the cluster head receives two copies of sensor readings for a sensor node, i.e. x_i and $x_{iFriend}$. The probability of channel error for the link between v_i and cluster head might not be the same as the probability of channel error for the link between $v_{iFriend}$ and cluster head. Hence, the proposed approach increases the possibility of receiving a correct reading from a healthy node under a dynamic channel environment and thus improves the performance of the FPR. Thus, the proposed outlier identification algorithm really bears communication channel faults. As shown in Fig.12., the DADA is noted to be terrible impacted by changing channel error rate. The cluster head does not consider channel fault while taking the decision and may declare a healthy node as an unhealthy node is the main reason in DADA.

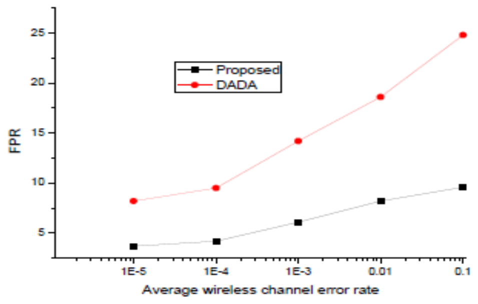

Fig.12. FPR in Presence of Channel Fault: Outlier Probability = 0.2.

B. *Network Life Time*

The efficiency from the perspective of overall network energy consumption is assessed by calculating the network life time. In this simulation we consider the worst case definition for the network-lifetime, when the first node becomes dead due to the low battery power. In this experiment if there is a loss of 99% of its initial energy, a node is treated as dead node. We introduce the proposed scheme and DADA in accordance with UCR protocol for better comparative study. As with the UCR protocol, we set the data packet size to 4000. First we simulated without taking into account any outlier detection method. Then, along with the UCR protocol, we implement the proposed scheme and the DADA. The life time of the network for varying outlier rates is indicated in Fig.13.

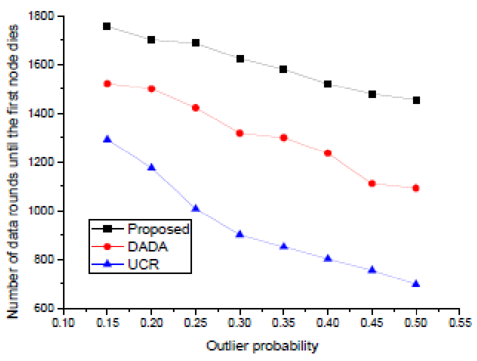

Fig.13. Network Life-Time.

When the outlier identification algorithms operate along with the clustering protocol, then there is an increase in the network lifetime. As seen, in a situation, where the outlier rate is high, this improvement is noteworthy. That's what the clarification is assuming unhealthy nodes are allowed to send their data to the sink node, and then relay nodes disperse energy by transmitting this inaccurate data. Because the inaccurate data generated by the unhealthy nodes are rejected in the proposed approach and DADA, energy wastage is avoided in relying these erroneous data. This increases network-lifetime. However, since the TPR and FPR of DADA is high compare to the proposed work, the network-lifetime of DADA is low as compared to the proposed work.

C. *Robustness Concerning the Distribution of Faulty Communication Channel*

It is clear that the state of the communication channel is not uniform across the geographical region where the sensor network is deployed. In this experiment, the robustness of the proposed outlier detection algorithm for different types of distribution of faulty communication link in the WSN is analyzed by estimating TPR and FPR. In this experiment we consider two scenarios: (i) Nodes with fault-free communication links to all its one-hop neighbors follow the uniform distribution. In contrast, nodes with faulty communication links to at least one of its one-hop neighbors follow the normal distribution. (ii) Nodes with fault-free communication links to all its one-hop neighbors follow the uniform distribution. In contrast, nodes with faulty communication links to at least one of its one-hop neighbors follow the beta distribution.

An example of a network where nodes with faulty communication links follow the normal distribution is shown in Fig.14. The normal distribution mean and standard deviation is set at 200 and 50, respectively. As expected, and shown in Fig.15, almost no change in TPR for both the proposed technique and DADA is noticed. However, with a higher fault rate, DADA is worst affected from the perspective of the FPR. This is because the concentration of nodes with faulty communication links to at least one of its one-hop neighbors in a particular geographical area increases with an increase in the fault rate. In DADA, the cluster head detects a healthy node as unhealthy if it receives a signal corrupted by the channel. In contrast, the cluster head in the proposed approach receives two copies of sensor readings for a sensor node, i.e., x_i and $x_{iFriend}$. As discussed earlier, the proposed approach increases the possibility of receiving a correct reading from a healthy node under a dynamic channel environment and thus improves the performance of the FPR.

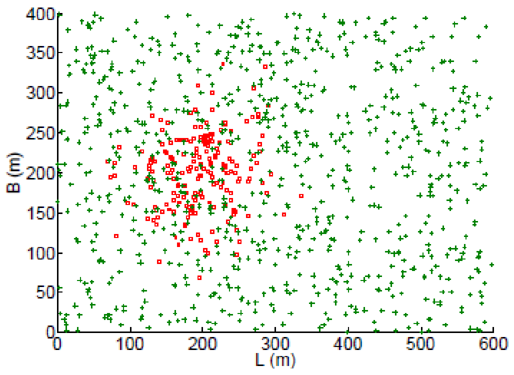

Fig.14. Example Network (Normal Distribution) P=0.2.

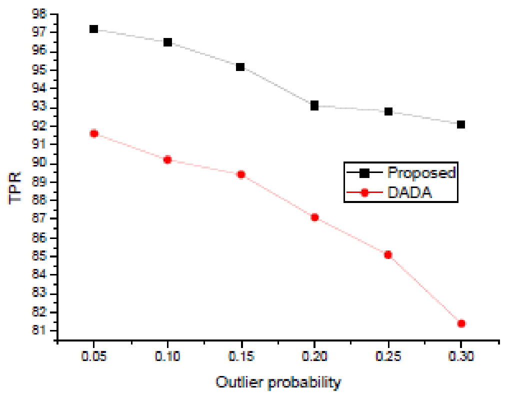

Fig.15. TPR (Normal Distribution).

An example network that follows a beta distribution for nodes with faulty communication links is shown in Fig.17.

The shape parameters (α and β) for beta distribution are set to 0.3. it is observed that both the proposed approach and DADA are least affected from TPR perspective. An imrovement in FPR for DADA is observed. This is because, in this scenario, nodes with faulty communication links are not distributed in a particular region of the network.

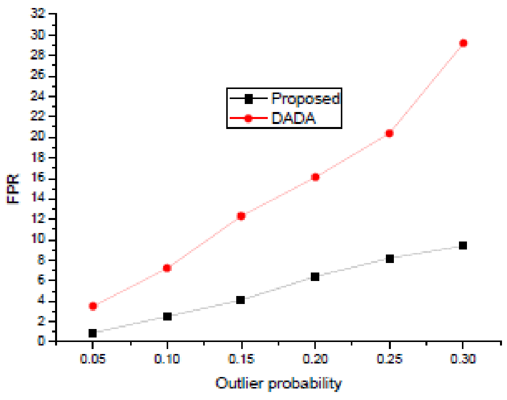

Fig.16. FPR (Normal Distribution).

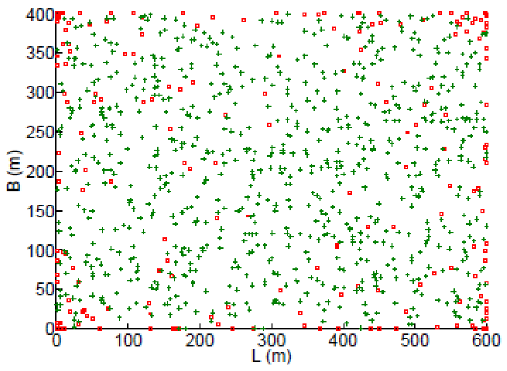

Fig.17. Example Network (Beta Distribution), P=0.2.

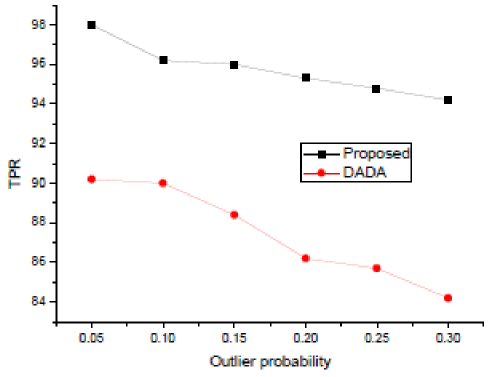

Fig.18. TPR (Beta Distribution).

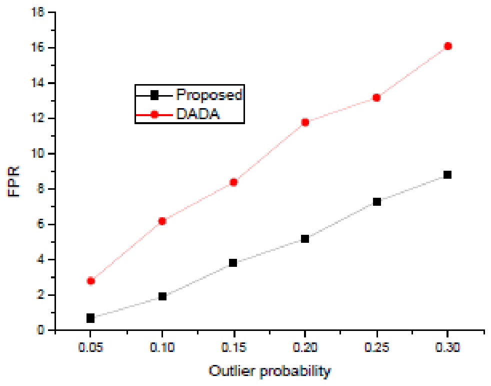

Fig.19. FPR (Beta Distribution).

5. Conclusion

This paper presents introduction of GAN with autoencoder neural network into resource-constrained WSN to perform online outlier identification that is integrated with the unequal cluster-based routing protocol. An optimal threshold for outlier detection is experimentally determined. The performance in regard to detection accuracy and network-lifetime is compared with the state-of-art techniques. Our model shows accuracy of 96.7% with a low FPR of 22.32%. In the simulation, robustness of outlier detection algorithm with respect to channel fault and robustness concerning different types of distribution of faulty communication channel is analyzed. The outperform simulation result concerning TPR and FPR is evaluated and compared with DADA. Further, the detection algorithm must detect a reading corrupted by the channel.

In the fire like event, nearby sensor nodes generate abnormal data. This, in effect, may cause some non-outlier data to be incorrectly identified as outlier. Naturally, the proposed study could be expanded to address this question of outlier event disambiguation.

References

[1] Sarangi, B., Mahapatro, A., & Tripathy, B. “Outlier Detection Using Convolutional Neural Network for Wireless Sensor Network”, International Journal of Business Data Communications and Networking (IJBDCN), 17(2), pp. 91-106. 2021. DOI: 10.4018/IJBDCN.286705.

[2] Sadia Anayat, Sheeza butt, Isma zulfiqar, Saher butt, "A Deep Analysis of Applications and Challenges of Wireless Sensor Network", International Journal of Wireless and Microwave Technologies(IJWMT), Vol.10, No.3, pp. 32-44, 2020.DOI: 10.5815/ijwmt.2020.03.03

[3] Amir F. Mukeri, Dwarkoba P. Gaikwad, "Adversarial Machine Learning Attacks and Defenses in Network Intrusion Detection Systems", International Journal of Wireless and Microwave Technologies(IJWMT), Vol.12, No.1, pp. 12-21, 2022.DOI: 10.5815/ijwmt.2022.01.02

[4] S. Rajasegarar, C. Leckie, M. Palaniswami, and J. C. Bezdek, “Quarter Sphere Based Distributed Anomaly Detection in Wireless Sensor Networks”, Proc. IEEE International Conference on Communications, pp.3864-3869, 2007.

[5] Abid, A., Kachouri, A., & Mahfoudhi, A. “Anomaly detection through outlier and neighborhood data in wireless sensor networks” in Advanced Technologies for Signal and Image Processing (ATSIP), (pp. 26–30), 2016.

[6] Chen, Y., Li, S. “A Lightweight Anomaly Detection Method Based on SVDD for Wireless Sensor Networks”. Wireless Pers Commun, 105, pp.1235-1256, 2019.

[7] T. Luo and S. G. Nagarajan, “Distributed Anomaly Detection Using Autoencoder Neural Networks in WSN for IoT,” 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, pp. 1-6, 2018.

[8] Y. Hong, U. Hwang, J. Yoo, and S. Yoon, “How generative adversarial networks and their variants work: An overview,” ACM Computing Surveys (CSUR), vol. 52, no. 1, p. 10, 2019.

[9] Van Der Maaten, L., Postma, E., Van den Herik, J.: Dimensionality reduction: a comparative review. J Mach Learn Res 10, 66–71 (2009).

[10] D. Kingma and J. Ba. “Adam: A method for stochastic optimization”. ICLR, 2015.

[11] Ch Rambabu, V.V.K.D.V.Prasad, K.Satya Prasad, " Multipath Cluster-based Hybrid MAC Protocol for Wireless Sensor Networks ", International Journal of Wireless and Microwave Technologies(IJWMT), Vol.10, No.1, pp. 1-16, 2020.DOI: 10.5815/ijwmt.2020.01.01

[12] Hitesh Mohapatra, Amiya Kumar Rath, "Survey on fault tolerance-based clustering evolution in WSN", *IET Networks*, vol.9, no.4, pp.145-155, 2020.

[13] G. Barrenetxea, Sensorscope Data. Zenodo. Dataset. 10.5281/zenodo.2654726. 2019.

[14] Hossin M, Sulaiman M. A review on evaluation metrics for data classification evaluations. Int J Data Min Knowl Manag Process. 2015;5(2):1.

[15] F. Shahzad “Pymote, 2.0: Development of an interactive python framework for wireless network simulations”, IEEE Internet of Things J., vol. 3, no. 6, pp. 1182-1188, 2016.

[16] Mohamed Sakr, Walid Atwa, Arabi Keshk, "Genetic-based Summarization for Local Outlier Detection in Data Stream", International Journal of Intelligent Systems and Applications(IJISA), Vol.13, No.1, pp.58-68, 2021. DOI: 10.5815/ijisa.2021.01.05

Authors’ Profiles

Biswaranjan Sarangi (Research Scholar) in Biju Patnaik University of Technology (BPUT), Odisha, (India) completed his M. Tech in Computer Science & Engineering from F.M. University, Balasore, Odisha and pursuing his Ph.D. in Computer Science & Engineering under BPUT. His current research interest is machine learning & AI, neural network, and wireless sensor networks. He is currently working as Asst. Professor and Head in Dept. of CSE, Synergy Institute of Technology, Bhubaneswar (India).

Biswajit Tripathy (Professor) received Ph.D. from Biju Patnaik University of Technology (2018). He is currently working as Professor and Head in Dept. of CST, GITA Autonomous College, Bhubaneswar (India). He is having total 29 years' experience both in industries and teaching in the area of Computer Science. His area of interest is E-commerce, Software Engineering, Artificial Intelligence, IOT, Software Project Management and Network Security.

How to cite this paper: Biswaranjan Sarangi, Biswajit Tripathy, "Outlier Detection Technique for Wireless Sensor Network Using GAN with Autoencoder to Increase the Network Lifetime", International Journal of Computer Network and Information Security(IJCNIS), Vol.15, No.1, pp.26-38, 2023. DOI:10.5815/ijcnis.2023.01.03