2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

# Intrusion Detection for Wireless Sensor Network Using Graph Neural Networks

Vida Gharavian*[1], Rasa Khosrowshahli*[1], Qusay H. Mahmoud[1], Masoud Makrehchi[1], Shahryar Rahnamayan*[2], SMIEEE
[1]*Department of Electrical, Computer, and Software Engineering, Ontario Tech University, Oshawa, ON, Canada*
**Nature-Inspired Computational Intelligence (NICI) Lab*
[2]*Engineering Department, Brock University, St. Catharines, ON, Canada*
{vida.gharavian, rasa.khosrowshahli}@ontariotechu.net, qusay.mahmoud@ontariotechu.ca

***Abstract*—Wireless Sensor Networks (WSNs) are rapidly employed in many applications due to highly demanded autonomous systems. These networks are of immense importance due to their ability to collect data from remote and challenging environments, their impact on various sectors like healthcare, agriculture, industry, environment, and their role in enabling smart technologies for a sustainable, secure, and connected future. Nevertheless, these systems can be attacked by adversaries. Usually, the WSNs are designed with lightweight sensor nodes with limited computation and memory resources. Therefore, employing a firewall system on every sensor node is unacceptable. This paper tackled this problem with a very lightweight Graph Neural Network-based model. The conducted experiment performed in this work demonstrates promising attack-type detection by our proposed approach to the WSN-DS dataset. In this article, our proposed method is compared with other the-state-of-the-art works, and we could discover all Blackhole attacks, one of the most common Denial-of-Service attacks.**

***Index Terms*—Wireless Sensor Network, WSN, Denial of Service, DoS, Graph Neural Network, GraphSAGE, LEACH, Intrusion Detection**

## I. Introduction

Wireless Sensor Networks (WSNs) are networks of numerous miniature, battery-operated sensors dispersed over a vast geographic region. These autonomous sensor nodes gather and transfer data wirelessly to a base station (BS), making them suitable for various real-world applications such as environmental monitoring, industrial automation, and health monitoring [1], [2]. In WSNs, a protocol is specialized to manage data transmission across the sensor network. Some external and internal threats can challenge the WSN's protocol in a way that can damage the network components and/or corresponding services.

Sensor nodes in WSNs consist of limited resources, including battery capacity, memory size, and processing power [2]. The batteries that generally power up the sensors in a WSN have a finite lifespan [3]. As a result, energy management is essential for the network's long-term functionality. Battery power can be preserved using strategies like duty cycling and sleep schedule [4]. Low-power microprocessors with constrained processing power are frequently used in WSNs. Complex processes like data processing and analysis on the sensors themselves may be challenging as a result [5]. A WSN's range is constrained by the sensitivity of the receivers and the transmission power of the sensors. In order to transfer data over long distances, the network might need to make several hops [6]. Interference from other devices using the same frequency range as WSNs and background noise from the surroundings can interfere with the wireless signals utilized by WSNs [7]. WSNs can be challenging to scale up or down since adding or deleting sensors; which might alter the architecture of the network and its performance [8]. Different factors, including hardware problems and battery depletion, might cause sensor nodes to malfunction [9]. The network should be built in a way to handle these faults and maintain continuous functioning smoothly. To this end, creating a firewall system on each sensor node is not a possible solution to protect them from external attacks to network system.

Additionally, since WSNs are required to broadcast packets on a regular basis, sensor nodes can be dispersed around the environment at random, making it simple for a malevolent WSN adversary. [10]. Attack types in WSNs can be categorized as follows: (1) compromising sensor nodes such as extracting cryptographic secrets or modifying sensor functionality; (2) eavesdropping messages; (3) injection of fake messages; (4) modification of data integrity; and (5) waste network resources. An attacker who compromises a sensor node may gain access to confidential information and command over the node's functions. As a result, the attacker may be able to listen to messages sent by the node, intercept them, and possibly utilize the data for bad intentions. The attacker may also try to trick the network by injecting false messages, which could lead to malfunctions or problems. The network may interpret sensor data incorrectly if the integrity of the data is compromised, producing false or misleading findings. In crucial applications where the data is utilized to make decisions that can affect people's life and safety, this can have catastrophic repercussions. Wasting network resources can make the network crowded, making it difficult for legitimate nodes to send data, and degrading the network's overall performance. This can impair the operation of the entire network and result in delays and data loss.

Another famous attack in WSNs is when several systems attack to flood the bandwidth or resources of sensor nodes to interrupt or suspend the services provided by WSNs [11].

978-0-7381-4408-5/23/$31.00 ©2023 IEEE

An alternative way to detect unknown and known attacks on WSN and alert the other nodes in the system is implementing an Intrusion Detection System (IDS). The IDS is a method for attack detection, making it one of the essential defense layers. An IDS allows the detection of anomalous activities to trigger an emergency call when needed. The IDSs are not usually implementable on sensor nodes due to the aforementioned limitations; therefore, it is required to implement them on the base station, which collects data from WSN. An example of IDS could be found in work by Mohammed Otair et al., where they proposed an improved optimization algorithm, combining the Grey Wolf Optimizer (GWO) and Particle Swarm Optimizer (PSO), for enhancing IDS in WSNs [12]. By this assumption, we are faced with unknown attack challenges which can be tricky and deceptive when they design similarly to normal profiles. In addition, the response time from IDS should be quick in real-time; otherwise, it would be failed to detect and secure the network against the attack.

Almomani et al. [13] assembled a specialized WSN dataset, called WSN-DS, to characterize four types of DoS attacks in addition to a normal state when a node is not attacked. They challenged themselves based on a routing Low Energy Aware Cluster Hierarchy (LEACH) protocol which is one of the most widely used hierarchical protocols used in WSNs. Although it is simple in design, sensor nodes in this protocol consume lower energy.

The inherent architecture of WSNs renders them highly suitable for Graph Networks. As WSNs are fundamentally structured as graphs, employing Graph Neural Networks (GNNs) becomes a natural choice to analyze and address challenges associated with networks. In addition, GNNs offer a powerful approach to processing graph-based data efficiently; thus making them well-suited for handling various issues in WSNs. Xu et al. [14] proposed an approach that aims to enhance the efficiency and performance of WSN by leveraging GNN's ability to handle graph-structured data and Deep reinforcement learning (DRL) capability to learn from the environment to make better decisions. By integrating these methodologies, the article addresses optimization challenges in WSN, potentially leading to improved network configurations, resource allocation, and overall system performance. Biswas et al. [15] enhanced intrusion detection in wireless sensor networks (WSNs) by combining the capabilities of Graph Neural Networks (GNNs) and Lyapunov optimization. The proposed approach leverages the inherent graph structure of WSNs, enabling efficient representation and analysis of network data using GNNs. GNNs are adept at handling graph-based data, making them well-suited for intrusion detection in WSNs.

The rest of this paper is organized as follows. Section II provides some background information to make the paper self-contained. The related works are discussed in Section III. The proposed method and the experimental results are presented in Section IV. Experimental analyses are discussed in Section V. Finally, conclusion remarks and future work are presented in Section VI.

## II. Background Review

This section provides a background review of the GNN model concepts, and terminologies used.

### A. Graph Neural Networks (GNNs)

A set of connectivity-driven models that have been addressing the need for deep geometric learning are proposed as GNNs [16]. They adapt their model to the structure of an input graph and, through an iterative process, capture the complex dependencies of the underlying system. This allows the prediction of properties for specific nodes, connections, or the graph as a whole, and generalizes to unseen graphs [17]. Consider a graph $G = (V, E)$ is defined where $V$ is a set of nodes and $E$ is the set of edges. The data is converted into graph $G$ where each node $V$ represents entities and each edge provides $E$ relationships between them.

### B. GraphSAGE

GraphSAGE is a propagation module based on a convolutional operator. This technique effectively creates node embedding for previously undiscovered material by utilizing node feature information (e.g., text properties) [18]. This type of model was used in our proposed GNN model. The method trains a function that creates embedding by sampling and aggregating information from a node's local neighborhood rather than training distinct embedding for each node. This algorithm consists of four main components as follows:

- **Embedding generation:** This step is the forward propagation of the algorithm. They assume that the weights and parameters are fixed. In each iteration, or search depth, nodes aggregate information from their local neighbors, and as this process iterates, nodes incrementally gain more information.
- **Learning the parameters of GraphSAGE:** for learning parameters with the unsupervised setting, they applied a graph-based loss function to tune weight matrices and parameters of the aggregator functions via stochastic gradient descent. The loss function of GraphSage, which was used to learn the weight of the aggregator and embedding for two neighbors $u$ and $v$ defined in Eq. 1:

$$L_{\varepsilon}(u, v) = -\log \sigma(z_u^T z_v) - N E_{v_n \sim P_n(v)} \log \sigma(-z_u^T z_v) \tag{1}$$

  where $\sigma$ is a sigmoid function and $N$ is the number of negative nodes. While $P_n$ is the negative sampling distribution, $v_n$ is the negative sample. The output representation is $z_u, \forall u \in v$, and $z_u^T$ is the transposed matrix of this matrix.
- **Aggregator:** The aggregator must operate on an unordered set of vectors since the node neighbors have no natural ordering. They proposed three types of aggregators that can be applied to arbitrarily ordered node neighborhood feature sets.
- **Mean aggregator:**. The mean aggregator is similar to the convolutional propagation rule used in the transductive

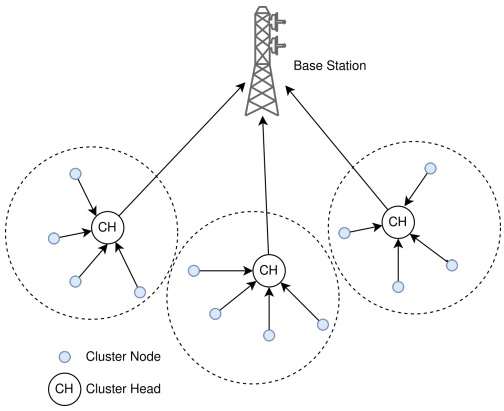


Fig. 1. Routing schema in LEACH protocol, where the transformation flow is from cluster nodes to cluster heads and cluster heads to base stations. There is no direct path from cluster nodes to base stations.

graph convolutional networks (GCN) framework [19]. This aggregator simply takes the element-wise mean $(mean)$ of the vectors in $h_u^{k-1}, \forall u \in N(v)$ which is formulated in Eq. 2:

$$h_v^k \leftarrow \sigma(W \cdot \text{MEAN}(\{h_u^{k-1}\} \cup h_u^{k-1}, \forall u \in N(v)) \quad (2)$$

Where the node's previous layer representation is $h^{k-1}$ with the aggregated neighborhood vector $h_{N(v)}^k$ and $W$ is the weight matrix.

### C. WSNs and the Denial-of-Service Intrusion

WSNs have a large number of distributed sensor nodes that are responsible for gathering the necessary information and transmitting them across the network wirelessly. In general, there are three types of nodes in WSNs, namely, sensor nodes, Cluster Heads (CH), and Sink or Base Stations (BS). Low Energy Adaptive Clustering Hierarchy (LEACH) is a protocol used in wireless sensor networks (WSNs) to reduce node energy consumption. It is a hierarchical protocol in which nodes are organized into clusters, and a node is elected as a cluster head to act as a coordinator for the cluster. Fig. 1 illustrates a simple view of the node-graph in LEACH protocol.

Many applications use these kinds of networks, for example, weather predictions crime investigation, forest fire detection [20], and medical purposes which have crucial impacts on human life [21]. Because of the importance of data integrity transmitted through the WSNs, they are always facing potential attacks. Among these kinds of attacks, we can point to Denial-of-Service (DoS) intrusions. The aim of DoS is to take control of the network by reducing and eliminating the network's capacity. A common point of most attacks in WSN is to target either the CHs or other sensors to interrupt their connection. In the following step, the attacks are in detail.

- **Flooding attack**. Flooding attacks are multi-way attacks that can occur by sending or receiving a significant number of advertising CH messages. As this attack has high transmission power, sensors will be drained due to spending time and energy on deciding which CH to join. This attack can be more harmful when it comes to multiple hops from the attacked CH [13].
- **Blackhole attack**. This attack is made when the WSN system tries to initialize CHs by sending advertisement messages to familiarize itself as CH to the other surrounding sensor nodes [13].
- **Grayhole attack**. The goal of Grayhole attacks is to randomly avoid some packets from being received by BS from CH. Once attackers advertise themselves as CH to the network, they can drop packets that are sent by other nodes. Therefore, BS would miss those packets and lose the connection from sensor nodes [13].
- **Scheduling (TDMA) attack**. This attack usually occurs in the configuration of the LEACH protocol. Each CH creates a Time Division Multiple Access (TDMA) schedule due to the number of nodes that are connected to it. This attack would change the TDMA and set the same time slot for all connected nodes. Therefore the behavior of TDMA will be changed from broadcast to unicast, and it would cause packet collision [13].

## III. Related works

Wireless sensor network security is a very active area of research. The first IDS monitoring system was proposed by Almomani et al. [13] and they introduced the WSN-DS dataset. They used a multi-layer perceptron neural network classifier to classify datasets using necessary features to detect every DoS attack, such as Blackhole, Grayhole, Flooding, and Scheduling. They evaluated the performance of their supervised learning work based on several metrics, namely, accuracy, precision, recall, F1-score, time, etc. Furthermore, Alsulaiman and Al-Ahmadi [22] published a work that evaluated various machine learning techniques such as Naive Bayes (NB), Support Vector Machine (SVM), J48 Decision Tree, and Random Forest (RF) classifier to WSN-DS dataset. Another similar work compared other classifiers such as $k$-Nearest Neighbors (KNN), AdaBoost (AB), Gaussian Naive Bayes (GNB), and Stochastic Gradient Descent (SGD) classifiers on different data splits. [23]

Redundant features in a dataset may cause classification to be inaccurate due to irrelevant information about the characteristics of the attack. In order to select useful features, two studies applied feature selection algorithms before the classification phase. Recently, Ismaeil et al. [24] used Pearson Correlation and Mutual Information feature selectors before KNN, Gaussian Naive Bayes (GNB), Gradient Boosting (GBM), Light Gradient Boosting (LGBM), Catboost (CB) algorithms, and Random Forest classification algorithms. In another work, Mahbooba et al. [25] used Correlation Matrix algorithm to select the best features to be evaluated by Decision Tree (DT), KNN, Random Forest (RF), Naive Bayes (NB), and Deep Neural Network (DNN) structures which include Long Short-Term Memory (LSTM) and Gated Recurrent Unit.

## IV. Proposed Method

We define our problem as a undirected $Graph(V, E)$. This problem is a node classification task. For this purpose, we demonstrate GraghSAGE as our utilised computational module, which is responsible for extracting the hidden features from nodes and edges between them.

Fig. 2 illustrates the model structure. We design a two-layer GraphSAGE to tackle node classification tasks. The nature of our problem allows for a simple definition in a graph. This problem can be addressed in five steps as follows

- **Data preprocessing**: In this step, rows with missing data are removed, and categorized data such as attack type is transformed to numeric form. Also, the unconnected nodes that are not in any clusters are removed.
- **Graph representation**: The graph representation is constructed, where each node $i$ possesses specific features denoted as $x_i$, and it is included in the set $V$ within the graph structure $Graph(V, E)$. When a node joins a cluster with a cluster head $ch_j$, their connection is recorded as the edge of $e_{(j,i,1)}$ where $e$ is the edge attributes, indicating the connection type and the corresponding node ids. The graph is designed to be undirected for this particular problem, and to ensure symmetry, the reverse form of the graph is also incorporated.
- **Sampling and split data**: After creating train, test and validation masks. which indicated which part of the dataset belongs to the training set, test set, or validation set. We need to create a data loader that has the responsibility to make a tairn set and validation set in different batch sizes. We are using neighbor sampling for this purpose. For destination node $i$ , a fixed set of neighborhoods, $u_k$, is sampled in Eq. 4:

$$u^0 = v, k = 0 \tag{3}$$

$$u^k = \cup_{v \in u^{k-1}} S(A_v, N^k), \qquad k = 1, 2, ..., K; \tag{4}$$

  where $A_v$ is a set of neighboring nodes of $V$ and $N^k$ is the sample size at depth $k$, and $S(A_v, N^k)$ is the sampler from a uniform distribution $U(1, deg(v))$ as a default setting [26]. For depth $k = 0$ , fixed set of neighborhoods is equal to the same node.
- **Train GNN model**: The details of GNN layers are presented in Table II. It includes two layers of GraphSage, which is responsible for feature selection or node embedding. This problem is a node classification task that tries to categorize nodes into several classes [27], and labels are the attack type for each node. Since that is a supervised task, after node embedding we need to define a loss function, and we used cross entropy for this purpose. Therefore, for each class, we will have a probability. The class with a higher probability will be considered as a label.
- **Evaluate GNN model**: After training the entire set of models, we will merge the validation set and test set to form a combined dataset, which will be used for evaluating the model. To assess the performance of our model, we employ the accuracy metric, calculated in Eq. 5:

$$Accuracy = \frac{(TN + TP)}{(TN + TP + FN + FP)} \tag{5}$$

  where true positive is (TP), true negative is (TN), false negative is (FN), and false positive is (FP). By comparing the predicted labels with the true labels for each instance in the combined dataset, we count the number of correctly classified instances and then determine the accuracy by dividing this count by the total number of instances in the dataset. It is important to note that accuracy is just one of several evaluation metrics, and depending on the nature of the problem and the data, other metrics may be more appropriate for assessing the model's effectiveness, such as precision, recall, F1 score, AUC-ROC, etc. These metrics were calculated to offer a more comprehensive understanding of the model's performance in various aspects.

The graph structure and algorithms are implemented using DGL [28]. This library is a powerful tool for implementing graph data structure and algorithms related to it. DGL offers a robust graph object that can be installed on either the CPU or the GPU. For improved control, it combines structural data with characteristics. They offer several methods for working with graph objects, such as practical and programmable primitives for passing messages in Graph Neural Networks.

## V. Experimental Analysis

This section presents and discusses the experiments which are conducted on WSN-DS dataset. This dataset is gathered from a public dataset initially published by Almomani et al. [13]. The dataset is constructed using the LEACH protocol. In order to gather the required data, NS-2 simulation was used [29]. Table V provides a detailed overview of the simulation parameters.

### A. WSN-DS Dataset

Almomani et al. [13], studied deeply on LEACH routing protocol to extract 23 features to aid in determining each network node's condition as they are described in Table VI

Finally, the last feature is the corresponding label for the node whether it is in normal mode or otherwise, the attack type is declared. The dataset is provided with 374,661 sample nodes where the data is separated into 60% training and 40% test subsets. The detail of each class distribution is summarized in Table IV.

### B. Numerical Analysis

This algorithm was run over 1000 epochs, and you can find more about the accuracy and loss of this method over each epoch in Fig.s 3 and 4. After 300 epochs, the model has achieved its best F1 score, but the train loss indicates that higher epochs should also yield better outcomes. Furthermore, we used a confusion matrix to evaluate the performance of our proposed classification approach on the test data, which is illustrated in Fig. 5. The detail of F1-score, precision,

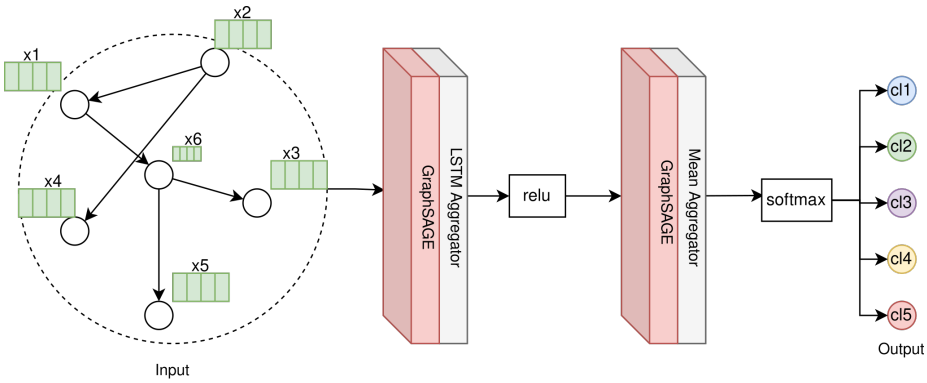


Fig. 2. Proposed GNN method structure for node embedding and node classification in attack detection

TABLE I
DETAILED COMPARISON OF TEST RESULTS BETWEEN THE PROPOSED METHOD AGAINST REGRESSION (LR), NAIVE BAYES (NB), K-NEAREST NEIGHBORS(KNN), DECISION TREE (DT), ADABOOST (AB), RANDOM FOREST (RF), SUPPORT VECTOR MACHINE-RADIAL BASIS FUNCTION (SVMRBF), AND DEEP NEURAL NETWORK (DNN) WITH DIFFERENT NUMBER OF LAYERS. THE PROPOSED METHOD OUTPERFORMS OTHER MACHINE LEARNING CLASSIFIERS ON THE MAJORITY OF ATTACKS

| Method | Normal | | | Blackhole Attack | | | Grayhole Attack | | | Flooding Attack | | | Scheduling Attack | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TPR | FPR | Acc | TPR | FPR | Acc | TPR | FPR | Acc | TPR | FPR | Acc | TPR | FPR | Acc |
| **LR** | 0.998 | 0.147 | 0.934 | 0.159 | 0.048 | 0.844 | 0.609 | 0.145 | 0.807 | 0.755 | 0.0 | 0.988 | 0.702 | 0.0 | 0.973 |
| **NB** | 0.946 | 0.066 | 0.946 | 0.989 | 0.146 | 0.87 | 0.478 | 0.039 | 0.875 | 0.8385 | 0.017 | 0.976 | 0.288 | 0.0 | 0.937 |
| **KNN** | 0.992 | 0.341 | 0.838 | 0.485 | 0.011 | 0.929 | 0.387 | 0.0888 | 0.809 | 0.403 | 0.007 | 0.969 | 0.664 | 0.0 | 0.96 |
| **DT** | 0.996 | 0.054 | 0.97 | 0.933 | 0.004 | 0.98 | 0.945 | 0.014 | 0.981 | 0.702 | 0.0 | 0.986 | 0.939 | 0.0 | 0.996 |
| **AB** | 0.999 | 0.085 | 0.964 | 0.885 | 0.008 | 0.977 | 0.839 | 0.019 | 0.957 | 0.984 | 0.0 | 0.999 | 0.847 | 0.0 | 0.98 |
| **RF** | 0.999 | 0.034 | 0.98 | 0.961 | 0.0 | 0.991 | **0.958** | 0.0 | **0.986** | 0.837 | 0.0 | 0.998 | 0.942 | 0.0 | 0.997 |
| **SVM-rbf** | 0.999 | 0.922 | 0.575 | 0.0 | 0.0 | 0.866 | 0.142 | 0.0 | 0.833 | 0.014 | 0.0 | 0.956 | 0.075 | 0.0 | 0.918 |
| **DNN 1 layer** | 0.998 | **0.027** | 0.98 | 0.965 | 0.078 | 0.939 | 0.616 | 0.007 | 0.919 | 0.978 | 0.005 | 0.994 | 0.917 | 0.0 | 0.992 |
| **DNN 2 layers** | 0.999 | 0.091 | 0.957 | 0.754 | 0.073 | 0.904 | 0.538 | 0.046 | 0.87 | 0.754 | 0.0 | 0.989 | 0.937 | 0.0 | 0.993 |
| **DNN 3 layers** | 0.999 | 0.107 | 0.953 | 0.883 | 0.037 | 0.956 | 0.666 | 0.026 | 0.916 | 0.776 | 0.0 | 0.987 | 0.916 | 0.0 | 0.992 |
| **DNN 4 layers** | 0.998 | 0.145 | 0.933 | 0.862 | 0.071 | 0.92 | 0.474 | 0.033 | 0.873 | 0.648 | 0.0 | 0.984 | 0.796 | 0.0 | 0.983 |
| **DNN 5 layers** | 0.994 | 0.047 | 0.975 | 0.946 | 0.069 | 0.939 | 0.676 | 0.019 | 0.925 | 0.819 | 0.0 | 0.998 | 0.879 | 0.0 | 0.988 |
| **Proposed Method** | **0.999** | 0.031 | **0.996** | **0.998** | 0.016 | **0.984** | 0.530 | **0.0** | 0.982 | **0.923** | **0.0** | **0.999** | **0.941** | **0.0** | **0.999** |

TABLE II
GNN MODEL PARAMETER SETTINGS

| Model Name | Type | Parameters |
|---|---|---|
| Convolution1 | SAGEConv | In size: 128, Out size: 64 |
| Convolution2 | SAGEConv | In size: 64, Out size: 5 |
| Optimizer | Adam | lr: 0.01, $\alpha$: 0.9, $\beta$: 0.999 |

TABLE III
TEST DATA CLASSIFICATION TABLE INCLUDING PRECISION, RECALL, AND F1-SCORE. THE MACRO AVERAGE IS CALCULATED USING THE ARITHMETIC MEAN (A.K.A. UNWEIGHTED MEAN) OF ALL THE PER-CLASS F1 SCORES, WHEREAS, THE WEIGHTED AVERAGE IS CALCULATED BASED ON THE SIZE OF EACH CLASS SUPPORT VALUE.

| Attack Types | Performance Metric | | | |
|---|---|---|---|---|
| | Precision | Recall | F1-score | Support |
| Blackhole | 0.663 | **0.998** | 0.796 | 4635 |
| Flooding | 0.963 | 0.923 | 0.943 | 1429 |
| Grayhole | 0.957 | 0.530 | 0.682 | 5351 |
| Normal | 0.997 | **0.999** | 0.998 | 135539 |
| Scheduling | 0.994 | 0.941 | 0.967 | 2911 |
| Accuracy | 0.980 | 0.980 | 0.980 | 0.980 |
| Macro Average | 0.915 | 0.878 | 0.877 | 149865 |
| Weighted Average | 0.985 | 0.980 | 0.979 | 149865 |

TABLE IV
DATASET SEPARATED 60% TRAINING SET AND 40% TEST SET.

| Attack type | Training set (60%) | Test set (40%) |
|---|---|---|
| Blackhole | 5,414 | 4,635 |
| Grayhole | 9,245 | 5,351 |
| Flooding | 1,883 | 1,429 |
| Scheduling | 3,727 | 2,911 |
| Normal | 204,527 | 135,539 |
| Total | 224,796 | 149,865 |

TABLE V
NS-2 SIMULATION PARAMETER SETTING

| Parameter | Value |
|---|---|
| Number of nodes | 100 nodes |
| Number of clusters | 5 |
| Network area | 100 m × 100 m |
| Base station location | (50, 175) |
| Size of data packet | 500 bytes |
| Size of packet header | 25 bytes |
| Maximum transmission range | 200 m |
| Routing protocol | LEACH |
| MAC protocol | CSMA/TDMA |
| Simulation time | 3600 s |
| Initial energy (in joule) | 5, 50 |
| Attackers' intensities | 10%, 30%, 50% |

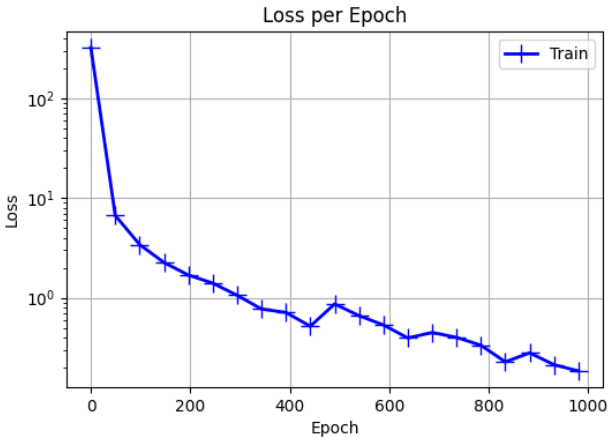


Fig. 3. Model loss on train data convergence plot over 1000 epochs which is sampled every 50 epochs.

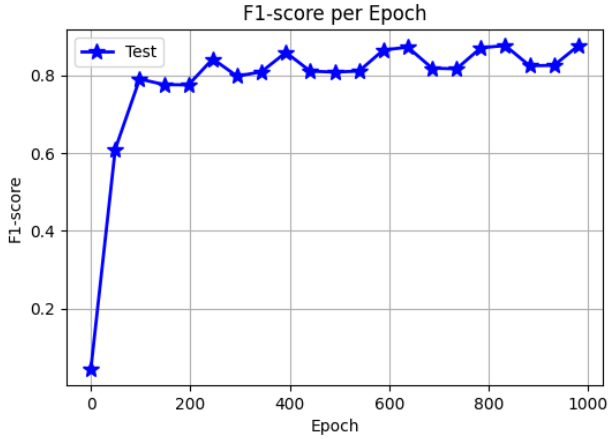


Fig. 4. Model F1-score on test data with 149,865 records, over 1000 epochs which is sampled every 50 epochs

and recall on the test data are presented in Table III. Our results were compared with Vinayakumar et al. [30], and the overall comparison is documented in Table I. As seen from Table I, we evaluated test data based on TPR, FPR, and Acc metrics. The column Method shows the nominated methods that are compared to our proposed work, including Linear Regression (LR), Naive Bayes (NB), $k$-Nearest Neighbors (KNN), Decision Tree (DT), AdaBoost (AB), Random Forest (RF), Support Vector Machine-radial basis function (SVM-rbf), and their proposed Deep Neural Network (DNN) model with 1-5 numbers of hidden layers. The proposed model could distinguish the sensor nodes from the others presenting in the Normal state. The difference between our proposed method to existing works is that we are not only embedding the features from nodes, but also we are embedding the link between them. Moreover, we could detect most likely every sensor node that is attacked by the Blackhole attack type. The FPR metric shows our work has fewer falsely labeled cases; for example, the FPR on flooding and scheduling attacks is zero. Looking

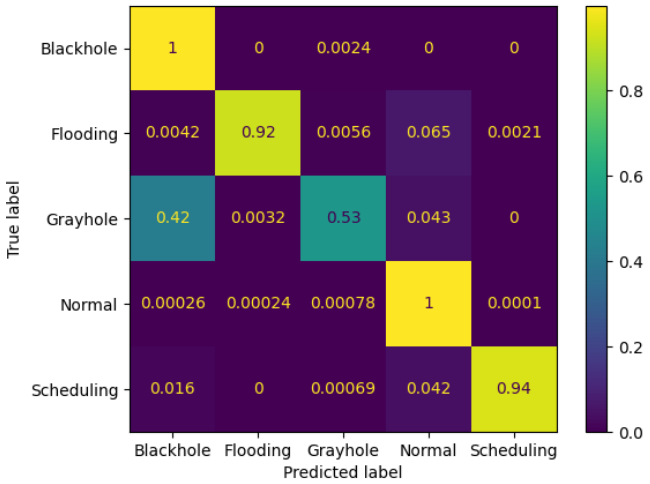


Fig. 5. Confusion matrix also known as the error matrix on test data with five classes. Each row of the matrix represents the instances in an actual class while each column represents the instances in a predicted class. All correct predictions are located in the diagonal of the table (highlighted in yellow), so it is easy to visually inspect the table for prediction errors, as values outside the diagonal will represent them.

TABLE VI
WSN-DS DATASET FEATURE DESCRIPTION

| Feature | Description |
|---|---|
| Node ID | Every node has a unique ID to be distinguished in the network |
| Time | The node's current simulation period |
| IsCH | A state to show the CH node |
| WhoCH | Node's related CH |
| RSSI | Strength of received signal between the node and the related CH |
| DistToCH | The distance value between the node and the related CH |
| MaxDistToCH | It is the maximum length between CH and nodes in the related cluster |
| AverageDistToCH | It is the average length between CH and nodes in the related cluster |
| CurrentEnergy | The energy level of each node |
| EnergyConsumption | The amount of energy consumed by the node |
| ADV-CH send / receives | The number of advertised messages sent / received by the node |
| ADV-SCH send / receives | The quantity of TDMA schedule messages that CHs sent/receives |
| send / receives | The number of join requests sent/received by the CH node |
| RANK | The TDMA schedule position of the node |
| DataSent | The total number of data packets sent from a node to the related CH |
| DataReceived | The total number of data packets received from CH |
| DataSentToBS | The quantity of data packets sent to the BS from CH |
| SendCode | The related cluster code |
| DistCHToBS | The distance between the CH and BS. |

at the Grayhole attack column, our proposed method could not outperform the Random Forest classifier; however, the difference in accuracy metric is 0.004. The results show that our proposed method outperforms the majority of Machine Learning classifiers on the attack types, including Blackhole, Flooding, and Scheduling.

## VI. Conclusion and Future Work

Wireless Sensor Network (WSN) is vulnerable to external and internal intrusions. The structural limitations in this network are caused by the constrained resources of sensor nodes, making it easy to be defenseless in front of cyber attacks. A solution is to conduct a machine-learning-based intrusion detection system on the base station, which collects data from cluster heads and alerts the system if an anomaly is detected. This intrusion detection system has to be dependable on its prediction. In this paper, we proposed a new machine learning technique to alleviate the critical DoS attack types using a proposed small and simple Graph Neural Network architecture. Based on performance analysis, we could detect nearly every node that is poisoned by the Blackhole attack type as well as other attack types, such as Flooding and Scheduling. In comparison to other previous works, we could outperform deep neural networks with many hidden layers. As one of the criteria of WSNs is to be lightweight and quick, our proposed graph neural network architecture uses smaller layers and parameters. Due to using the GraphSAGE structure, our proposed strategy is scalable to the newly expanded parts of WSN, and it is not required to re-train the whole network. We are aware of how crucial this problem is; thus, in the future, we plan to present it as a heterogeneous graph with a variety of node types and different feature vectors for the cluster heads and their attached nodes in order to improve the model to avoid false predictions. Additionally, we have plan to test it using various datasets and attacks.

## References


[1] V. C. Gungor, B. Lu, and G. P. Hancke, "Opportunities and challenges of wireless sensor networks in smart grid," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 10, pp. 3557–3564, 2010.

[2] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 266–282, 2014.

[3] W. Guo and W. M. Healy, "Power supply issues in battery reliant wireless sensor networks: A review," 2014.

[4] N. Qi, K. Dai, F. Yi, X. Wang, Z. You, and J. Zhao, "An adaptive energy management strategy to extend battery lifetime of solar powered wireless sensor nodes," *IEEE Access*, vol. 7, pp. 88289–88300, 2019.

[5] P. Mareca and B. Bordel Sánchez, "Robust hardware-supported chaotic cryptosystems for streaming commutations among reduced computing power nodes," *Analog Integrated Circuits and Signal Processing*, vol. 98, 01 2019.

[6] J. Guo and H. Jafarkhani, "Sensor deployment with limited communication range in homogeneous and heterogeneous wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 10, pp. 6771–6784, 2016.

[7] M. Haenggi and R. K. Ganti. 2009.

[8] L. Alazzawi and A. Elkateeb, "Performance evaluation of the wsn routing protocols scalability," *Journal of Computer Systems, Networks, and Communications*, vol. 2008, 01 2008.

[9] M. Shyama and A. Pillai, *Fault-Tolerant Techniques for Wireless Sensor Network—A Comprehensive Survey*, pp. 261–269. 01 2019.

[10] H. Modares, R. Salleh, and A. Moravejosharieh, "Overview of security issues in wireless sensor networks," in *2011 Third International Conference on Computational Intelligence, Modelling Simulation*, pp. 308–311, 2011.

[11] Z. Huanan, X. Suping, and W. Jiannan, "Security and application of wireless sensor network," *Procedia Computer Science*, vol. 183, pp. 486–492, 2021. Proceedings of the 10th International Conference of Information and Communication Technology.

[12] M. Otair, O. T. Ibrahim, L. Abualigah, M. Altalhi, and P. Sumari, "An enhanced grey wolf optimizer based particle swarm optimizer for intrusion detection system in wireless sensor networks," *Wireless Networks*, vol. 28, Feb 2022.

[13] I. Almomani, B. Kasasbeh, and M. AL-Akhras, "Wsn-ds: A dataset for intrusion detection systems in wireless sensor networks," *Journal of Sensors*, vol. 2016, pp. 1–16, 01 2016.

[14] X. Xu, Y. Lu, and Q. Fu, "Applying graph neural network in deep reinforcement learning to optimize wireless network routing," in *2021 Ninth International Conference on Advanced Cloud and Big Data (CBD)*, pp. 218–223, 2022.

[15] P. Biswas, T. Samanta, and J. Sanyal, "Intrusion detection using graph neural network and lyapunov optimization in wireless sensor network," *Multimedia Tools and Applications*, vol. 82, pp. 14123–14134, Apr 2023.

[16] M. Gori, G. Monfardini, and F. Scarselli, "A new model for earning in raph domains," vol. 2, pp. 729 – 734 vol. 2, 01 2005.

[17] S. Abadal, A. Jain, R. Guirado, J. López-Alonso, and E. Alarcón, "Computing graph neural networks: A survey from algorithms to accelerators," 2021.

[18] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," 2018.

[19] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[20] U. Dampage, L. Bandaranayake, R. Wanasinghe, K. Kottahachchi, and B. Jayasanka, "Forest fire detection system using wireless sensor networks and machine learning," *Scientific Reports*, vol. 12, p. 46, Jan 2022.

[21] H. Liang, S. Yang, L. Li, and J. Gao, "Research on routing optimization of wsns based on improved leach protocol," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, pp. 1–12, 2019.

[22] L. Alsulaiman and S. Al-Ahmadi, "Performance evaluation of machine learning techniques for dos detection in wireless sensor network," 2021.

[23] M. Kurtkoti, B. S. Premananda, and K. Vishwavardhan Reddy, "Performance analysis of machine learning algorithms in detecting and mitigating black and gray hole attacks," in *Innovative Data Communication Technologies and Application* (J. S. Raj, K. Kamel, and P. Lafata, eds.), (Singapore), pp. 945–961, Springer Nature Singapore, 2022.

[24] S. Ismail, T. T. Khoei, R. Marsh, and N. Kaabouch, "A comparative study of machine learning models for cyber-attacks detection in wireless sensor networks," in *2021 IEEE 12th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, pp. 0313–0318, 2021.

[25] B. Mahbooba, R. Sahal, M. Serrano, and W. Alosaimi, "Trust in intrusion detection systems: An investigation of performance analysis for machine learning and deep learning models," *Complexity*, vol. 2021, p. 23, 03 2021.

[26] J. Oh, K. Cho, and J. Bruna, "Advancing graphsage with a data-driven node sampling," *arXiv preprint arXiv:1904.12935*, 2019.

[27] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[28] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, and Z. Zhang, "Deep graph library: A graph-centric, highly-performant package for graph neural networks," *arXiv preprint arXiv:1909.01315*, 2019.

[29] "The network simulator—ns-2." http://www.isi.edu/nsnam/ns/.

[30] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019.