Frontiers in Health Informatics
ISSN-Online: 2676-7104
www.healthinformaticsjournal.com
2024; Vol 13: Issue 3
Open Access

Security Enhancement through Intrusion Detection Systems in Wireless Mesh Networks

[1]Jaffar Amin chacket,[2]Khalid Hafiz mir, [3]Anzar Hussain Lone, [4]G.Akilarasu, [5]Rajeshkumar J
1Assistant Professor, Department of Computer Science and Engineering, Lovely Professional University, Phagwara, Punjab 144001, India
2Assistant Professor, Department of Computer Science and Engineering, Lovely Professional University, Phagwara, Punjab 144001, India
3Assistant Professor, Department of Computer Science and Engineering, Lovely Professional University, Phagwara, Punjab 144001, India
4Associate Professor, Department of Computer Science and Engineering, Lovely Professional University, Phagwara, Punjab 144001, India
5Assistant Professor, School of computing and Information Technology, REVA University, Yelahanka, Bangalore 560064, India
[1]Princejaffar24@gmail.com,[2]mirkhalid081@gmail.com,[3]anzarhussainlone@gmail.com
[4]akilarasu.26014@lpu.co.in,[5]rajeshkumar.j@reva.edu.in

Cite this paper as: Jaffar Amin chacket ,Khalid Hafiz mir, Anzar Hussain Lone,G.Akilarasu R,ajeshkumar J (2024) Security Enhancement through Intrusion Detection Systems in Wireless Mesh Networks. *Frontiers in Health Informatics, 13*(3),1013-1030.

Abstract

Any unauthorized action that interferes with the regular functioning of a wired or wireless network is referred to as an intrusion. Wireless mesh networking (WMN) technology has been essential in providing people with ubiquitous access to the Internet at a reasonable and affordable price. Such networks provide universal, high-speed, and cost-effective connection, which is critical for many public services. Malicious attacks, particularly in multi-hop environments, can take advantage of WMNs' decentralized design and accessible media. As a result, it is vital to design these networks with privacy, security, and resiliency. Intrusion detection systems (IDS) are a great way to detect both internal and external attacks. In this work, we used modern technologies such as machine learning to automate IDS. We used publicly available CICIDS2017 data downloaded from the internet for IDS purposes. The raw data was pre-processed to prepare it for analysis. Significant features were chosen using three separate algorithms: Mutual Information (MI), Correlation-based Feature Selection (CFS), and Particle Swarm Optimization (PSO). These selected features were then fed into Machine Learning (ML) models, specifically Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest (RF), and Attention-based AutoEncoder (AAE). Accurate IDS relies on determining the best combination of Feature Selection (FS) and ML model. Through experimental study, we discovered that the combination of PSO and AAE produced the best accuracy of 99.54%, with the lowest false prediction rates of 0.48% and 0.44%. Following closely, the combination of PSO and RF yielded higher results, with an accuracy of 98.9%.

Keywords— Intrusion, Wireless Network, Machine Learning, Feature Selection, Optimization, Performance Metrics

Introduction

WMN designs are divided into two categories: infrastructure-based and infrastructure-less [1]. To provide broadband service coverage over a large region, an infrastructure-based WMN uses mesh clients, gateways, and routers. With the multi-hop connection mechanism, mesh routers may be easily set up and expanded to cover large areas [2]. The further you are from the gateway, the slower your internet speed will be. Both mobile and stationary nodes can be supported by WMNs with the right configuration. Ad hoc networks, on the other hand, can function without mesh routers or gateways, in contrast to infrastructure-free WMNs. Instead, these WMNs establish routing capabilities between source and destination nodes through a multi-hop pathway. In situations characterized by frequent movement and change, infrastructure-free WMNs are preferable [3].

Infrastructure-less WMNs are limited in scope due to memory restrictions, processing speed constraints, limited bandwidth, and energy consumption. As a result, it is preferable to incorporate WMN characteristics that are more energy-efficient and consume fewer resources [4]. Conventional WMNs are vulnerable to a wide range of active and passive network security risks, including Denial of Service attacks. Multi-hop decentralized and heterogeneous networks, such as WMNs, have several features and weaknesses that academics are working to address. A secure network is built on three pillars: availability, integrity, and privacy. At present, WMNs may be safeguarded in a variety of methods. However, many solutions are insufficient, either because they are overly specific to a single application or because they fail to handle a wide range of security risks. It's worth mentioning that the bulk of the proposed solutions have centered on the network layer of WMN. Network layer security approaches can manage a few routing attacks, but they cannot handle physical, transport, or Medium Access Control (MAC) attacks [5].

A lightweight IDS is ideal for WMNs as it can detect and prevent a variety of security threats [6]. An IDS is not the ideal initial line of defense against network security breaches due to its passive nature. IDSs detect security risks and alert relevant parties when triggered by programmed alarms. In reality, there are two main forms of IDS. Rule-based IDSs get their information from a database of previously documented attack signatures. In contrast, anomaly-based IDSs monitor network patterns and identify any deviations from the usual potential intrusions. When identifying IDS using the provided detection method, a third kind may be seen. Specification-based IDSs identify suspicious activity by labeling certain program behaviors. IDS affects all three levels of the OSI model, although the MAC, physical, and network layers have attracted the majority of study focus [7].

In WMNs, IDSs play a vital role in ensuring network security. To automate the IDS process, ML models are employed in conjunction with FS techniques. Three ML models are utilized for automatic IDS in WMNs: SVM, RF, and KNN. These ML models are chosen for their effectiveness in handling complex datasets and their ability to classify network intrusion accurately.

Literature Survey

This section provides a review of the most recently published works on IDS in WMNs. The article [8] proposes an IDS for BLE mesh networks that is based on ML. To detect known threats in the BLE mesh network, such as black hole and grey hole attacks, the IDS employs a single watchdog node. Data from the network layer is regularly updated by the watchdog configuration into the Message Queuing Telemetry Transport (MQTT) broker. The database also keeps the BLE mesh packets that have been timestamp-encoded. To test IDSs, the target attack node can switch between attacks with the touch of a button. The technology uses machine-learning techniques to detect network breaches. In this project, they alter the watchdog's position and test its ability to detect threats. The ML model categorizes assaults as legitimate, grey-hole, or black-hole. It all boils down to a web app that shows the detection status of the BLE mesh network. Stacked ensemble learning, as explained in the article [9], integrates numerous ML approaches to increase attack detection, as opposed to classical learning, which focuses on a single algorithm. To assess the stacked ensemble system's performance, researchers utilized NSL-KDD as a benchmark and compared it to other well-known ML techniques. Compared to other methodologies currently available, the experimental results show that stacked ensemble learning is the best strategy for attack categorization. The recommended solution outperforms rival intrusion detection approaches.

The paper [10] proposes an enhanced empirical-based component analysis for FS. This proposed FS strategy combines the advantages of principal component analysis with empirical mode decomposition to conserve the majority of significant traits. The author utilized LSTM (Long Short Term Memory) to categorize the attack node based on the selected criteria. When compared to state-of-the-art techniques, the suggested framework validated the datasets. In terms of performance measurements, a comparison with existing approaches revealed that the suggested solution was effective. An intrusion detection system (IDS) for mobile ad hoc networks (MANETs) that makes use of stacked autoencoders is the subject of research [11]. A Stacked autoencoder-based MANET approach (Stacked AE-IDS) is proposed in the study to model essential traits with high-level representation and decrease correlation. This technique applies a correlation reduction to the autoencoder's output, making it identical. The proposed DL-based intrusion detection system use labeled datasets for intrusion detection and focuses on Denial of Service (DoS) attacks, particularly those that target mobile network routing services. By implementing the proposed Stacked AE-IDS method, MANET security can be improved. This method enhances the effectiveness of IDS. Researchers [12] employ ML algorithms for intrusion detection. A further use of FS is in selecting the most effective and efficient features. An approach to FS that combines the RF model with the Pearson correlation coefficient is suggested in the study. Train and test the ML models using the TON_IoT dataset. This dataset contains whole new types of attacks and attributes. Train and evaluate multilayer perceptrons (MLPs) and LSTMs. The three primary criteria for evaluation are recall, precision, and accuracy. The results show that employing Decision Trees (DT) for ML and MLP for DL yields

optimal accuracy with low false-positive and false-negative rates. The results also suggest that ML algorithms are effective at detecting network breaches.

The fundamental goal of the study [13] is to evaluate the endurance of ML-based IDSs. The testing dataset accounts for the development of both attack types and network architecture, allowing the proposed approach to more properly assess the long-term effectiveness of an ML-based IDS. We've included six of the most popular ML models for IDSs. The studies using the CIC-IDS datasets showed that ANN and SVM are the most resistant to overfitting. Furthermore, while both DT and RF perform well on the training dataset, the experimental findings show that they are the most prone to overfitting. However, when the difference between the two datasets is small, all of the models may perform well, according to the LUFlow dataset research. Research [14] provides an investigation of ML-based IDS in the IoT that incorporates a variety of feature extraction approaches and many ML models. Several feature extractors, including image filters and TL models like DenseNet and VGG-16, were examined in this work. Examined different ML methods while considering all of the identified feature extraction methodologies. Using the IEEE Dataport dataset, the study evaluated all integrated models comprehensively. The findings showed that the most accurate approach was the combination of VGG-16 with stacking.

Methodology

The primary aim of this study is to develop an IDS for WMNs. To accomplish this objective, publicly available intrusion data specific to WMNs is utilized. However, as this data may contain imperfections, various data pre-processing techniques are applied to enhance its quality. Following data pre-processing, the most pertinent features are selected using three distinct methods: MI, CFS, and PSO. These FS techniques help identify the key features that contribute significantly to the detection of intrusions in WMNs. Subsequently, the selected features from each technique are inputted into three different ML models: SVM, KNN, RF and AE. The performance of each combination of FS technique and ML model is then evaluated using various performance measures. Through this evaluation process, the optimal combination of FS and ML model for IDS in WMNs is determined. The research flow of the ML-based IDS methodology is visually represented in Figure 1.

The CICIDS2017 dataset [15] was created by the Canadian Institute of Cyber Security as an element of the intrusion traffic categorization method. There are seven different types of attacks employed in this dataset. The test infrastructure has been split into two separate networks: the attack and the victim. In both networks, we discovered the usual suspects: routers, firewalls, and switches. This dataset was created with attack variety, anonymity, available protocols, comprehensive network traffic and interaction capture, and entire network configuration definition. The CICFlowMeter application was used to tag the 2,830,540 flows in the CICIDS2017 dataset, each of which had nearly 80 attributes. As a result, the dataset is high-dimensional, multi-class, and unevenly distributed among classes. There are 11,522,402 packets in total, representing network activity in packet format.

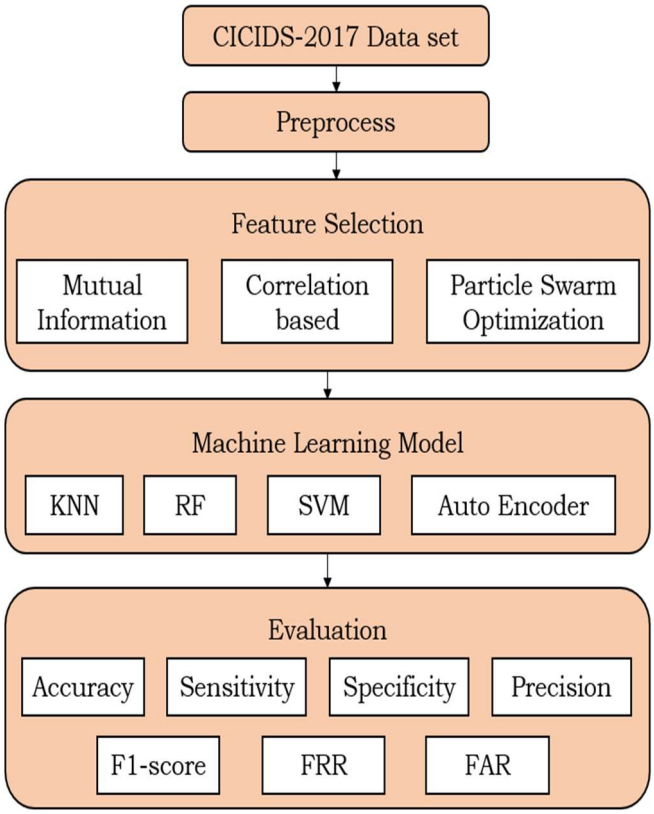

Fig. 1. Research flow of IDS using ML in WMN.

It is common practice to process raw data before preparing it for use in an ML model [16]. This is the first and most critical step in creating an ML model. Data cleansing entailed filtering out undesired noise. At this phase, the study addressed missing values by either removing them or replacing them with an average value. All outliers and superfluous features were removed from the data, making it usable. At this point, we standardized the data by scaling the features to a common range, where necessary. Normalization is defined as the scaling of attribute values in a dataset [17]. It is a data preparation approach. Normalization occurs when data dimensions are changed to match those of a standard distribution. In the last phase, the dataset was divided into two parts: training and testing. To determine the standards, use Equation (1):

$$z = \frac{x_i - \mu}{\sigma} \quad [1]$$

Let z represent the normalized value of x, x_i represent the individual data point, μ represent the mean of the dataset, and σ represents the standard deviation of the CICIDS2017 dataset.

Feature Selection

In this section, we focus on the selection of important features from the dataset using three

distinct FS methods: MI, PSO, and CFS.

MI

MI is a prominent FS method that accurately identifies important features irrespective of data distribution [18]. Because of this quality, it is perfect for use in early detection instances when there aren't enough attack patterns in the data. MI is a statistic that quantifies the amount of information that two discrete variables have about one another. Equation (2) provides the computation for MI.

$$MI(X;Y) = H(X) - H(X|Y) = \sum_{y\in Y}\sum_{x\in X} p(x,y)\ log\frac{p(x,y)}{p(x)\ p(y)} \quad [2]$$

where $p(x)$ and $p(y)$ are the marginal distributions of x and y, respectively, and $p(x,y)$ denotes the joint distribution of x and y, and $H(X|Y)$ is the conditional entropy of X given Y. Equation (3) is used to calculate entropy $H(X)$.

$$H(X) = \sum_{x_i\in X} p(x_i)log\big(p(x_i)\big) \quad [3]$$

To determine the conditional entropy $H(X|Y)$, we use Equation (4).

$$H(X|Y) = -\sum_{y_j\in Y} p(y_j)\sum_{x_i\in X} p(x_i|y_j)\ log\big(p(x_i|y_i)\big) \quad [4]$$

Equation (5) represents the generic expression for the linear combination of Shannon information elements.

$$J(X_k) = I(X_k;Y) - \beta\sum_{X_j\epsilon S} I(X_j;X_k) + \gamma\sum_{X_j\epsilon S} I(X_j;X_k|Y) \quad [5]$$

This Expression consists of equations (6) and (7), representing relevancy and redundancy. Equation (6) represents the relevancy term, whereas Equations (8) and (9) combine marginal and conditional redundancy to form the redundancy term. The weights of these two factors, represented by parameters β and γ, range from 0 to 1.

$$I(X_k;Y) \quad [6]$$

$$\beta\sum_{X_j\epsilon S} I(X_j;X_k) + \gamma\sum_{X_j\epsilon S} I(X_j;X_k|Y) \quad [7]$$

$$\beta\sum_{X_j\epsilon S} I(X_j;X_k) \quad [8]$$

$$\gamma\sum_{X_j\epsilon S} I(X_j;X_k|Y) \quad [9]$$

$I(X_k;Y)$ denotes the candidate feature X_k's MI with the label Y, while $I(X_j;X_k|Y)$ denotes the conditional MI with other features X_j in the set S given the label Y.

CFS

Classical filter algorithms, such as CFS [19], select features based on the results of correlation-based heuristic assessment functions. If a subset's features are closely related to the class but not to one another, this function will favor that subset. Features with poor class linkages should be ignored,

but features with strong relationships to other features should be prioritized when selecting recurring features. The extent to which a feature anticipates classes in parts of the instance space not previously anticipated by other features is the criterion for recognizing the feature. The CFS function for evaluating feature subsets is as follows:

$$M_s = \frac{k\overline{r_{cf}}}{\sqrt{k+k(k-1)+\overline{r_{ff}}}} \quad [10]$$

Eq. (1) denotes the heuristic evaluation for a IDS feature subset s with k features as M_s. Here, $\overline{r_{cf}}$ is the average degree of correlation between features and the category label, and $\overline{r_{ff}}$ is the average degree of inter-correlation among IDS features. A technique that uses feature-subset-based correlation can be used to evaluate CFS. An approach's evaluation value will rise if it identifies smaller $\overline{r_{ff}}$ or larger $\overline{r_{cf}}$subgroups in the acquired data. The training and testing sets are reduced using the features that have the highest evaluation value.

PSO

Kennedy and Eberhart [20] developed the PSO algorithm based on evolutionary computation swarms. This meta-heuristic method constructs a less complex model by taking cues from avian actions and habits. By first examining what is known as the "local best" response on an individual particle level, PSO eventually determines the "global best" solution by combining the findings of all procedures [21]. Each practice has two features: position and velocity. The former shows the direction of travel, while the latter denotes the rate of progress along that route. Throughout the search phase, the algorithm is constantly updating the velocity and position features; iteration ceases when the algorithm reaches the termination condition. The PSO method uses the following five steps to achieve optimal searching. To begin, specify the beginning population and velocity for each particle.

The initial velocity is determined in the first step, and each particle seeks an optimal solution. Every particle in each loop is assigned a solution that matches the local best, which affects how fast particles move. When it comes to particles, the best strategy is one that functions on a global level. In this stage, the velocity is updated as indicated in Equation (11):

$$v_j^{i+1} = \omega v_j^{(i)} + \left(c_1 * r_1\left(localbest_j - x_j^i\right)\right) + \left(c_2 * r_2\left(globalbest_j - x_j^i\right)\right), v_{min} \le v_j^i \le v_{max} \quad [11]$$

where $v(i)$ is the velocity of the ith iteration, $x(i)$ is the position of each particle, w is the inertial weight coefficient, r_1 and r_2 are interval numbers, and I is the number of repetitions. Following the measurement and updating of the particles' velocity, each particle searches in the search space with a new velocity. The fitness function is then used to compute and update the fitness level. The fitness value is utilized to refine the local and optimal solution. Here is a description of the local best update:

$$X_j^{i+1} = X_j^{(i)} + v_j^{(i+1)}; j = 1,2,\ldots,n \quad [12]$$

After ensuring that the termination criteria have been met, the search procedure proceeds; otherwise, it returns to Step 2. Finally it write the important IDS features from the CICIDS2017 dataset.

Machine Learning Model

After FS, the most important features are fed into three ML models: kNN, SVM, RF and AE. In this section, we will delve into the workings of each ML model.

KNN

The kNN algorithm, one of the most common ML algorithms, has long been used to solve a variety of real-world problems [22]. In the feature space, the kNN algorithm assumes that if the majority of its k-nearest neighbors belong to the same category, the sample must as well. With kNN, each new instance is allocated to the class with the most occurrences among the k training examples. The k-nearest instance appears instantly in the training data set. The k-value setting, distance measuring method, and decision rules all influence the classification outcome of kNN. Among these, "the minority obeys the majority" is a typical rule for reaching a consensus. The k-value and distance tests are frequently used as optimization objectives.

In kNN, the only parameter is the k value. When using kNN for prediction, the k value you select makes a significant difference. Using a moderate k number will result in large prediction errors and may introduce noise. Underfitting will occur if the k-value is large. The prediction model is oversimplified in this scenario. It is also a significant factor influencing kNN prediction for spatial sample distance measurement. The angle cosine distance, the Mahalanobis distance, and the Euclidean distance are some of the most commonly used distance measurements [23]. For the sake of the calculation, a closer proximity between the two samples implies a strong likeness, while a greater gap between them suggests low similarity. If the D-dimensional feature has two samples, x_i and x_j, we may simply describe $x_i = (x_{i1}, x_{i2}, \ldots., x_{iD})$ and $x_j = (x_{j1}, x_{j2}, \ldots., x_{jD})$. The value of $d(x_i, x_j)$indicates the distance between two representative samples. As shown in Equation (13), kNN frequently uses the Euclidean distance to measure sample similarity.

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{D}(x_{ik} - x_{jk})^2} \quad [13]$$

SVM

Vapnik [24] introduced SVMs for pattern detection and classification. SVMs employ the concept of inductive structural risk minimization from statistical learning theory to construct sets of high-dimensional hyperplanes. The primary objective of SVMs is to identify a hyperplane in an n-dimensional space that effectively separates points belonging to different classes [25]. The functional margin, representing the distance between the hyperplane and the training data point, serves as a measure of the classification's reliability. In maximum margin classifiers, hyperplanes with the largest functional margin to the nearest training data point are considered optimal for data separation. These

nearest points in the training data set are referred to as support vectors.

$$f(x) = (\sum_{i=1}^{n} \alpha_i y_i x_i)^T x + b \quad [14]$$

$$= \sum_{i=1}^{n} \alpha_i y_i (x_i, x) + b \quad [15]$$

Let x be a vector with n dimensions that represents the data to be categorized. x_i is the feature vector of a training data and y_i is equal to 1 or -1. If x_i is a member of either class 1 or class 2, α_i represents the Lagrange multiplier, and $\langle , \rangle$ represents the inner product operation. The membership status of x can be determined by the sign (+ or -) of $f(x)$ computed using Equation (14). While there may be finite-dimensional classification problems in practice, discriminations requiring sample data are frequently non-linear, necessitating more complex separation procedures within this finite space. The data is projected onto a high-dimensional space to aid separation, as these samples cannot be separated linearly. Next, we can recast the decision function as follows:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i \langle \phi(x_i), \phi(x) \rangle + b \quad [16]$$

The change from the initial, lower dimension to the higher one is represented by ϕ. The hyperplanes are displayed as groups of points with a constant dot product when a vector is present in the higher-dimensional new space. Hyperplanes are vectors that can be considered as such. Dot product calculations in three-dimensional spaces might be tricky, but the kernel function is there to help. An accurate definition of a kernel function would be one that:

$$K(x, z) = \langle \phi(x), \phi(z) \rangle \quad [17]$$

On a low-dimensional space, the inner product $\langle \phi(x_i), \phi(x) \rangle$ can be easily computed using $K(x_i, x)$. As a result, the proper specification of kernel parameters and the functional margin determine SVM accuracy. It is possible to accurately separate small data samples using appropriate Support Vectors, but additional data points may necessitate more complicated mappings.

RF

One of the most well-known ensemble learning approaches, RF, is frequently used for classification issues [26]. It is an advance over bagging in that it employs feature randomness to generate a network of autonomous DTs. The RF method averages the results of numerous DTs to arrive at the final decision [27]. The RF algorithm combines predictions from several DTs, making it an excellent example of ensemble learning. Its classification and regression capabilities make it an adaptable tool. Follow these steps to finish the RF learning process.

- To begin, RF divides the initial training data into pieces using bootstrap sampling.
- The RF randomly selects a subset of features for each DT. It is usual practice to utilize the "$max\_features$" hyper-parameter to define how many features should be included in the subset.

- DTs for each subset are created by combining the bootstrapped sample with a randomly selected feature subset.
- Following the construction of each DT, predictions are created by averaging or voting on each tree's predictions, depending on the task at hand. When creating a prediction in a classification task, the most voted class is selected.

An RF ensemble with N DTs can be defined. Every DT, T_i, is constructed from the first training dataset, D, with a bootstrapped sample, D_i. Furthermore, for each tree, a random subset of features (F_i) is selected. One approach to describing the RF prediction function for classification tasks is:

$$y_{pred} = argmax\left(\frac{sum(T_i(X))}{N}\right), i = 1\ to\ N) \quad [18]$$

Tree predictions for input features X are denoted as $T_i(X)$, y_{pred} is the predicted intrusion, and the class with the most votes is returned by the function $argmax\ (*)$. Here is a definition of the RF prediction function for regression tasks:

$$y_{pred} = \frac{sum(T_i(X))}{N}, i = 1\ to\ N \quad [19]$$

The predicted intrusion is denoted as y_{pred}, and the prediction of tree T_i for input features X is represented as $T_i(X)$.

Attention-based Autoencoder

In the proposed attention-based autoencoder, encoder, decoder, and attention module are built using dense layers of a feedforward neural network. The encoder module encodes the input features into a latent space, which the decoder subsequently reconstructs back to the original input dimensions. Equations (20) and (21) express the two components of the autoencoder as follows:

$$\phi : \chi \rightarrow Z \quad [20]$$

$$\psi : Z \rightarrow \ddot{\chi} \quad [21]$$

Where ϕ represents the encoder function, χ represents the extracted features, Z represents the latent representation (LR), which is the compressed feature space learned by the encoder, ψ represents the decoder function, and $\ddot{\chi}$ represents the reconstructed output.

Encoder: This module learns to compress the input data's dimensions to encode the LR of the features, while the decoder reconstructs the output from this encoded LR. The encoder in this design consists of three dense layers, with 64, 32, and 16 neurons, respectively. Equations (22) and (23) represent the first and last layers, respectively.

$$Z_1 = \sigma(W_1 * x_1 + b_1) \quad [22]$$

$$Z_3 = \sigma(W_3 * [\sigma(W_2 * [\sigma(W_1 * x_1 + b_1) + b_2] + b_3) \quad [23]$$

Where, x_1 represents the input data, W_1, W_2, W_3 and b_1, b_2, b_3 represents weight and bias, σ represents activation function, and Z_1, Z_3 represents the outcome of the first and last layer.

Attention: This module employs a probability operation to predict the result from the bottleneck vector representation. Multiplying the predicted result produces a context vector. The decoder then uses this context vector as input and reconstructs the data to the original dimensions encoded by the encoder. Equation (24) computes the attention score using a single neural network (NN) layer e_{ij} , following Luong's multiplicative attention mechanism.

$$e_{ij} = W * Z + b \quad [24]$$

Where, e_{ij} represents the NN outcome, W and b represents the weight and bias, and Z represents the LR. The result value σ_{ij} is derived using the sigmoidal function, as shown in Equation (25). The obtained result values σ_{ij} are then multiplied by the encoder's LR, as calculated in Equation (26).

$$\sigma_{ij} = \frac{1}{1+exp^{-x}} \quad [25]$$

$$C_t = \sum_{j=1}^{n} \sigma_{ij} * Z_j \quad [26]$$

Where, C_t represents the attention vector of the final layer and Z_j represents the encoder's LR.

Decoder: This module reconstructs the LR which consists of 3 layers with 32, 64, and 128 neurons, respectively. The first two layers employ the ReLU, while the final layer uses a sigmoid to determine the probability distribution. The initial and final layer is indicated by the Equation (27), and (28).

$$O_1 = \sigma(W_1 * C_t + b_1) \quad [27]$$

$$O_3 = \sigma(W_3 * [\sigma(W_1 * O_1 + b_1) + b_2] + b_3) \quad [28]$$

Where, C_t represents the context vector, W_1, W_2, W_3 and b_1, b_2, b_3 represents weight and bias, σ represents activation function, and O_3 represents the final decoder reconstructed output.

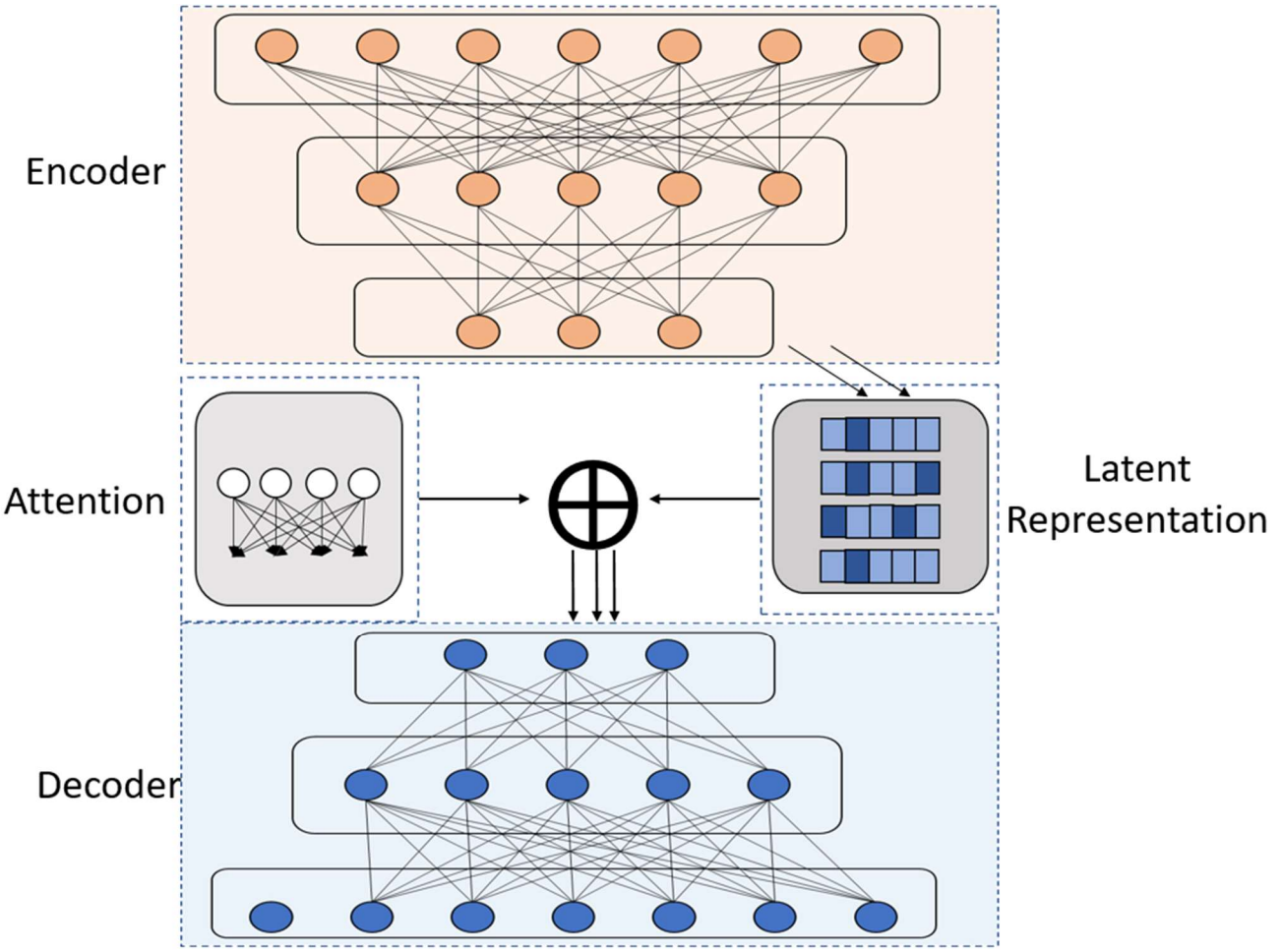

Fig. 2. Attention-based Autoencoder

Finally, the Categorical Cross-Entropy Loss is determined by comparing the actual intrusion x_{val} with the anticipated intrusion$\hat{x}_{val}$.

$$Loss = -\sum_{i=1}^{N}\sum_{i=1}^{C} y_{ij}\log\left(p_{ij}\right) \quad [29]$$

Where N stands for the sample size, C for the number of classes, $p_{ij}$for the anticipated probability that sample i belongs to class j, and y_{ij} for the binary indicator (0 or 1) that class label j is the correct classification for sample i. The AE model uses the RE function for network intrusion detection tasks, which involves identifying suspicious network traffic samples.

Experimental Outcome

Intrusion detection is critical for improving the WMN's security. In this investigation, we used the publicly accessible IDS dataset CICIDS2017. Following the requisite preprocessing, which included managing missing values and feature scaling, the dataset with many features was submitted to FS methods such as MI, CFS, and PSO. The selected features from each approach were then fed

into three machine learning models: SVM, KNN, RF, and AE. The goal was to find the optimal combination of FS and ML models for IDS. Performance was evaluated using a variety of metrics, including accuracy, specificity, sensitivity, precision, F1 score, false rejection rate (FRR), and false acceptance rate (FAR) [28, 29].

The combination of MI and ML models produced encouraging results. The accuracy of KNN, SVM, RF, and AAE was 95.76%, 96.08%, 96.72%, and 97.25%, respectively. SVM has the highest accuracy of the three models. Other parameters, such as specificity, sensitivity, accuracy, and F1 score, were all high, with RF displaying the highest values. Furthermore, AAE had the lowest FRR and FAR scores at 2.24% and 3.07%, respectively.

Table 1. ML model performance using MI features for IDS

Model	KNN	SVM	RF	AAE
Accuracy	95.76	96.08	96.72	97.25
Specificity	95.9	96.36	96.27	97.05
Sensitivity	95.61	95.79	97.19	98.4
Precision	95.69	96.1	96.17	97.56
F1	95.65	95.95	96.67	97.97
FRR	4.39	4.21	2.81	2.24
FAR	4.1	3.64	3.73	3.07

When using CFS features, AAE demonstrated the highest positive metrics and the lowest negative metrics. All positive metrics were greater than 98.5%, while negative metrics were lower than 1.5%, indicating excellent performance.

Table 2. ML model performance using CFS features for IDS

Model	KNN	SVM	RF	AAE
Accuracy	96.38	98.16	97.52	99.2
Specificity	96.18	98.11	97.75	98.7
Sensitivity	96.58	98.21	97.3	98.89
Precision	96.16	98.13	97.76	99.04
F1	96.37	98.17	97.53	98.47

FRR	3.42	1.79	2.7	1.24
FAR	3.82	1.89	2.25	0.86

Evaluation of PSO features with ML models revealed superior performance compared to the other methods. The metrics of PSO+AAE surpassed 99.5% for positive metrics, while negative metrics were very low, less than 0.5%.

Table 3. ML model performance using PSO features for IDS

Model	KNN	SVM	RF	AAE
Accuracy	97.86	97.95	98.9	99.54
Specificity	98.31	98.3	99.04	99.56
Sensitivity	97.42	97.6	98.76	99.52
Precision	98.32	98.29	99.04	99.56
F1	97.87	97.94	98.9	99.54
FRR	2.58	2.4	1.24	0.48
FAR	1.69	1.5	0.96	0.44

Comparing the performance of the three FS methods with the four ML models, the combination of PSO features with AAE showed the best overall performance. The positive metrics comparison of various FS methods with three different ML models is illustrated in Figure 3, while the negative metrics are depicted in Figure 4. These results demonstrate the effectiveness of PSO in combination with AAE for enhancing IDS in WMNs.

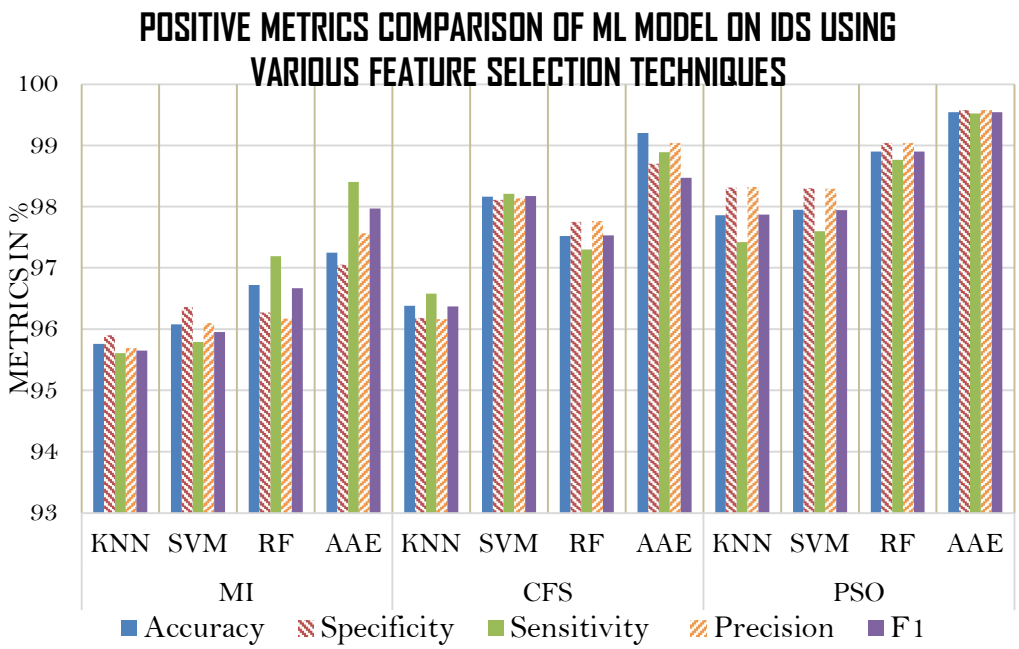

Fig. 3. Positive performance comparison for different combination of feature selection and ML models

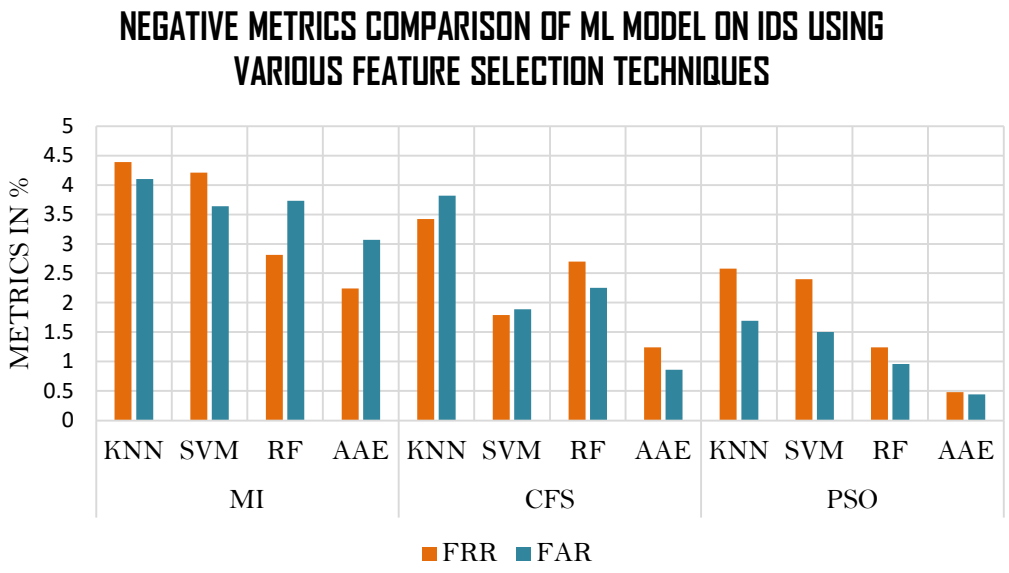

Fig. 4. Negative performance comparison for different combinations of feature selection and ML models

Our evaluation indicates that the proposed PSO+AAE method achieves superior results for IDS in WMN. To assess the performance, we compared our results with four existing works from the years 2022 and 2023, all of which utilized the same CICIDS2017 dataset for IDS evaluation. The detailed comparison is summarized in Table 4. The study referenced as [30] implemented a Decision Tree model, resulting in an accuracy of 94.72%. The work cited as [31] utilized a Convolutional Neural

Network (CNN), significantly improving the accuracy to 98.61%. This marked improvement highlights the potential of deep learning models in enhancing IDS performance. In reference [32], a hybrid model combining CNN with Gated Recurrent Units (GRU) was employed, further boosting the accuracy to 98.73%. Reference [33] also employed a CNN model, achieving a slightly higher accuracy of 98.96%. However, our proposed PSO+AAE method surpasses all the compared methods, achieving an accuracy of 98.99%. This result signifies the potential of our approach in providing more reliable and secure WMN environments.

Table 3. Comparison of our proposed method with existing works

Model	[30]	[31]	[32]	[33]	Proposed Method
Accuracy	94.72	98.61	98.73	98.96	98.9
Specificity	-	-	-	-	99.04
Sensitivity	99.33	96.95	-	99.2	98.76
Precision	98.38	97.05	-	98.7	99.04
F1	86.93	98.09	-	98.94	98.9
FRR	-	-	-	-	1.24
FAR	-	-	-	-	0.96

Conclusion

Security is a significant concern in WMNs. An IDS protects against these threats by monitoring data flow in real-time. In our research, we endeavor to address this critical need for network security by proposing a comprehensive methodology for automatic IDS within WMNs. This methodology is structured around four key steps. First, the data is collected and processed. Second, FS methods (MI, CFS, PSO) are employed to identify important features. Third, the selected features are used as input for ML models (SVM, KNN, RF, and AE) to detect intrusions. Finally, the best combination of FS and ML models is identified using performance measures. The PSO method combined with the three ML models yields better results than the other two methods. Among the three ML models, AAE achieves the highest accuracy of 99.54%. While we use publicly available data in our research, in the future, we aim to compile our dataset utilizing state-of-the-art simulation tools like Network Simulator 3 (NS-3). This will allow us to verify and expand our findings. We aim to enhance the reliability and practicality of our proposed technique by supplementing it with real-world data.

References

[1] Zhu, Chunhui, Myung J. Lee, and Tarek Saadawi. "On the route discovery latency of wireless mesh networks." In *Second IEEE Consumer Communications and Networking Conference, 2005. CCNC. 2005*, pp. 19-23. IEEE, 2005.

[2] Draves, Richard, Jitendra Padhye, and Brian Zill. "Routing in multi-radio, multi-hop wireless mesh networks." In *Proceedings of the 10th annual international conference on Mobile computing and networking*, pp. 114-128. 2004.

[3] Biri, Andreas, Neal Jackson, Lothar Thiele, Pat Pannuto, and Prabal Dutta. "SociTrack: Infrastructure-free interaction tracking through mobile sensor networks." In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pp. 1-14. 2020.

[4] Al-Anzi, Fawaz S. "Design and analysis of intrusion detection systems for wireless mesh networks." *Digital Communications and Networks* 8, no. 6 (2022): 1068-1076.

[5] Rehman, Attique Ur, Muhammad Sajid Mahmood, Shoaib Zafar, Muhammad Ahsan Raza, Fahad Qaswar, Sumayh S. Aljameel, Irfan Ullah Khan, and Nida Aslam. "A survey on MAC-based physical layer security over wireless sensor network." *Electronics* 11, no. 16 (2022): 2529.

[6] Hai, Tran Hoang, Eui-Nam Huh, and Minho Jo. "A lightweight intrusion detection framework for wireless sensor networks." *Wireless Communications and mobile computing* 10, no. 4 (2010): 559-572.

[7] Boubiche, Djallel Eddine, and Azeddine Bilami. "Cross layer intrusion detection system for wireless sensor network." *International Journal of Network Security & Its Applications* 4, no. 2 (2012): 35.

[8] Sivanandam, Nishanth, and T. Ananthan. "Intrusion detection system for bluetooth mesh networks using machine learning." In *2022 International Conference on Industry 4.0 Technology (I4Tech)*, pp. 1-6. IEEE, 2022.

[9] Rajadurai, Hariharan, and Usha Devi Gandhi. "A stacked ensemble learning model for intrusion detection in wireless network." *Neural computing and applications* 34, no. 18 (2022): 15387-15395.

[10] Zhiqiang, Liu, Ghulam Mohiuddin, Zheng Jiangbin, Muhammad Asim, and Wang Sifei. "Intrusion detection in wireless sensor network using enhanced empirical based component analysis." *Future Generation Computer Systems* 135 (2022): 181-193.

[11] Meddeb, Rahma, Farah Jemili, Bayrem Triki, and Ouajdi Korbaa. "A deep learning-based intrusion detection approach for mobile Ad-hoc network." *Soft Computing* 27, no. 14 (2023): 9425-9439.

[12] Hidayat, Imran, Muhammad Zulfiqar Ali, and Arshad Arshad. "Machine learning-based intrusion detection system: an experimental comparison." *Journal of Computational and Cognitive Engineering* 2, no. 2 (2023): 88-97.

[13] Chua, Tuan-Hong, and Iftekhar Salam. "Evaluation of machine learning algorithms in network-based intrusion detection system." *arXiv preprint arXiv:2203.05232* (2022).

[14] Musleh, Dhiaa, Meera Alotaibi, Fahd Alhaidari, Atta Rahman, and Rami M. Mohammad. "Intrusion detection system using feature extraction with machine learning algorithms in IoT." *Journal of Sensor and Actuator Networks* 12, no. 2 (2023): 29.

[15] Sharafaldin, Iman, Arash Habibi Lashkari, and Ali A. Ghorbani. "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSp* 1 (2018): 108-116.

[16] Brownlee, Jason. *Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python*. Machine Learning Mastery, 2020.

[17] Singh, Dalwinder, and Birmohan Singh. "Investigating the impact of data normalization on classification performance." *Applied Soft Computing* 97 (2020): 105524.

[18] Torkkola, Kari. "Feature extraction by non-parametric mutual information maximization." *Journal of machine learning research* 3, no. Mar (2003): 1415-1438.

[19] Gopika, N., and A. Meena Kowshalaya ME. "Correlation based feature selection algorithm for machine learning." In *2018 3rd international conference on communication and electronics systems (ICCES)*, pp. 692-695. IEEE, 2018.

[20] Eberhart, Russell, and James Kennedy. "A new optimizer using particle swarm theory." In *MHS'95. Proceedings of the sixth international symposium on micro machine and human science*, pp. 39-43. Ieee, 1995.

[21] Houssein, Essam H., Ahmed G. Gad, Kashif Hussain, and Ponnuthurai Nagaratnam Suganthan. "Major advances in particle swarm optimization: theory, analysis, and application." *Swarm and Evolutionary Computation* 63 (2021): 100868.

[22] Zhang, Shichao. "Challenges in KNN classification." *IEEE Transactions on Knowledge and Data Engineering* 34, no. 10 (2021): 4663-4675.

[23] Mukahar, Nordiana. "Performance comparison of k nearest neighbor classifier with different distance functions." In *AIP Conference Proceedings*, vol. 2895, no. 1. AIP Publishing, 2024.

[24] Vapnik, Vladimir. *The nature of statistical learning theory*. Springer science & business media, 2013.

[25] Brown, Matilda JM, Barbara R. Holland, and Greg J. Jordan. "hyperoverlap: Detecting biological overlap in n-dimensional space." *Methods in Ecology and Evolution* 11, no. 4 (2020): 513-523.

[26] Parmar, Aakash, Rakesh Katariya, and Vatsal Patel. "A review on random forest: An ensemble classifier." In *International conference on intelligent data communication technologies and internet of things (ICICI) 2018*, pp. 758-763. Springer International Publishing, 2019.

[27] Probst, Philipp, and Anne-Laure Boulesteix. "To tune or not to tune the number of trees in random forest." *Journal of Machine Learning Research* 18, no. 181 (2018): 1-18.

[28] Tharwat, Alaa. "Classification assessment methods." *Applied computing and informatics* 17, no. 1 (2020): 168-192.

[29] Vujović, Ž. "Classification model evaluation metrics." *International Journal of Advanced Computer Science and Applications* 12, no. 6 (2021): 599-606.

[30] Jaradat, Ameera S., Malek M. Barhoush, and Rawan Bani Easa. "Network intrusion detection system: Machine learning approach." *Indones. J. Electr. Eng. Comput. Sci* 25, no. 2 (2022): 1151.

[31] Jose, Jinsi, and Deepa V. Jose. "Deep learning algorithms for intrusion detection systems in internet of things using CIC-IDS 2017 dataset." *International Journal of Electrical and Computer Engineering (IJECE)* 13, no. 1 (2023): 1134-1141.

[32] Henry, Azriel, Sunil Gautam, Samrat Khanna, Khaled Rabie, Thokozani Shongwe, Pronaya Bhattacharya, Bhisham Sharma, and Subrata Chowdhury. "Composition of hybrid deep learning model and feature optimization for intrusion detection system." *Sensors* 23, no. 2 (2023): 890.

[33] Qazi, Emad Ul Haq, Abdulrazaq Almorjan, and Tanveer Zia. "A one-dimensional convolutional neural network (1D-CNN) based deep learning system for network intrusion detection." *Applied Sciences* 12, no. 16 (2022): 7986.