Journal of Education for Pure Science
Vol.16, No. 1(2026)
DOI: https://doi.org/10.32792/jeps.v16i1.787
Website: jceps.utq.edu.iq
Email: jceps@eps.utq.edu.iq

Immune-Inspired Intrusion Detection: An Ensemble Learning Approach using A multiple binary classifications with NSA Validation for WSN Attack Detection

Anwer Hassan Makttof Neamah'

[1]Dept. name of Affiliation, name of organization, City, Post, Country

* Email : anwarhassanmaktof@gmail.com

Received 8 / 9 /2025, Accepted 28/ 10 /2025, Published 1/ 3 /2026

Abstract:

Wireless Sensor Networks (WSNs) have become a vital part of modern applications, but their resource-constrained nature makes them vulnerable to various security threats, such as blackhole, grayhole, and flooding attacks. Conventional security measures are often impractical for WSNs due to their high computational demands, leading to a growing need for lightweight, efficient intrusion detection systems. Many machine learning traditional models struggle with the imbalanced datasets and complex attack patterns found in WSN environments. This research addresses these challenges by introducing a new ensemble classification model that uses decision tree detectors combined with the Negative Selection Algorithm (NSA) to convert the multi-class classification problem into multiple binary classifications. This unique approach enhances accuracy while ensuring the detectors are tolerant of "self-class" behavior, thereby reducing false alarms. The proposed model achieved an accuracy of 98.4%, demonstrating that it significantly outperforms traditional algorithms and indicating its potential as a practical, robust intrusion detection solution for wireless sensor networks.

Keywords: Wireless Sensor Networks, Intrusion Detection System, Attacks on WSNs, Ensemble Classification Model, Negative Selection Algorithm, Decision Tree.

1-Introduction Section

Wireless Sensor Networks (WSNs) have become an essential component of many modern applications, such as environmental monitoring and healthcare systems, where these networks rely on large numbers of small, resource-constrained sensor nodes that cooperate to collect and transmit data [1] [2]. WSNs are exposed to a wide range of security threats such as blackhole, grayhole, flooding, and TDMA manipulation, which can disrupt communication, drain limited energy resources, and compromise information integrity [3]. Traditional security mechanisms, such as encryption protocols, are unsuitable for this type of network due to their high computational requirements and the significant energy they consume during implementation [4].

In response to these challenges, machine learning (ML) techniques have emerged as promising solutions thanks to their ability to learn complex patterns and distinguish between normal and malicious behavior but the ML models often face difficulties when dealing with highly imbalanced datasets, overlapping class distributions, and the dynamic behavior of attackers [5] [6]. These limitations highlight the need for more adaptive and robust detection strategies.

This research proposes an ensemble classification model that combines the Negative Selection Algorithm (NSA) [7] and a decision tree [8] to convert a multi-class classification problem into a multi-class binary classification problem. The model was tested on the WSN-DS dataset, which contains a variety of real-world attack types. The results demonstrate that the proposed method provides significant improvements over traditional algorithms, confirming its potential as a practical intrusion detection solution for wireless sensor networks.

2- Related Work Section

The application of artificial intelligence has revolutionized intrusion detection in WSN, leading to the development of more accurate and scalable detection frameworks. This section provides a systematic classification of recent anomaly-detection studies in wireless sensor networks, highlighting their key advantages and current limitations.

A self-supervised approach was developed in [9]. The authors integrate temporal and spatial features. They extract temporal features using dense networks and use GNNs to integrate topological information, thereby capturing node interactions at both local and global levels. Their model achieved 90.6% accuracy. This study is limited by scalability issues arising from the computational requirements of processing multimedia data, which hinder real-time usability.

The study in [10] applied deep learning in WSN to balance energy-efficient computation with real-time inference. Their work underscores difficulties, including memory and energy limitations and delays in communication.

The study [11] examined various strategies for anomaly detection in IoT-based WSNs. The study concludes that AI-based IDSs improve intrusion detection, but they face significant scalability and computational overhead challenges.

The study [12] developed AHGFFA, an integrated approach that combines unsupervised detection with secure routing for MANET-IoT sensor networks. The study demonstrates effective defence against Blackhole and Grayhole attacks, yet the use of predetermined trust thresholds constrains its responsiveness to new attack behaviors.

According to the research in [13], the combination of a convolutional Autoencoder and Bi-Directional GRUs provides a robust framework for detecting industrial anomalies. The approach has limited applicability to different types of sensors.

1. Proposed Methodology:

3.1 Problem Setting:

In this research, the WSN-DS data collection was relied upon. The data collection includes 374661 data samples and 19 features. The dataset can be described as shown in the formula (1).

$$D = (x_i, y_i) \qquad (1)$$

Where:

$i \in [1, \dots, N]$ represents the samples count.

x_i: is a vector representing the features extracted from WSN-DS without target column (Attack_type).

$y_i \in C = \{1, \dots, k\}$: Label class (target column: Attack_type).

k: Classes count =5 types of attacks.

The research goal is to learn a multiclass classifier $f: x_i \rightarrow C$ that maximizes classification performance on unseen samples.

3.2 Data preparation:

At first, categorical encoding -using LabelEncoder() [14] function L - is applied on \`Attck_type\` column. This process can be described as a function $L: String \rightarrow \{1, \dots, k\}$, so if y_i^{raw} is the original string label, then:

$$y_i = L(y_i^{raw}) \qquad (2)$$

After that, the numerical features which contain missing-values are imputed with the mean computed from the column. For a feature j:

$$\bar{x}_j = \frac{1}{Size\ (\mathcal{I}_j)} \sum_{i \in \mathcal{I}_j} x_{ij} \qquad (3)$$

Where:

$\mathcal{I}_j = \{i | x_{ij} \neq NaN\}$.

So, The imputer transforms:

$$x_{ij} = \begin{cases} x_{ij} & if\ x_{ij} \neq NaN \\ \bar{x}_j & if\ x_{ij} = NaN \end{cases} \qquad (4)$$

Then, duplicate records are removed from the dataset. Duplicate records are data samples containing the same values for each feature. Having duplicate records will consume more memory space and slow down the training process, making it pointless [15]. Retaining a single duplicate record will deliver information to the model during training. This step also prevents leakage from training data to test data when the dataset is split [16].

3.3 Class Imbalance Handling:

To mitigate class imbalance, we use Synthetic Minority Over-sampling Technique (SMOT) with K-Nearest Neighbors (KNN) [17]. For each minority sample x and a randomly chosen neighbor x^{nn} of the same class, synthetic samples are generated by linear interpolation as shown in formula (5):

$$\tilde{x} = x + \lambda\,(x^{nn} - x) \qquad (5)$$

Where: $\lambda \sim \mathcal{U}(0,1)$.

Figure 1 shows an example of how SMOT works.

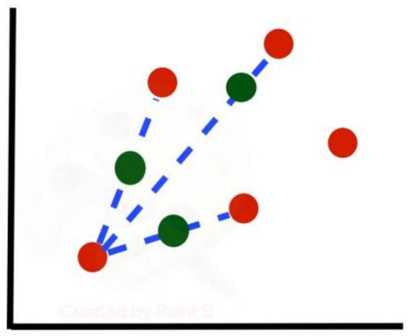


Figure 1. An example of how SMOT works

Figure 2 shows the samples count of each attack before SMOT.

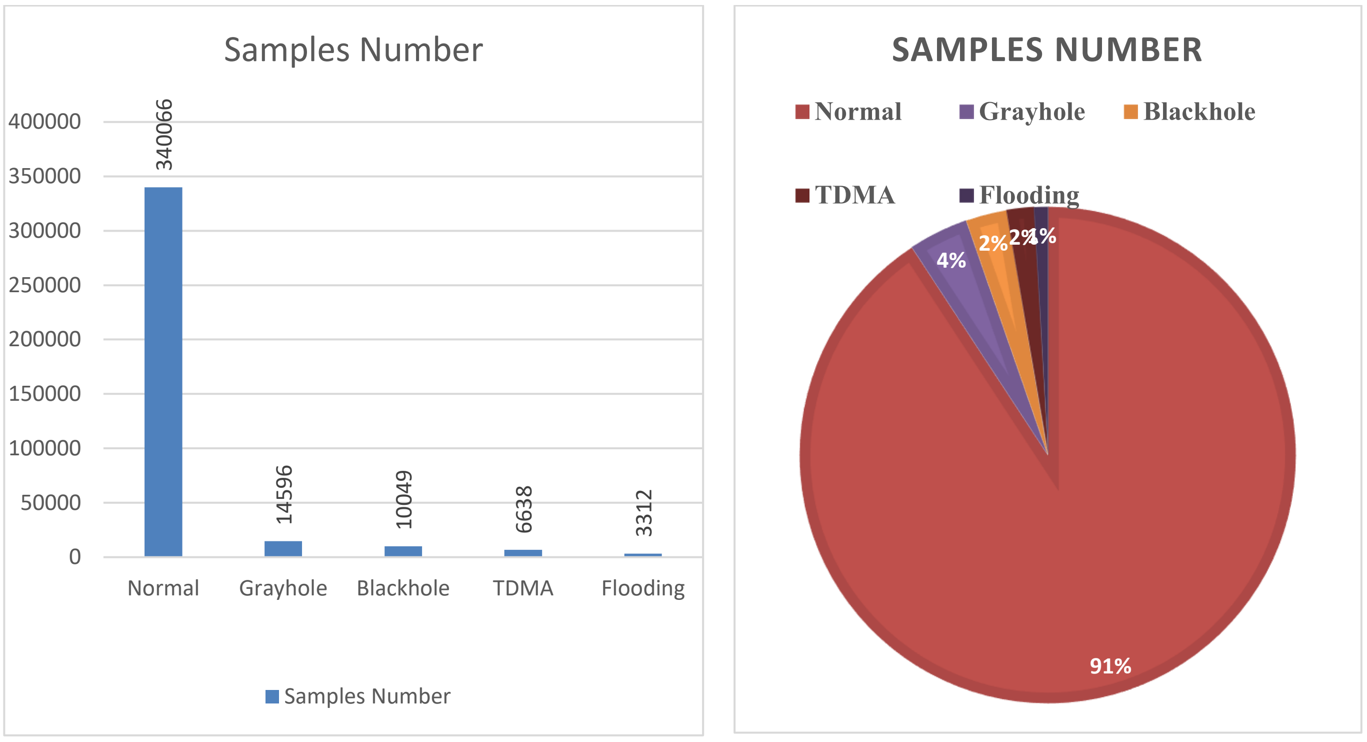


Figure 2. The samples count of each attack before SMOT

Figure 3 shows the samples count of each attack after SMOT

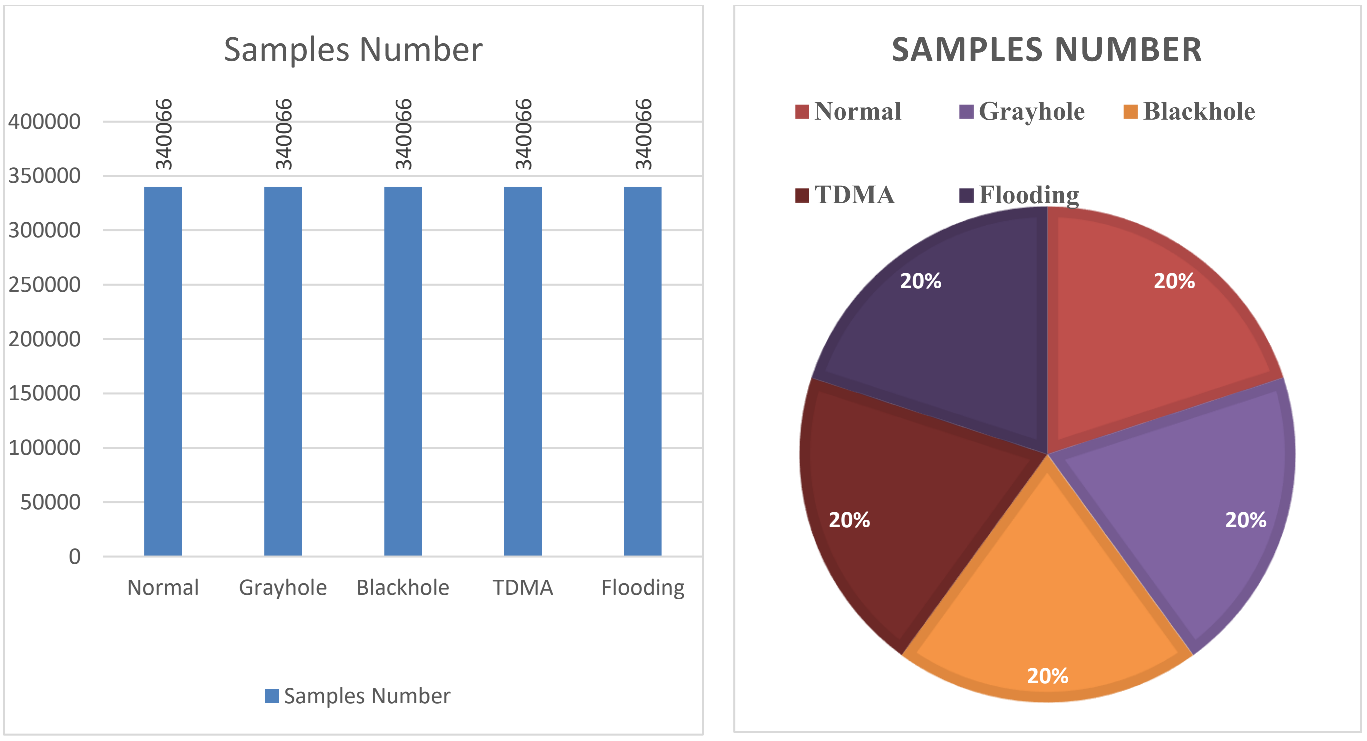


Figure 3. The samples count of each attack after SMOT

3.4 Proposed ensemble method with NSA filtering:

The proposed method in this research is based on ensemble learning where the multi-class classification is converted into a set of binary classifications so that in each process the samples belonging to a specific attack are taken as belonging to one class and the rest of the dataset samples belong to another class, for example the samples belonging to the Normal class are taken and are symbolized by (0) and the rest of the samples that do not belong to the Normal class are symbolized by (1), and this method is repeated for all types of classes. Since the dataset includes 5 classes, we will have 5 binary classifications, each class represents a specific attack as a class and the rest of the attacks as a second class.

For each class $c \in C$, a group of decision-tree detectors are generated and trained to be specialized in recognizing class c against the other classes.
In addition, the random subspace and bootstrap (Bagging) are used to generate diverse candidate trees.
Then, Negative Selection Algorithm (NSA) validation is used to keep only self-tolerant detectors for each class.
The binary relabeling for class c is defined as shown in formula (6).

$$y_i^{(c)} = \begin{cases} c & if\ y_i = c \\ -1 & if\ y_i \neq c \end{cases} \qquad (6)$$

3.4.1 Candidate detectors generation:

For each class c, M candidate trees are generated (we adjusted $M = 2$). After that, a Random subspace of features S are selected.

$$S = \omega d \qquad (7)$$

Where:
$S \subset \{1, \dots, d\}$
d: features count.
$\omega \in [0,1]$: Sub-feature area ratio (we adjusted $\omega = 0.6$).

We will denote $x_i^{(S)}$ to a projection of x_i onto S.

The generated Decision Trees (DTs) h are trained on $\left\{\left(x_i^{(S)}, y_i^{(c)}\right)\right\}_{i \in \text{Bagging samples}}$.

The set of candidate trees for class c is:

$$\mathcal{H}_c^{cand} = \{(h_{cj}, S_{cj})\}_{j=1}^{M} \qquad (8)$$

3.4.2 NSA filtering:

For each candidate tree (h_{cj}, S_{cj}), we compute self-class accuracy on the self-subset (samples with $y_i = c$) as shown in formula (9).

$$Accuracy_{self}(h_{cj}) = \frac{1}{N_c} \sum_{i|y_i=c} h_{cj}\left(x_i^{S_{cj}}\right) = c \qquad (9)$$

Formula (9) represents the accuracy of each decision tree in predicting class c, which represents the number of correct predictions in class c to the number of actual samples belonging to class c within the data set. We simulate the principle of negative selection in the human immune system, where class c represents self-cells (normal body cells) that must be recognized by the immune system detectors with high accuracy to avoid autoimmunity (the immune system cells attacking the body's normal cells). Everything that is contrary to self-cells will be considered abnormal (non-self cells), which mimics germs that enter the body for the first time and are recognized as abnormal and attacked by the body's immune system.

After generating the detectors (decision trees) for each class, the detector is retained if its accuracy in detecting the class is greater than a specific threshold as shown in formula (10).

$$\mathcal{H}_c = \{(h_{cj}, S_{cj}) \in \mathcal{H}_c^{cand} \mid Accuracy_{self}(h_{cj}) \geq \gamma\} \qquad (10)$$

Where:
$\mathcal{H}_c$: The validated detector set for class c.
γ: Threshold (we adjusted γ =0.95).

Detectors failing to recognize `self-cells` reliably are filtered out and this is the NSA principle of self-tolerance.

3.5 Voting for Multiclass Prediction:

To test the performance of the model in general, the test is performed on the test set (the data set is divided into 80% for training and 20% for testing) which includes x_{test} number of samples. Each class-specific detector ensemble $\mathcal{H}_c$ produces votes only when predicting its own class (see formula (11)).

$$s_c = \sum_{(h,S)\in\mathcal{H}_c} h(x_{test} = c) \qquad (11)$$

The final prediction is the class with the maximum votes using argmax as shown in formula (12).

$$\hat{y} = argmax_{c\in C}(s_c(x_{test})) \qquad (12)$$

3.6 Evaluation metrices:

We report the confusion matrix (See figure 4) and the per-class and aggregated metrics used by classification_report [18] [19] [20].

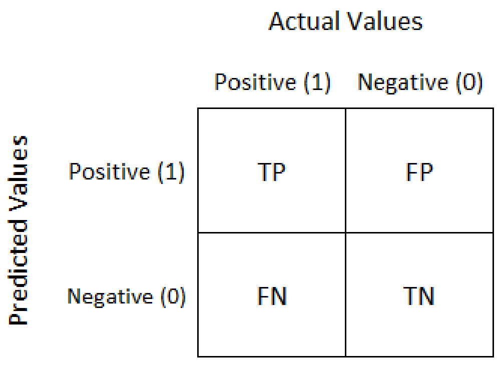


Figure 4. The structure of the CM

Where:

- TP: True Positive.
- TN: True Negative.
- FP: False Positive.
- FN: False Negative.

The following formulas shows the performance metrices based on CM:

$$Precision_c = \frac{TP_c}{TP_c + FP_c} \qquad (13)$$

$$Recall_c = \frac{TP_c}{TP_c + FN_c} \qquad (14)$$

$$F1_c = \frac{2 \times Precision_c \times Recall_c}{Precision_c + Recall_c} \qquad (15)$$

$$Accuracy = \frac{\sum_c TP_c}{\sum_c (TP_c + TN_c + FP_c + FN_c)} \qquad (16)$$

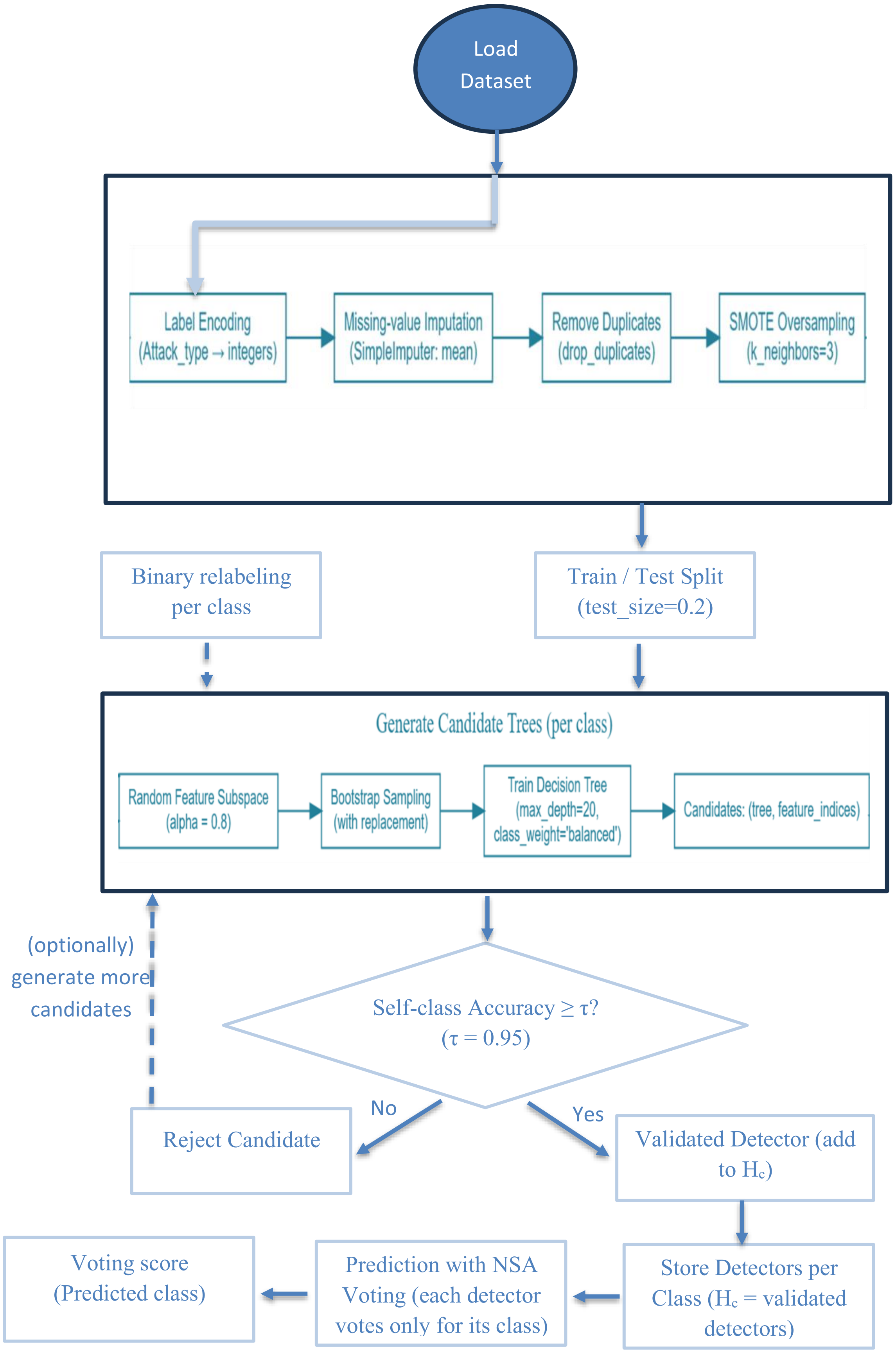
3.7 Algorithm Summary:

Figure 5 shows the methodology pseudocode.

```
Pseudocode
Input: preprocessed $X \in R^{n \times d}$, $y \in \{1, \dots, K\}$, $w = 0.6$, $M = 2$, $\gamma = 0.95$
  1. Clean dataset.
  2. Apply SMOTE on $(X, y)$ with KNN (k=3) to get $(X', y')$.
  3. Split into train and test: $(X_{train}, y_{train}), (X_{test}, y_{test})$.
  4. For each class $c \in C$:
       • Create binary labels $y^{(c)}$.
       • For $j = 1$ to $M$:
           ▪ Sample feature subset $S_{cj}$ where the size of $S_{cj} = wd$.
           ▪ Train tree $h_{cj}$ on $\{(x_i^{S_{cj}}, y_i^{(c)})\}$.
           ▪ Compute $Accuracy_{self}(h_{cj})$ on $\{i: y_i = c\}$.
           ▪ If $Accuracy_{self}(h_{cj}) \geq \gamma$, add $(h_{cj}, S_{cj})$ to $\mathcal{H}_c$.
  5. Prediction: for any $x$ in $X_{test}$, compute $s_c(x)$ and the output $f(x) = argmax_c(s_c(x))$.
  6. Calculate the performance evaluation metrices.
```

Figure 5. The methodology pseudocode

Figure 6 shows the work flow diagram of the proposed methodology.


Load Dataset
Label Encoding (Attack_type → integers)
Missing-value Imputation (SimpleImputer: mean)
Remove Duplicates (drop_duplicates)
SMOTE Oversampling (k_neighbors=3)
Binary relabeling per class
Train / Test Split (test_size=0.2)
Generate Candidate Trees (per class)
Random Feature Subspace (alpha = 0.8)
Bootstrap Sampling (with replacement)
Train Decision Tree (max_depth=20, class_weight='balanced')
Candidates: (tree, feature_indices)
(optionally) generate more candidates
Self-class Accuracy ≥ τ? (τ = 0.95)
No
Yes
Reject Candidate
Validated Detector (add to Hc)
Store Detectors per Class (Hc = validated detectors)
Prediction with NSA Voting (each detector votes only for its class)
Voting score (Predicted class)

2. Results and Discussions:

2.1 Results:

In this section, we demonstrate the performance metrics achieved by applying the proposed model. We used hold-out method to divide the data to 80% of samples for training and 20% for testing. Figure 7 shows the CM results from applying the proposed model.

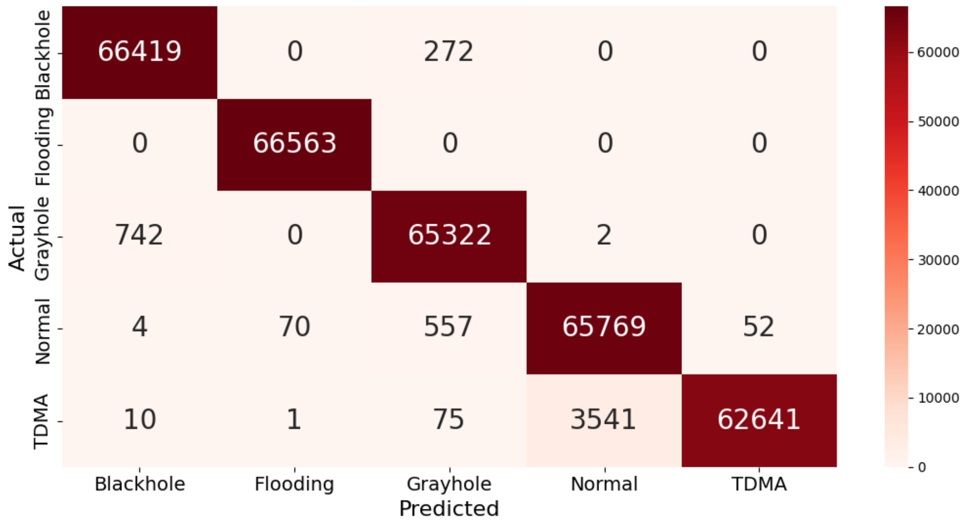


Figure 7. The CM results from applying the proposed model

The results in Figure 7 illustrate the correct and incorrect predictions made by the model. The elements in main-diagonal in Figure 7 indicate the model's correct predictions. We note that the model predicted 66,416 samples as class `Blackhole` (row 1 column 1) when they were actually `Blackhole`. Thus, the model made 326,714 correct predictions out of 332,040 data samples. Elements which not in main-diagonal represent incorrect predictions. We note that the model misclassified 272 samples as `Grayhole` when they were actually `Blackhole`, and misclassified 3541 samples as `Normal` when they were actually `TDMA`. Therefore, the model misclassified 5326 samples out of 332,040. Table 1 shows the proposed model results. Table 2 shows the DT results.

Table 1. The proposed model results

	Precision	Recall	F1 Score	Accuracy
Blackhole_attack	0.989	0.996	0.992	98.4%
Flooding_attack	0.999	1.00	0.999	
Grayhole_attack	0.986	0.989	0.988	
Normal	0.949	0.990	0.969	
TDMA_attack	0.999	0.945	0.971	

Table 2. The DT results

	Precision	Recall	F1 Score	Accuracy
Blackhole_attack	0.75	1.00	0.86	92%
Flooding_attack	1.00	1.00	1.00	
Grayhole_attack	0.97	0.67	0.79	
Normal	0.95	0.97	0.96	
TDMA_attack	1.00	0.94	0.97	

Table 3 shows the Naïve Bayes results.

Table 3. The Naïve Bayes results

	Precision	Recall	F1 Score	Accuracy
Blackhole_attack	0.52	0.98	0.68	69%
Flooding_attack	0.74	0.88	0.80	
Grayhole_attack	0.67	0.52	0.58	
Normal	0.94	0.86	0.90	
TDMA_attack	0.99	0.23	0.37	

Table 4 shows the logistic regression results.

Table 4. The logistic regression results

	Precision	Recall	F1 Score	Accuracy
Blackhole_attack	0.51	1.00	0.68	64%
Flooding_attack	0.61	0.79	0.69	
Grayhole_attack	0.58	0.38	0.46	
Normal	0.94	0.84	0.89	
TDMA_attack	1.00	0.19	0.32	

Table 5 shows the support vector machine results.

Table 5. The support vector machine results

	Precision	Recall	F1 Score	Accuracy
Blackhole_attack	0.69	1.00	0.81	85%
Flooding_attack	0.93	0.98	0.95	
Grayhole_attack	0.83	0.66	0.74	
Normal	0.93	0.88	0.91	
TDMA_attack	0.99	0.76	0.86	

5. Discussions:

The results demonstrate that the proposed NSA-based ensemble model achieved superior performance compared to all baseline algorithms, where the model obtained 98.4% accuracy with consistently high precision, recall, and F1-scores across all attack types. This level of performance indicates that the combination of SMOTE balancing, decision-tree ensembles, and NSA filtering provided strong generalization and robustness when distinguishing between different WSN attacks.

By contrast, the baseline classifiers struggled with overlapping feature distributions. The Decision Tree model (Table 2) reached reasonable performance, with accuracy 92%, but its precision and recall varied considerably between classes. Naïve Bayes (Table 3) and Logistic Regression (Table 4) performed the weakest, particularly for TDMA and Grayhole attacks, where recall dropped sharply because these models tend to rely heavily on distributional assumptions, which may not hold in a dataset with complex and nonlinear decision boundaries. The SVM model (Table 5) showed better resilience, achieving accuracy of 85% and balanced performance across most classes, but it still could not match the proposed approach.

What stands out is that the ensemble's ability to validate detectors using the NSA principle helped filter out weak classifiers that might otherwise contribute to misclassification where this mechanism ensured that only reliable detectors voted for their respective classes, leading to fewer false positives and false negatives. The comparison highlights this point: while baseline methods achieved moderate success in certain classes, the proposed model consistently produced stable predictions across all categories. The results confirm that introducing NSA validation into ensemble learning and transforming the classification problem into a multi-class binary classification improves classification accuracy and enhances attack detection, which is crucial for practical intrusion detection in WSN environments.

6- Conclusion

This study aimed to improve attack detection in WSNs by developing an ensemble method combining decision trees with the negative selection algorithm. The results demonstrated that this approach outperforms traditional ML classifiers. The proposed model relied on detector validation using NSA principles and transformed the problem model from multi-class classification to multi-class binary classification. The model achieved remarkable accuracy, precision, recall, and F1 scores across all classes. These findings confirm that integrating immune-inspired mechanisms into ensemble learning can effectively address the challenges posed by overlapping attack patterns, which often hinder the performance of conventional models.

While the proposed framework has proven effective, there is still room for refinement, such as increasing the number of candidate detectors to further enhance diversity. Future studies could also examine integrating deep learning models, such as convolutional or recurrent architectures, with the NSA framework to capture temporal or spatial dependencies that decision trees alone may miss. Also, extending the methodology to larger or more heterogeneous datasets would help verify its scalability and robustness in realistic environments

References

[1] **H. M. A. Fahmy**, "WSNs applications," in *Concepts, applications, experimentation and analysis of wireless sensor networks*: Springer, 2023, pp. 67-242.

[2] **S. Hudda and K. Haribabu**, "A review on WSN based resource constrained smart IoT systems," *Discover Internet of Things,* vol. 5, no. 1, p. 56, 2025.

[3] **A. Banu, S. Nikitha, K. Vinitha, E. Laxmi, and A. Tarannum**, "Machine Learning Driven Advanced Défense Mechanisms Against Blackhole and Flooding Attacks in Wireless Sensor Networks."

[4] **M. U. Mushtaq, J. Hong, M. Owais, and S. A. Danso**, "Enhancing security and energy efficiency in wireless sensor network routing with IOT challenges: a thorough review," *LC international journal of stem (ISSN: 2708-7123),* vol. 4, no. 3, pp. 1-24, 2023.

[5] **M. Rabbani *et al.***, "A review on machine learning approaches for network malicious behavior detection in emerging technologies," *Entropy,* vol. 23, no. 5, p. 529, 2021.

[6] **M. Altalhan, A. Algarni, and M. T.-H. Alouane**, "Imbalanced data problem in machine learning: A review," *IEEE Access,* 2025.

[7] **K. D. Gupta and D. Dasgupta**, "Negative selection algorithm research and applications in the last decade: A review," *IEEE Transactions on Artificial Intelligence,* vol. 3, no. 2, pp. 110-128, 2021.

[8] **V. G. Costa and C. E. Pedreira**, "Recent advances in decision trees: an updated survey," *Artificial Intelligence Review,* vol. 56, no. 5, pp. 4765-4800, 2023.

[9] **M. Ye, Q. Zhang, X. Xue, Y. Wang, Q. Jiang, and H. Qiu**, "A novel self-supervised learning-based anomalous node detection method based on an autoencoder for wireless sensor networks," *IEEE Systems Journal,* vol. 18, no. 1, pp. 256-267, 2024.

[10] **Y. Qiu, L. Ma, and R. Priyadarshi**, "Deep learning challenges and prospects in wireless sensor network deployment," *Archives of Computational Methods in Engineering,* vol. 31, no. 6, pp. 3231-3254, 2024.

[11] **S. H. Rafique, A. Abdallah, N. S. Musa, and T. Murugan**, "Machine learning and deep learning techniques for internet of things network anomaly detection—current research trends," *Sensors,* vol. 24, no. 6, p. 1968, 2024.

[12] **S. Alangari**, "An unsupervised machine learning algorithm for attack and anomaly detection in IoT sensors," *Wireless Personal Communications,* pp. 1-25, 2024.

[13] **L. S. John, S. Yoon, J. Li, and P. Wang**, "Anomaly detection using convolutional autoencoder with residual gated recurrent unit and weak supervision for photovoltaic thermal heat pump system," *Journal of Building Engineering,* vol. 100, p. 111694, 2025.

[14] **N. Selayanti, S. A. Putri, M. Kristanaya, M. P. Azzahra, M. G. Navsih, and K. M. Hindrayani**, "Penerapan Machine Learning Algoritma Random Forest Untuk Prediksi Penyakit Jantung," in *PROSIDING SEMINAR NASIONAL SAINS DATA*, 2024, vol. 4, no. 1, pp. 895-906.

[15] **A. Isenko, R. Mayer, J. Jedele, and H.-A. Jacobsen**, "Where is my training bottleneck? hidden trade-offs in deep learning preprocessing pipelines," in *Proceedings of the 2022 International Conference on Management of Data*, 2022, pp. 1825-1839.

[16] **A. Apicella, F. Isgrò, and R. Prevete**, "Don't push the button! exploring data leakage risks in machine learning and transfer learning," *Artificial Intelligence Review,* vol. 58, no. 11, pp. 1-58, 2025.

[17] **S. A. Alex, J. J. V. Nayahi, and S. Kaddoura**, "Deep convolutional neural networks with genetic algorithm-based synthetic minority over-sampling technique for improved imbalanced data classification," *Applied Soft Computing,* vol. 156, p. 111491, 2024.

[18] **Z. Guoping**, "Invariance Properties and Evaluation Metrics Derived from the Confusion Matrix in Multiclass Classification," *Mathematics,* vol. 13, no. 16, p. 2609, 2025.

[19] **E. Helmud, F. Fitriyani, and P. Romadiana**, "Classification comparison performance of supervised machine learning random forest and decision tree algorithms using confusion matrix," *Jurnal Sisfokom (Sistem Informasi dan Komputer),* vol. 13, no. 1, pp. 92-97, 2024.

[20] **M. M. Mijwil and M. Aljanabi**, "A comparative analysis of machine learning algorithms for classification of diabetes utilizing confusion matrix analysis," *Baghdad Science Journal,* vol. 21, no. 5, p. 24, 2024.