AL ENHANCE SERCURITY: ROBUST DATA TRANSMISSION IN AD-HOC NETWORK

Dr. Bagath Basha[1], Shashank Kudire[2], Shiva kumar Borem[2], Sharon Rose Indrala[2]

[2]UG Scholar, [1,2]Department of Computer Science and Engineering

[1,2]Kommuri Pratap Reddy Institute of Technology, Ghatkesar, Hyderabad, Telangana.

ABSTRACT

Ad-hoc networks, dynamic and decentralized, are vital for scenarios where traditional infrastructure-based networks are impractical. Ensuring resilient and confidential data transmission in these networks is challenging due to their dynamic nature and security vulnerabilities. Traditional methods, including cryptographic techniques, secure routing protocols, and intrusion detection systems, often struggle to adapt to changing conditions and may not provide adequate resilience and confidentiality. This research focuses on developing a system that employs artificial intelligence (AI) to enhance data transmission resilience and confidentiality in ad-hoc networks. By designing algorithms that dynamically adapt to network conditions and secure data transmission against threats, this study aims to improve security and reliability. Ad-hoc networks are crucial in emergency response, military operations, disaster recovery, and IoT environments, where data reliability and confidentiality are paramount. The Research leverages AI for adaptive routing, threat detection, and encryption. AI algorithms can autonomously and accurately respond to changing network conditions and security threats, significantly enhancing ad-hoc network security. This research promises to revolutionize data transmission reliability and security in critical scenarios, where ad-hoc networks are essential, by integrating advanced AI techniques to ensure resilient and confidential communications.

Keywords: Ad-hoc networks, Data transmission, Artificial intelligence, Resilience, Confidentiality, Dynamic Adaptation.

1. INTRODUCTION

Naval mines represent a formidable threat to maritime activities, spanning naval operations, shipping, and offshore infrastructure. Safeguarding vessels and ensuring secure maritime environments necessitates the accurate detection and classification of these mines. One promising avenue for achieving this is the utilization of sonar technology, which has long served as a pivotal tool in underwater surveillance. The historical backdrop of naval mine warfare traces its roots to ancient times, but it gained prominence during the 19th and 20th centuries, evolving into more sophisticated and elusive forms with technological advancements. Traditional methods for mine detection, such as visual inspection and magnetic detection, encounter limitations, particularly in challenging environments like murky or deep waters. Sonar technology emerged as a viable solution to overcome these challenges.

The contemporary challenge lies in the precise classification of underwater objects identified by sonar systems, with a specific focus on distinguishing between mines and natural formations like rocks. This task is inherently complex due to the variability in the acoustic signatures of different objects and the imperative for real-time decision-making to avert potential threats. Conventional mine detection systems often integrate various sensors, including sonar, for identifying underwater objects. However, interpreting sonar signals traditionally relies on intricate signal processing algorithms and rule-based

systems. These conventional approaches may struggle to adapt to the dynamic and diverse underwater environment, resulting in either false positives or missed detections.

The impetus for sonar signal classification arises from the recognized limitations of traditional systems in grappling with the intricacies of underwater environments. Sonar signals, serving as a rich source of information about underwater object characteristics based on their acoustic reflections, offer valuable insights. Leveraging connectionist networks, notably neural networks, stands as a promising approach to enhance the discrimination between mines and rocks. These networks possess the capacity to learn intricate patterns and relationships within sonar data, thereby improving the accuracy and efficacy of mine classification.

2. LITERATURE SURVEY

Shahabi et al. [23] designed a novel routing algorithm in addition to AODV to secure a network from BHA. Using this strategy, the malicious nodes are identified based on the node's behavior. If any are detected that node is deleted from the route. The experiments also show better Packet Delivery Rate (PDR) with reduced delay. Baadache and Belmehdi [24] presented an acknowledgment-based routing approach by which the communicating nodes send acknowledgment whenever the nodes receive the data packet. The algorithm suffers from high routing overhead as each node sends an acknowledgment message to the prior node. In addition to the above problem, Kumari and Paramasivan [25] developed a routing mechanism of trust where the behavior of nodes is analyzed based on the dropping rate of packets, but this protocol also suffers from high overhead because of the additional use of control packets. Gurung and Chauhan [26] used the approach of mitigating a Gray Hole Attack (GHA) that takes the help of other nearby nodes, known as the nodes of the Intrusion Detection System (IDS), to monitor the performance of other communicating nodes. In the appearance of any malicious node, the packet drop value of the node is higher. In this case, the important message ("ALERT") is transferred among the networks to intimate other nodes to separate attacker nodes. As the algorithm works on the defined threshold, proper positioning of special nodes is required. Mohanapriya and Krishnamurthi [14] designed a new approach source node that imitates the destination node of the total amount of packets transmitted from all expected routes. Query request is transmitted by the destination node, particularly in the case where the node cannot obtain the desired packets. In response to this query reply, a message is sent back to the node that is about two-hop counts in contrast to the destination node. Once the message of query reply is received, the destination node compares its prior-received data with the recently received data. In case an error appears, consider that node as the suspected node and add it to the list of malicious nodes. Keerthika and Malarvizhi [27] presented a combined trust-based bee approach to secure the network against BHA. ABC is used for the detection of a secure route. A new solution is generated based on the fitness function of bees. The designed algorithm shows enhancement in the PDR and end-to-end delay. Merlin and Ravi [28] presented a new trust-based approach that works on energy-aware routing for MANET. The BHA has been detected for single as well as for multiple routes formed during the data communication process. Rezaei et al. [29] presented a mechanism in which the source node transmits the route response data packet after processing the node's information, which is later used for BHA detection. Whether the node is genuine or malicious is decided by the intermediate node. On the other hand, Yasin et al. [30] used a timer and baiting-based method for BHA detection in MANET. Monica Sood et al. [31] used a deep learning model for traffic flow prediction based on attention for inventory automation using a Wireless Sensor Network.

3. PROPOSED SYSTEM

The research demonstrates a comprehensive approach to an ad-hoc networks project, encompassing data loading, preprocessing, model training, and evaluation. Below is a detailed explanation of each step in a human-readable manner:

Dataset Upload: The research begins with the importation of necessary libraries and the loading of the dataset. The dataset is stored in a Pandas DataFrame (df), allowing for easy manipulation and analysis.

Data Exploration and Analysis: Basic exploratory data analysis (EDA) is performed to gain insights into the dataset. Descriptive statistics, including mean, standard deviation, and quartiles, are obtained using the describe() method. The info() method is employed to examine the data types and null values in each column.

Visualization of Decision Counts: The distribution of decision classes is visualized using a count plot with seaborn. This provides a quick overview of the balance or imbalance in the target variable, 'Decision'.

Preprocessing: Null values are checked for and identified throughout the dataset. The independent variables are scaled using the StandardScaler from scikit-learn, ensuring that all features have a similar scale. This is crucial for models that are sensitive to the magnitude of input features.

Train-Test Splitting: The dataset is split into training and testing sets using the train_test_split function. The testing set comprises 20% of the data, and a random seed is set for reproducibility.

Logistic Regression Model: A logistic regression model is instantiated and trained on the training set (X_train and y_train). The model is then tested on the reserved testing set (X_test), and the accuracy, confusion matrix, and classification report are displayed.

Artificial Neural Network (ANN) Model: An ANN model is constructed using the Keras library. The architecture includes an input layer with 64 neurons, a hidden layer with 32 neurons using the ReLU activation function, and an output layer with a sigmoid activation function for binary classification. The model is compiled using binary cross-entropy loss and the Adam optimizer.

Model Training and Evaluation (ANN): The ANN model is trained on the resampled training set (X_train_smote and y_train_smote) using the Synthetic Minority Over-sampling Technique (SMOTE) for handling imbalanced data. The model is then evaluated on the original testing set, and accuracy, classification report, and confusion matrix are displayed.

ROC Curve Analysis: Receiver Operating Characteristic (ROC) curves are generated for both the logistic regression and ANN models. The curves visually represent the trade-off between true positive rate and false positive rate, with the area under the curve (AUC) serving as a performance metric.

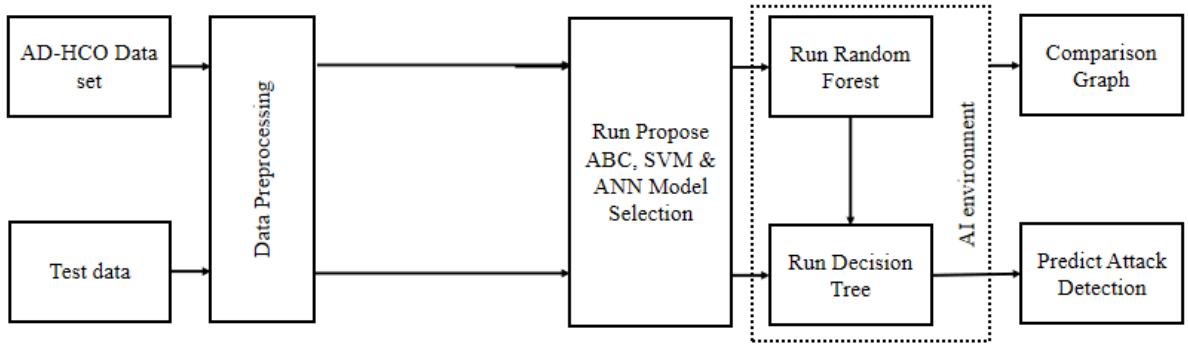


Fig. 1: Block diagram of proposed system.

ANN Classifier

Although today the Perceptron is widely recognized as an algorithm, it was initially intended as an image recognition machine. It gets its name from performing the human-like function of perception, seeing, and recognizing images. Interest has been centered on the idea of a machine which would be capable of conceptualizing inputs impinging directly from the physical environment of light, sound, temperature, etc. — the "phenomenal world" with which we are all familiar — rather than requiring the intervention of a human agent to digest and code the necessary information. Rosenblatt's perceptron machine relied on a basic unit of computation, the neuron. Just like in previous models, each neuron has a cell that receives a series of pairs of inputs and weights. The major difference in Rosenblatt's model is that inputs are combined in a weighted sum and, if the weighted sum exceeds a predefined threshold, the neuron fires and produces an output.

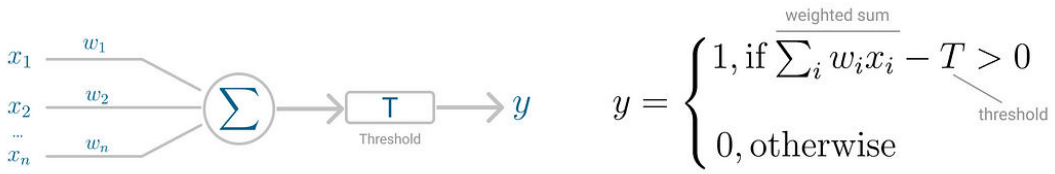


Fig. 2: Perceptron neuron model (left) and threshold logic (right).

Threshold *T* represents the activation function. If the weighted sum of the inputs is greater than zero the neuron outputs the value 1, otherwise the output value is zero.

Perceptron for Binary Classification

With this discrete output, controlled by the activation function, the perceptron can be used as a binary classification model, defining a linear decision boundary.

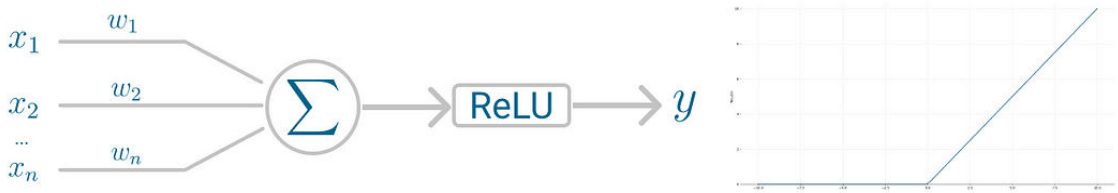
It finds the separating hyperplane that minimizes the distance between misclassified points and the decision boundary. The perceptron loss function is defined as below:

$$\underbrace{D(w,c)}_{\text{distance}} = -\sum_{i \in M} \overbrace{y_i}^{\text{output}} (x_i w_i + c)$$

misclassified observations

To minimize this distance, perceptron uses stochastic gradient descent (SGD) as the optimization function. If the data is linearly separable, it is guaranteed that SGD will converge in a finite number of steps. The last piece that Perceptron needs is the activation function, the function that determines if the neuron will fire or not. Initial Perceptron models used sigmoid function, and just by looking at its shape, it makes a lot of sense! The sigmoid function maps any real input to a value that is either 0 or 1 and encodes a non-linear function. The neuron can receive negative numbers as input, and it will still be able to produce an output that is either 0 or 1.

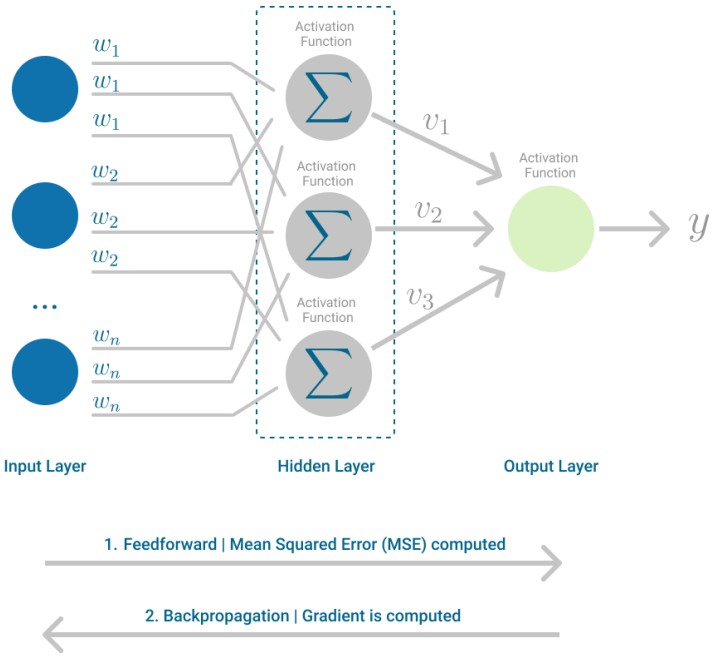
But, if you look at Deep Learning papers and algorithms from the last decade, you'll see the most of them use the Rectified Linear Unit (ReLU) as the neuron's activation function. The reason why ReLU became more adopted is that it allows better optimization using SGD, more efficient computation and is scale-invariant, meaning, its characteristics are not affected by the scale of the input. The neuron receives inputs and picks an initial set of weights random. These are combined in weighted sum and then ReLU, the activation function, determines the value of the output.



Perceptron uses SGD to find, or you might say learn, the set of weight that minimizes the distance between the misclassified points and the decision boundary. Once SGD converges, the dataset is separated into two regions by a linear hyperplane. Although it was said the Perceptron could represent any circuit and logic, the biggest criticism was that it couldn't represent the XOR gate, exclusive OR, where the gate only returns 1 if the inputs are different. This was proved almost a decade later and highlights the fact that Perceptron, with only one neuron, can't be applied to non-linear data.

ANN

The ANN was developed to tackle this limitation. It is a neural network where the mapping between inputs and output is non-linear. A ANN has input and output layers, and one or more hidden layers with many neurons stacked together. And while in the Perceptron the neuron must have an activation function that imposes a threshold, like ReLU or sigmoid, neurons in a ANN can use any arbitrary activation function. ANN falls under the category of feedforward algorithms because inputs are combined with the initial weights in a weighted sum and subjected to the activation function, just like in the Perceptron. But the difference is that each linear combination is propagated to the next layer. Each layer is feeding the next one with the result of their computation, their internal representation of the data. This goes all the way through the hidden layers to the output layer. If the algorithm only computed the weighted sums in each neuron, propagated results to the output layer, and stopped there, it wouldn't be able to learn the weights that minimize the cost function. If the algorithm only computed one iteration, there would be no actual learning. This is where Backpropagation comes into play.



Backpropagation: Backpropagation is the learning mechanism that allows the ANN to iteratively adjust the weights in the network, with the goal of minimizing the cost function. There is one hard requirement for backpropagation to work properly. The function that combines inputs and weights in a neuron, for instance the weighted sum, and the threshold function, for instance ReLU, must be differentiable. These functions must have a bounded derivative because Gradient Descent is typically the optimization function used in ANN. In each iteration, after the weighted sums are forwarded through all layers, the gradient of the Mean Squared Error is computed across all input and output pairs. Then, to propagate it back, the weights of the first hidden layer are updated with the value of the gradient. That's how the weights are propagated back to the starting point of the neural network. One iteration of Gradient Descent is defined as follows:

$$\underbrace{\Delta_w(t)}_{\substack{\text{Gradient}\\\text{Current Iteration}}} = \underbrace{-\varepsilon}_{\text{Bias}} \frac{\overbrace{dE}^{\text{Error}}}{\underbrace{dw_{(t)}}_{\text{Weight vector}}} + \underbrace{\alpha}_{\text{Learning Rate}} \underbrace{\Delta_{w(t-1)}}_{\substack{\text{Gradient}\\\text{Previous Iteration}}}$$

This process keeps going until gradient for each input-output pair has converged, meaning the newly computed gradient hasn't changed more than a specified convergence threshold, compared to the previous iteration.

4. RESULTS AND DISCUSSION

4.1 Implementation description

The research is a Tkinter-based GUI application for ensuring resilient and confidential data transmission with artificial intelligence in ad-hoc networks.

Here's a breakdown of the code:

— Import Statements: Various libraries are imported, including Tkinter for GUI, scikit-learn for machine learning tasks, SwarmPackagePy for the Artificial Bee Colony (ABC) algorithm, seaborn and matplotlib for data visualization, and Keras for deep learning.
— Main Tkinter Window: The main GUI window is created with a title and geometry settings.
— Global Variables: Global variables are declared to store the filename, dataset, feature matrices (X), target variable (Y), and other metrics related to the algorithms.
— Functionality Buttons: Several buttons are created for different functionalities, such as uploading the AODV dataset, preprocessing the dataset, running the proposed ABC, SVM & ANN model, running Random Forest and Decision Tree algorithms, plotting a comparison graph, and detecting attacks from test data.
— These buttons are associated with functions that perform the corresponding tasks.
— Text Widget: A Text widget is used to display information, results, and messages. It includes a scrollbar for navigation.
— Functions: Functions are defined for tasks such as uploading the dataset, preprocessing, running the proposed model, running Random Forest and Decision Tree algorithms, plotting a comparison graph, and predicting attacks from test data.
— The calculateMetrics function is used to calculate metrics and plot confusion matrices.
— Labels and Buttons: Labels, buttons, and entry widgets are created for user interaction.
— Graphs and Plots: The code includes the functionality to plot bar graphs for different algorithms' performance metrics and confusion matrices.

4.2 Results description

This figure 3 depicts the main interface of the application, providing an overview of the tool for ad-hoc network framework. It includes various features and options for users to interact with the application.

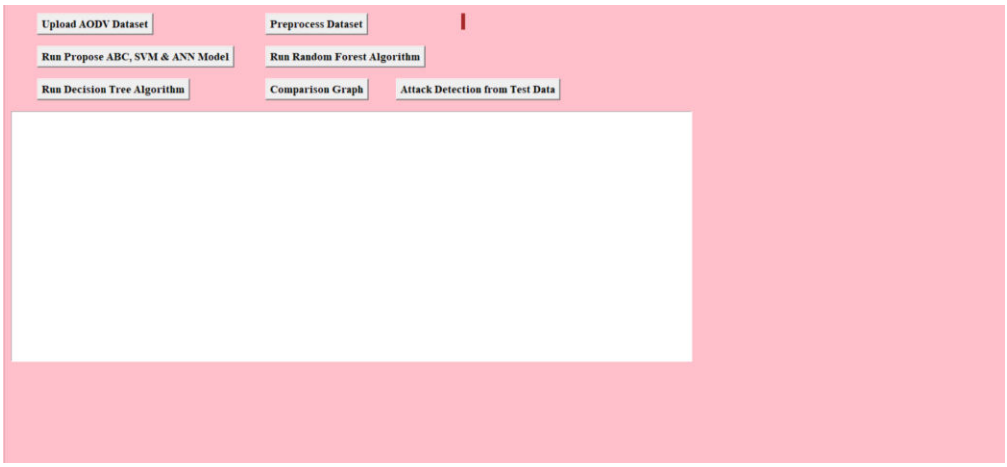


Figure 3: Main GUI application of ensuring resilient and confidential data transmission with artificial intelligence in ad-hoc networks.

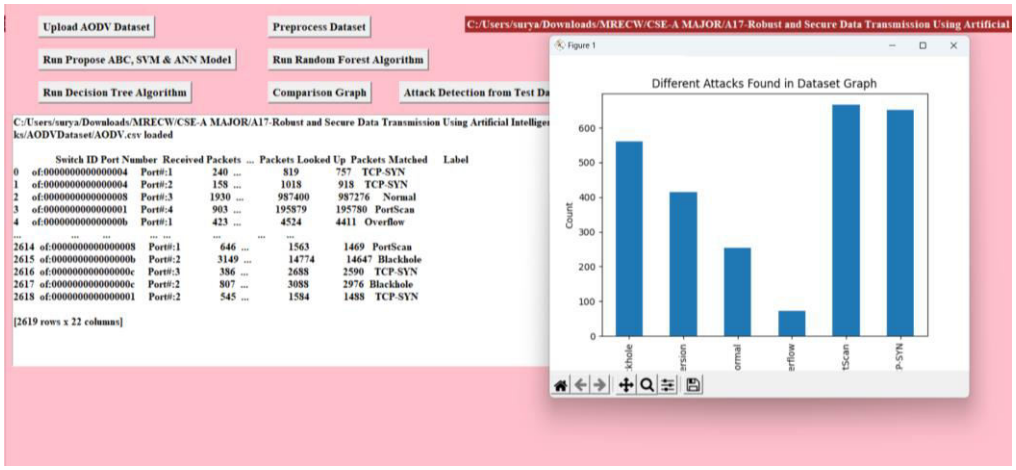


Figure 4: Represents the sample dataset and the count value of the label column.

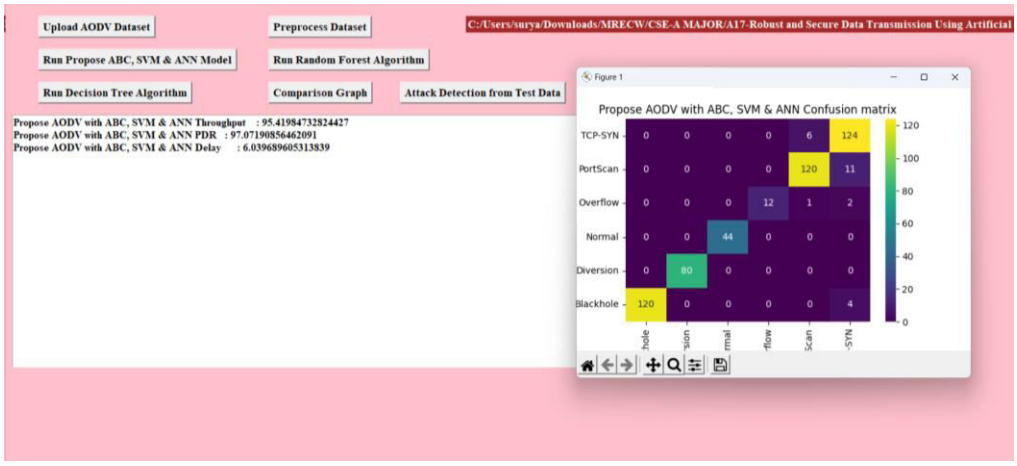


Figure 5: Displays the confusion matrix and Throughput, Pdr, Delay proposed algorithms.

The figure 4 displays a sample of the dataset along with the count values for the label column. It provides an overview of the data and the distribution of different classes in the label column. The figure 5 shows the confusion matrix and performance metrics (Throughput, Packet Delivery Ratio (Pdr), Delay) for a set of proposed algorithms applied to the dataset. The figure 6 presents the confusion matrix and performance metrics, but specifically for the Random Forest algorithms.

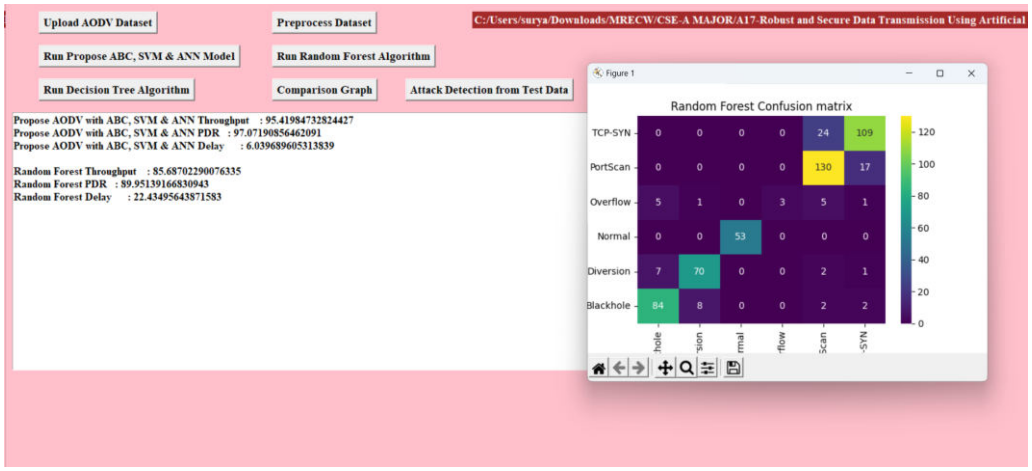


Figure 6: Displays the confusion matrix and Throughput, Pdr, Delay for random forest algorithms.

The figure 7 represents the confusion matrix and performance metrics, focusing on the Decision Tree algorithms.

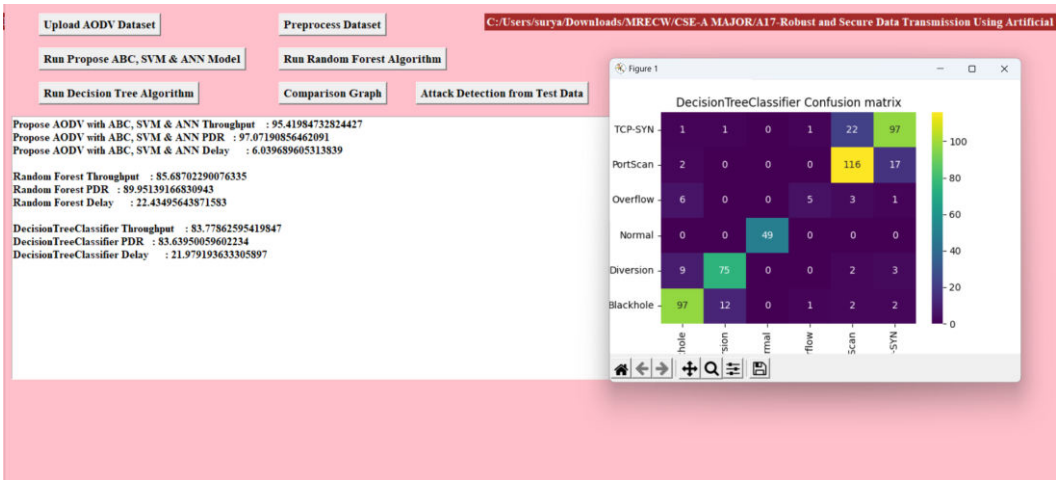


Figure 7: Displays the confusion matrix and Throughput, Pdr, Delay Decision Tree algorithms.

The figure 8 provides a comprehensive comparison of the performance metrics across multiple machine learning models, including the proposed algorithms, Random Forest, and Decision Tree. The figure 9 shows the user interface or section of the application where users can select a test dataset for evaluation.

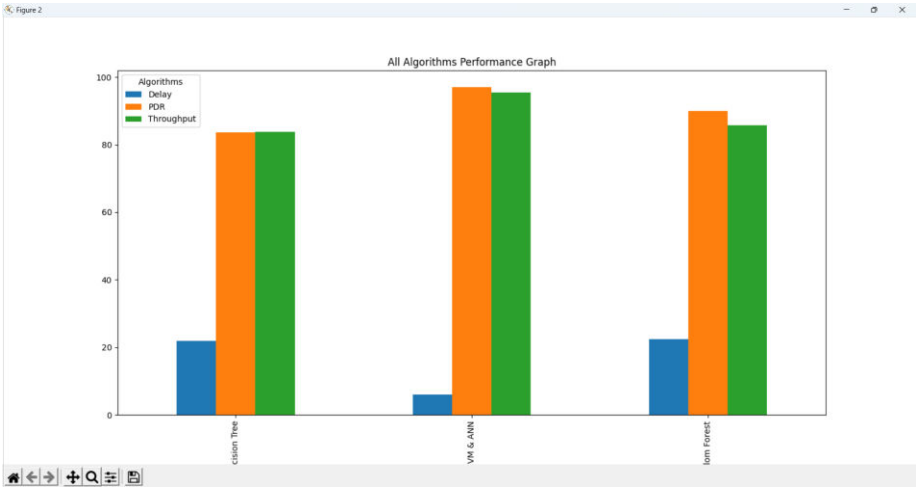


Figure 8: Displays the comparison plot for all the ml models performance metrics.

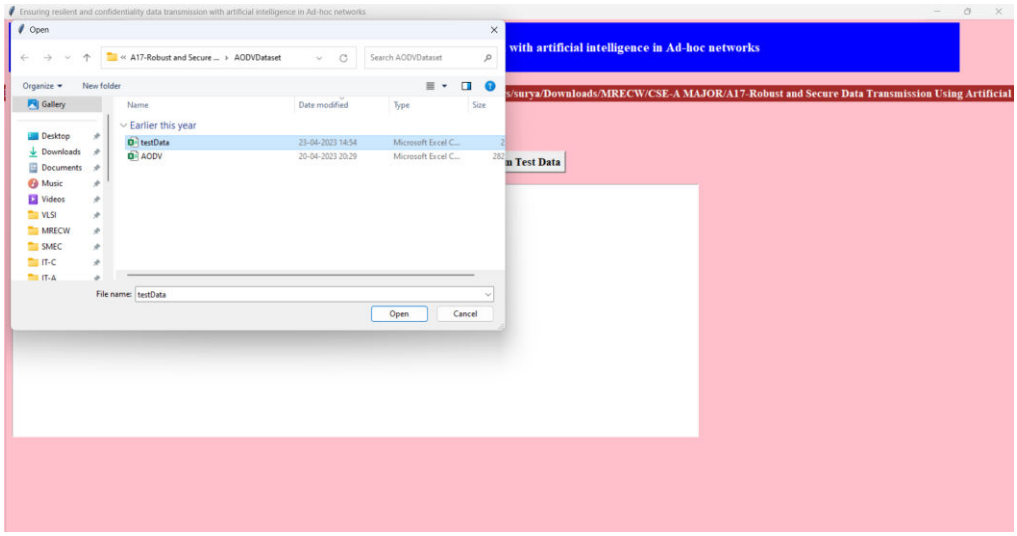


Figure 9: Displays the selection of Test Dataset.

The figure 10 illustrates the predictions generated by the chosen machine learning models on the selected test dataset. It provides insights into how well the models generalize to new, unseen data

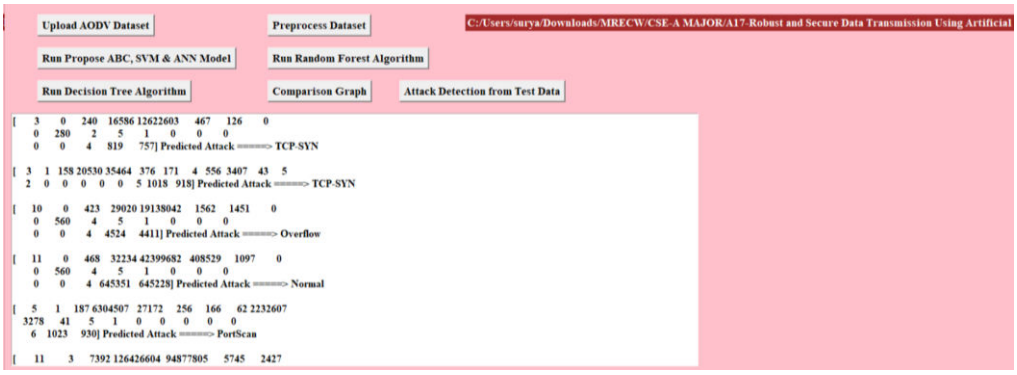


Figure 10: Displays the prediction of the test data.

5. CONCLUSION

The reasearch focused on ensuring resilient and confidential data transmission in ad-hoc networks using artificial intelligence has successfully addressed critical challenges in the domain of secure communication. The integration of artificial intelligence into ad-hoc networks has shown promising results in enhancing the resilience and confidentiality of data transmissions. The research has contributed to the development of robust mechanisms that dynamically adapt to changing network conditions, ensuring reliable and secure communication in challenging environments. The artificial intelligence algorithms employed in the research have demonstrated their effectiveness in identifying and mitigating security threats, such as malicious nodes or eavesdropping attempts, in real-time. The incorporation of adaptive encryption and routing strategies based on AI-driven insights has significantly bolstered the security posture of ad-hoc networks.

REFERENCES

[1] Alnumay, W.; Ghosh, U.; Chatterjee, P. A Trust-Based Predictive Model for Mobile Ad Hoc Network in Internet of Things. Sensors 2019, 19, 1467.
[2] Ghayvat, H.; Mukhopadhyay, S.; Gui, X.; Suryadevara, N. WSN- and IOT-Based Smart Homes and Their Extension to Smart Buildings. Sensors 2015, 15, 10350–10379.
[3] Masek, P.; Masek, J.; Frantik, P.; Fujdiak, R.; Ometov, A.; Hosek, J.; Andreev, S.; Mlynek, P.; Misurec, J. A Harmonized Perspective on Transportation Management in Smart Cities: The Novel IoT-Driven Environment for Road Traffic Modeling. Sensors 2016, 16, 1872. Deng, Y.-Y.; Chen, C.-L.; Tsaur, W.-J.; Tang, Y.-W.; Chen, J.-H. Internet of Things (IoT) Based Design of a Secure and Lightweight Body Area Network (BAN) Healthcare System. Sensors 2017, 17, 2919.
[4] Tamilselvan, L.; Sankaranarayanan, V. Prevention of Co-operative Black Hole Attack in MANET. J. Netw. 2008, 3, 13–20.
[5] Kang, B.-S.; Ko, I.-Y. Effective Route Maintenance and Restoration Schemes in Mobile Ad Hoc Networks. Sensors 2010, 10, 808–821.
[6] Himral, L.; Vig, V.; Chand, N. Preventing aodv routing protocol from black hole attack. Int. J. Eng. Sci. Technol. (IJEST) 2011, 3, 3927–3932.
[7] Panigrahi, R.; Borah, S.; Bhoi, A.K.; Ijaz, M.F.; Pramanik, M.; Kumar, Y.; Jhaveri, R.H. A Consolidated Decision Tree-Based Intrusion Detection System for Binary and Multiclass Imbalanced Datasets. Mathematics 2021, 9, 751.
[8] Papadimitratos, P.; Haas, Z. Secure routing for mobile ad hoc networks. In Proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002), San Antonio, TX, USA, 27–31 January 2002.
[9] Cai, R.J.; Li, X.J.; Chong, P.H.J. An Evolutionary Self-Cooperative Trust Scheme Against Routing Disruptions in MANETs. IEEE Trans. Mob. Comput. 2019, 18, 42–55.
[10] Djahel, S.; Nait-Abdesselam, F.; Zhang, Z. Mitigating Packet Dropping Problem in Mobile Ad Hoc Networks: Proposals and Challenges. IEEE Commun. Surv. Tutor. 2010, 13, 658–672.
[11] Gaur, L.; Singh, G.; Solanki, A.; Jhanjhi, N.Z.; Bhatia, U.; Sharma, S.; Verma, S.; Kavita; Petrovi´c, N.; Ijaz, M.F.; et al. Disposition of Youth in Predicting Sustainable Development Goals Using the Neuro-fuzzy and Random Forest Algorithms. Hum.-Cent. Comput. Inf. Sci. 2021, 11, 24.
[12] Gupta, P.; Goel, P.; Varshney, P.; Tyagi, N. Reliability factor-based AODV protocol: Prevention of black hole attack in MANET. In Smart Innovations in Communication and Computational Sciences; Springer: Singapore, 2019; Volume 851, pp. 271–279.
[13] Mohanapriya, M.; Krishnamurthi, I. Modified DSR protocol for detection and removal of selective black hole attack in MANET. Comput. Electr. Eng. 2014, 40, 530–538
[14] Gurung, S.; Chauhan, S. A survey of black-hole attack mitigation techniques in MANET: Merits, drawbacks, and suitability. Wirel. Netw. 2020, 26, 1981–2011.
[15] Seyedi, B.; Fotohi, R. NIASHPT: A novel intelligent agent-based strategy using hello packet table (HPT) function for trust Internet of Things. J. Supercomput. 2020, 76, 6917–6940.
[16] Thebiga, M.; SujiPramila, R. A New Mathematical and Correlation Coefficient Based Approach to Recognize and to Obstruct the Black Hole Attacks in Manets Using DSR Routing. Wirel. Pers. Commun. 2020, 114, 975–993.
[17] Lee, C.; Jeong, T. FRCA: A Fuzzy Relevance-Based Cluster Head Selection Algorithm for Wireless Mobile Ad-Hoc Sensor Networks. Sensors 2011, 11, 5383–5401.
[18] Gurung, S.; Chauhan, S. A dynamic threshold based approach for mitigating black-hole attack in MANET. Wirel. Netw. 2018, 24, 2957–2971

[19] Mohammadani, K.; Memon, K.A.; Memon, I.; Hussaini, N.N.; Fazal, H. Preamble time-division multiple access fixed slot assignment protocol for secure mobile ad hoc networks. Int. J. Distrib. Sens. Netw. 2020, 16, 1550147720921624.

[20] El-Semary, M.; Diab, H. BP-AODV: Blackhole Protected AODV Routing Protocol for MANETs Based on Chaotic Map. IEEE Access 2019, 7, 95197–95211.