AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
Peer Reviewed, Referred & Indexed Journal
E-ISSN:3069-0102
VOL.6, NO. 2(2026)

Deep Feature-Driven TAO Tree Fusion Model for Autonomous Intrusion Detection in Software-Defined 6G Network Slicing Architectures

B. L. N. Swamy[1], G. Udaykiran Bhargava[1], G. Raviraju[1], V. Ravindra Naik[1]
[1]Department of Electronics & Communication Engineering, Mother Teresa Institute of Science & Technology, Sanketika Nagar, Kothuru, Sathupally, Khammam, 507303, Telangana, India.

ABSTRACT

The evolution of 6G networks introduces significant demands for secure, adaptive, and high-performance communication systems. Conventional network monitoring and intrusion detection approaches, largely based on static rules and manual inspection, are insufficient for managing the complexity and scale of modern network environments. These methods often fail to detect advanced cyber threats and are ineffective in optimizing network performance under dynamic conditions. This research proposes a Machine Learning (ML)-driven framework based on Classification and Regression Tree (CART) principles for dual-purpose analysis, including attack classification and throughput prediction. The system integrates multiple ML techniques, such as Support Vector Machine (SVM), k-Nearest Neighbors (KNN), and a custom-designed Tree-based Adaptive Optimization (TAO) and an enhanced hybrid Deep Convolutional Neural Network (CNN) with TAO-CART (Deep CNN-TAO CART) architecture. The Deep CNN module performs automated feature extraction, while the TAO-CART ensemble improves classification and regression performance for complex network traffic patterns. These models are trained to accurately identify malicious traffic patterns while simultaneously predicting network throughput. To improve model performance, preprocessing techniques including Label Encoding, feature scaling through Standardization, and class balancing using Synthetic Minority Over-sampling Technique (SMOTE) are applied. This integrated approach enhances generalization capability and robustness across diverse network scenarios. Experimental results indicate that the Deep CNN – TAO CART ensemble model achieves superior classification accuracy along with reliable regression performance compared to individual models. Furthermore, the system is implemented through a web-based interface using the Flask framework, enabling real-time analysis and user interaction. The proposed solution provides a scalable and intelligent framework for improving network security and optimizing performance in next-generation communication systems.

Keywords: Intrusion Detection System (IDS), Machine Learning, Deep Learning, Classification and Regression Tree (CART), Deep CNN-TAO CART, Throughput Prediction, Network Traffic Classification, Flask Framework.

1. INTRODUCTION

Network monitoring and management depend on data collection protocols such as SNMP (Simple Network Management Protocol), NetFlow, IPFIX, and NETCONF (Network Configuration Protocol). Organizations design and implement customized tools using these protocols to obtain detailed insights into network performance and to effectively address faults and operational issues. Due to the rapid increase in demand for network services, the requirements placed on network components and monitoring systems have grown considerably. As a result, modern network devices continuously generate enormous amounts of data, including control information, statistical records, and user traffic, at an increasingly high rate.

Processing and analyzing such large-scale and complex datasets through manual or human-assisted methods demand significant expertise, time, and operational resources. With network infrastructures becoming larger and more complex, these traditional approaches are gradually becoming inefficient and difficult to maintain. Therefore, there has been an increasing transition toward automated and intelligent

solutions that can efficiently process high-dimensional data and support faster as well as more accurate decision-making in contemporary network environments.

A key approach enabling advanced management and monitoring of telecommunication networks is the concept of Software-Defined Networking (SDN), which introduces a centralized and programmable architecture for network control. This paradigm simplifies network configuration and enhances flexibility; however, it still requires continuous monitoring and efficient anomaly detection mechanisms. Centralized data collection offers significant advantages by enabling better coordination, aggregation, and correlation of network information. Despite these benefits, processing and analyzing such large-scale datasets presents substantial computational challenges, often referred to as "big data analytics".

To address these challenges, various tools and platforms have been developed to support the implementation, deployment, and utilization of large datasets. One such example is Platform for Network Data Analytics (PNDA), an open-source solution that enables efficient data collection, storage, and real-time analysis. Given the vast volume of network data, integrating Artificial Intelligence and Machine Learning techniques becomes essential for extracting meaningful insights and improving decision-making processes.

Continuous data collection plays a crucial role in effective network monitoring; however, it can increase device workload and impact network throughput, particularly within transmission and management networks. To mitigate these issues, adaptive data collection strategies can be employed, where polling intervals are dynamically adjusted based on network conditions. For instance, increasing the frequency of data collection upon detecting anomalies can enhance responsiveness while maintaining overall system efficiency.

2. RELATED WORK

The evolution of sixth-generation (6G) wireless communication networks has introduced unprecedented opportunities alongside complex cybersecurity challenges. The integration of advanced technologies such as artificial intelligence (AI), Open Radio Access Networks (O-RAN), terahertz communication, software-defined networking (SDN), and massive Internet of Things (IoT) connectivity has transformed network architecture while simultaneously expanding the attack surface. Existing research has largely focused on the use of machine learning (ML), deep learning (DL), and ensemble learning techniques to improve intrusion detection, anomaly detection, malware classification, and network optimization in 6G environments.

2.1 Machine Learning for 6G Network Optimization

Machine learning has emerged as a fundamental component in achieving the performance and intelligence requirements of 6G networks. Puspitasari et al. [3] examined the role of ML algorithms and their derivatives in addressing communication challenges and concluded that ML integration would be essential for future wireless systems. Similarly, Okere et al. [5] reviewed multiple 6G-enabling technologies, including digital twins, intelligent reflecting surfaces, visible light communication, blockchain, UAVs, and non-orthogonal multiple access, emphasizing that ML techniques are necessary for solving complex networking problems and optimizing resource allocation. Rekkas et al. [6] further categorized ML applications into supervised, unsupervised, and reinforcement learning approaches while identifying future research directions for ML-driven wireless communication systems.

2.2 Intrusion and Anomaly Detection in 6G Networks

Real-time intrusion detection remains a critical requirement for protecting increasingly interconnected 6G infrastructures. Nassreddine et al. [2] highlighted that the rapid expansion of smart devices and network connectivity has increased policy violations and security threats, making intrusion detection systems

indispensable for maintaining information security. Saeed et al. [10] extended this direction by developing an ensemble-learning-based anomaly detection framework that incorporated preprocessing, feature selection, and hybrid classification techniques across multiple benchmark datasets. Likewise, Ismail et al. [11] proposed an adaptive intrusion detection framework combining convolutional neural networks and attention-based XGBoost to improve learning efficiency and detection accuracy in high-dimensional wireless communication environments. These studies collectively demonstrate the growing reliance on intelligent detection systems for safeguarding 6G communication networks.

2.3 Deep Learning and Ensemble Security Models

Deep learning and ensemble methods have shown considerable effectiveness in detecting sophisticated cyber threats. Damaševičius et al. [4] investigated malware detection using stacked dense and convolutional neural networks combined with meta-learning approaches. Their results demonstrated that ensemble architectures could significantly improve classification performance compared with conventional machine learning methods. Mahmoud et al. [7] further explored deep-learning-driven security by integrating neural networks with physical layer security mechanisms to detect spoofing, jamming, and eavesdropping attacks. Their multi-stage framework achieved improved detection accuracy and enhanced communication reliability under hostile conditions. These findings reinforce the effectiveness of hybrid and ensemble learning models in strengthening cyber defense mechanisms.

2.4 Emerging Security Challenges in Advanced 6G Environments

The incorporation of intelligent and decentralized technologies has introduced new security challenges that extend beyond traditional cyberattacks. Kaur et al. [1] emphasized that technologies such as O-RAN, terahertz communication, and native AI introduce vulnerabilities including eavesdropping, supply chain attacks, model poisoning, and adversarial manipulation. To improve decision transparency and security resilience, they proposed integrating explainable AI techniques such as SHAP and LIME with advanced learning models. In addition, Zahid et al. [8] examined the growing threat of drone and swarm attacks, highlighting the need for efficient radio-frequency-based drone classification to support secure mobility and resource management in 6G systems. Their findings indicate that deep ensemble learning for drone signal analysis remains insufficiently explored.

2.5 Intelligent Network Monitoring and Software-Defined Security

The growing adoption of software-defined architectures has encouraged research into intelligent monitoring and automated security frameworks. Rzym et al. [9] proposed a deep-learning-based anomaly detection framework specifically designed for software-defined networks operating in 6G environments. Their approach combined dynamic telemetry with deep neural networks to automate traffic monitoring and strengthen network visibility. Such intelligent monitoring mechanisms are increasingly recognized as essential for managing the scale, complexity, and dynamic behavior of future communication infrastructures.

2.6 Research Gap

Although substantial progress has been achieved in machine learning-based intrusion detection, malware analysis, anomaly detection, and network optimization for 6G systems, existing studies predominantly address isolated security problems or specific attack scenarios. Most approaches rely on standalone detection models and lack an integrated framework capable of simultaneously addressing multiple evolving threats while maintaining transparency and adaptability. Furthermore, limited attention has been given to combining explainable AI, ensemble learning, and real-time adaptive security mechanisms within a unified 6G security architecture. Therefore, there remains a significant need for comprehensive and intelligent

security frameworks capable of ensuring robust, scalable, and resilient protection for future 6G communication networks.

3. PROPOSED METHODOLOGY

This research focuses on developing an integrated analytical framework capable of performing both security threat classification and network performance prediction within a unified system. Instead of treating intrusion detection and performance evaluation as separate tasks, the proposed approach combines classification and regression techniques to establish a relationship between network attacks and their impact on throughput. This enables simultaneous analysis of security conditions and performance behavior in modern communication networks. The framework uses ML models such as SVM, KNN, and a custom TAO Tree, Deep CNN- TAO ensemble model. These models are designed to classify different types of network attacks while predicting throughput based on network conditions. The system is implemented as a web-based platform using Flask, providing an interactive and user-friendly interface. It offers a complete pipeline that includes dataset upload, preprocessing, Exploratory Data Analysis EDA, model training, performance evaluation, and real-time prediction as demonstrated in Fig. 1. This integrated design ensures efficient handling of large-scale network data and supports informed decision-making for improving both network security and performance in next-generation environments.

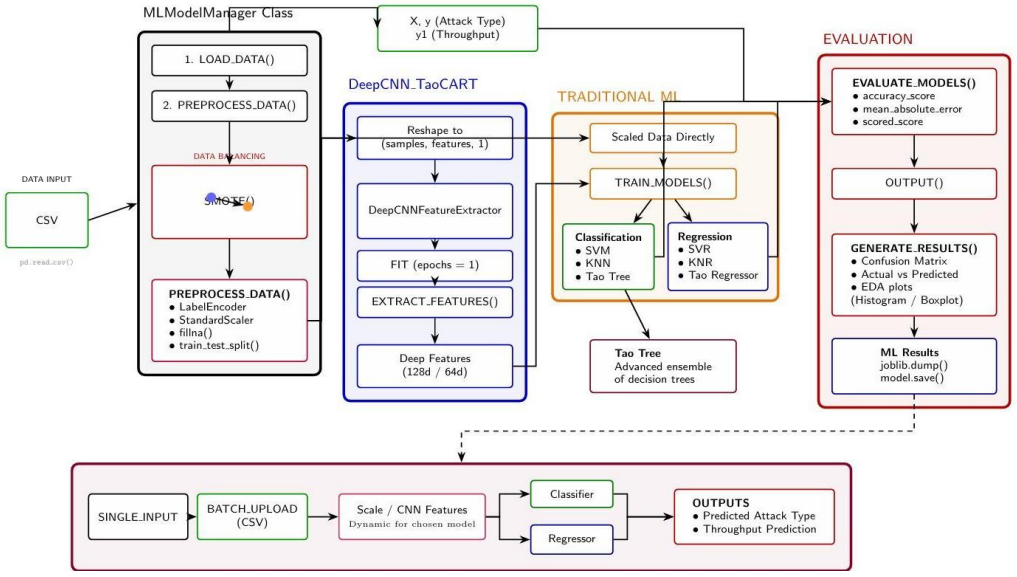


Fig. 1: Proposed system architecture

Data Acquisition and Upload: The process begins with the user uploading a network traffic dataset, typically in a CSV format, through the web application's main interface. The system is configured to validate the file type to ensure it is a .csv file, and it uses secure filename handling to prevent malicious uploads. This step acts as the primary entry point for all subsequent analysis and model training activities. Upon successful upload, the system saves the dataset to a dedicated uploads directory and stores its path and basic information, like its shape, in the user's session for continuous access throughout the workflow.

Data Preprocessing and Splitting: In the background, the ML Model Manager performs a series of essential preprocessing tasks to prepare the raw data for machine learning. First, it identifies and drops any irrelevant columns, such as "Unnamed" columns that often appear in datasets. Next, it handles categorical features, like different network services or protocols, by converting them into numerical representations using a Label Encoder. Missing values are filled using the mean of their respective columns. The prepared data is then split into two distinct parts: a training set (80%) and a testing set (20%). The data is also split

into two target variables: a categorical one for Intrusion Detection (Attack Type) and a numerical one for Performance Prediction (Throughput). Finally, the system addresses class imbalance in the intrusion detection dataset by applying SMOTE on the training data. This ensures that the machine learning models do not become biased toward the majority class (normal traffic) and can effectively detect rare attack instances.

EDA: Once the data is loaded, the system automatically redirects the user to the EDA page. This crucial step provides a quick and informative overview of the dataset's characteristics. The system's ML Model Manager generates several key visualizations, including histograms to show the distribution of attack types and network throughput, box plots to analyze the relationship between categorical features (like signal strength) and numerical ones (like throughput), and scatter plots to visualize relationships between continuous variables. A correlation heatmap is also generated to identify which features are most strongly related to the target variables (attack type and throughput). These plots are then displayed on the web page, allowing the user to gain an immediate understanding of the data's structure, potential issues like class imbalance, and key relationships before proceeding to model training.

Model Training and Selection: The user can choose from a variety of ML and DL models to train, including SVM, KNN, and a custom TAO Tree, Deep CNN – TAO Tree. Upon selection, the ML Model Manager trains a classifier for intrusion detection and a regressor for throughput prediction for each chosen model. To enhance efficiency and reusability, the trained models are saved as .pkl files using joblib. This allows the system to load pre-trained models in subsequent sessions, eliminating the need for retraining and significantly reducing processing time.

Performance Evaluation: After training, the system evaluates the performance of each model on the held-out test data. For the classification models, it calculates key metrics such as accuracy, precision, recall, and F1-score. It also generates a confusion matrix, which visually represents the number of correct and incorrect predictions for each attack type. For the regression models, it calculates metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R^2) to assess the accuracy of throughput predictions. These detailed performance results, including the visual plots, are then displayed on dedicated pages, allowing the user to compare the effectiveness of different models and choose the best one for their specific needs.

Prediction Interface: The final step provides two methods for making predictions on new data. The single prediction interface allows users to manually input values for each network feature via a form. The system preprocesses this single data point and feeds it to the selected trained models to predict the attack type and network throughput. The batch prediction interface enables users to upload a new CSV file containing multiple instances of network data. The system automatically preprocesses the entire file and returns a table with the predicted attack type and throughput for each instance, providing a powerful tool for large-scale network security and performance analysis.

Deep CNN Model

Deep CNN is a deep learning architecture designed to automatically learn important patterns and hidden relationships from input data. It is widely used for feature extraction because it can identify complex structures without requiring manual feature engineering. In this project, Deep CNN is used to extract high-level traffic features from 6G network data before applying the TAO Tree algorithm for classification and regression tasks. The integration of Deep CNN improves attack detection accuracy and throughput prediction by learning meaningful representations from network traffic patterns. Compared to traditional machine learning methods, Deep CNN provides better generalization and robustness for handling complex and high-dimensional network datasets.

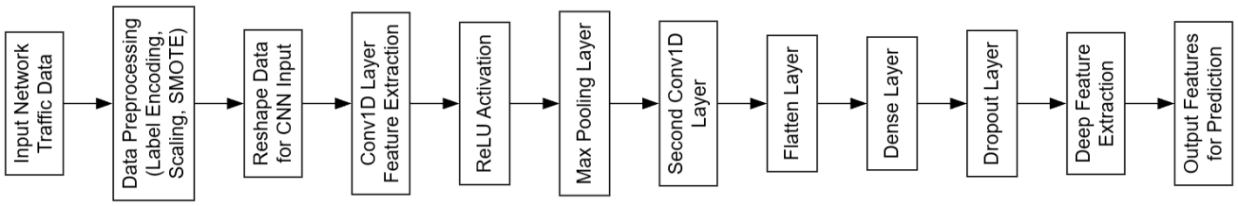


Fig. 2: Internal working process of Deep CNN

Step 1: Input Data Preparation

The pre-processed network traffic dataset is first normalized using StandardScaler to ensure all feature values are within a similar range. The scaled data is then reshaped into a three-dimensional format because CNN models require structured input data. This reshaped data becomes the input for the Deep CNN model.

Step 2: Convolution Layer

The convolution layer applies multiple filters to the input data to identify hidden traffic patterns and important network characteristics. Each filter extracts specific local features such as traffic variations, packet behavior, or abnormal activity patterns. This process helps the model automatically learn useful representations from the dataset.

Step 3: Activation Function

After convolution, the ReLU activation function is applied to introduce non-linearity into the model. ReLU converts negative values into zero while retaining positive values, helping the network learn complex relationships efficiently. This also improves training speed and prevents unnecessary computations.

Step 4: Max Pooling Layer

The max pooling layer reduces the dimensional size of extracted feature maps by selecting the most significant values. This operation decreases computational complexity and minimizes overfitting while preserving important traffic information. Pooling also helps the model focus on dominant network patterns.

Step 5: Flatten Layer

The flatten layer converts the pooled multi-dimensional feature maps into a one-dimensional feature vector. This transformation allows the extracted deep features to be processed by fully connected layers or external machine learning models. It acts as a bridge between CNN feature extraction and prediction stages.

Step 6: Dense Layer

The dense layer performs high-level learning using the extracted features obtained from previous layers. It combines and refines the learned information to generate meaningful deep representations of network traffic behavior. This improves the model's ability to distinguish between normal and malicious traffic.

Step 7: Dropout Layer

The dropout layer randomly disables certain neurons during training to prevent overfitting. This improves the generalization capability of the Deep CNN model and ensures better performance on unseen network data. It also increases the robustness and reliability of the prediction system.

Step 8: Deep Feature Extraction

After training, the Deep CNN model extracts optimized deep features from the network dataset. These features contain high-level information learned automatically from the traffic patterns. The extracted features are then passed to the TAO Tree classifier and regressor for final attack classification and throughput prediction.

Step 9: Final Prediction using TAO Tree

The TAO Tree model uses the deep features generated by the CNN for classification and regression tasks. The classifier predicts the attack type, while the regressor estimates network throughput. This hybrid Deep

CNN-TAO CART approach improves both prediction accuracy and system intelligence in 6G network environments.

TAO Tree CART Model

TAO Tree is a custom tree-based machine learning model inspired by the Random Forest algorithm and designed for efficient classification and regression tasks. It combines multiple tree estimators to improve prediction accuracy and robustness in network security analysis. In this project, the TAO Tree model is used to detect cyber-attacks and predict network throughput in 6G environments. The model performs well on complex and imbalanced datasets because it uses ensemble learning and data balancing techniques such as SMOTE. Compared to traditional decision trees, TAO Tree provides better generalization, improved stability, and reduced overfitting.

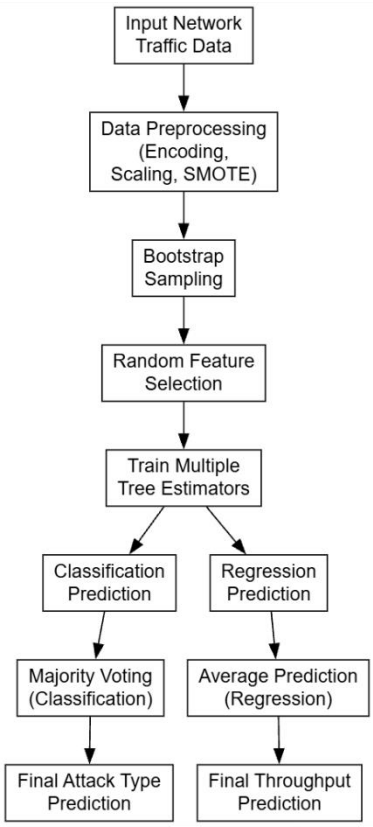


Fig. 3: Internal working flow of TAO Tree model

Step 1: Data Input and Preprocessing

The network traffic dataset is first pre-processed using Label Encoding and Standard Scaling techniques. Missing values are handled, and important numerical features are normalized for better model performance. SMOTE is also applied to balance minority attack classes in the training dataset.

Step 2: Bootstrap Sampling

TAO Tree creates multiple subsets of the training data using bootstrap sampling. Each tree receives a randomly generated sample from the original dataset. This randomness increases diversity among trees and improves the overall learning capability of the ensemble model.

Step 3: Random Feature Selection

For every tree, a random subset of input features is selected during training. This prevents the model from depending heavily on a few dominant features. Random feature selection improves model robustness and reduces correlation between trees.

Step 4: Tree Construction

Each estimator independently learns patterns from the selected training samples and features. The model builds decision boundaries based on network traffic characteristics and attack behavior. Separate TAO Tree models are developed for classification and regression tasks.

Step 5: Classification Prediction

For attack detection, all trained trees generate individual predictions for the input sample. The final attack class is selected using majority voting among the trees. This improves classification accuracy and reduces prediction errors.

Step 6: Regression Prediction

For throughput prediction, each tree produces a numerical output value independently. The final throughput value is calculated by averaging the predictions from all estimators. This ensemble averaging improves prediction stability and accuracy.

Step 7: Final Output Generation

The final model outputs both the predicted attack type and the estimated network throughput. The system supports real-time single prediction as well as batch prediction for large datasets. The generated results are displayed through the Flask-based web interface.

4. RESULTS AND DESCRIPTION

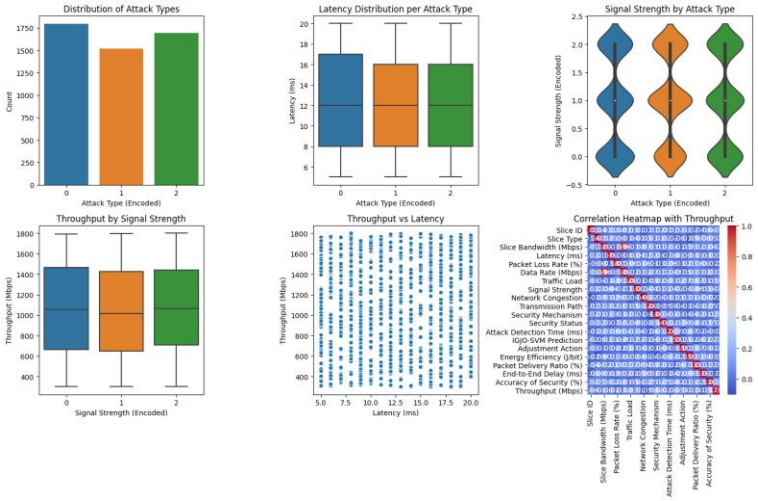


Fig. 4: Data Analysis of Network Dataset. (a), (b), (c), (d), (e), (f).

Fig. 4 Exploratory data analysis of network data. countplot of attack types (toprow-left). Boxplot of latency distribution per attack type (toprow - middle). Voilin plot of signal strength by attack type (toprow-right). Boxplot of throughput by signal strength (bottom row - left). Scatter plot of throughput vs latency (bottom row - middle). Correlation heatmap with throughput (bottom row - right).

Fig a: Distribution of Attack Types

- This bar chart displays the distribution of three attack types (encoded as 0, 1, and 2) based on their count. The x-axis represents the attack types, and the y-axis shows the count. The bars indicate that each attack type has a different frequency, with attack type 0 having the highest count, followed by type 2, and type 1 having the lowest.

Fig b: Latency Distribution per Attack Type

- This box plot illustrates the latency distribution (in milliseconds) for the three attack types (encoded as 0, 1, and 2). The x-axis represents the attack types, and the y-axis shows the latency. The boxes and whiskers provide a summary of the median, quartiles, and range of latency values for each attack type, showing variations in latency across the types.

Fig c: Signal Strength by Attack Type

- This violin plot depicts the signal strength (encoded values) for the three attack types (encoded as 0, 1, and 2). The x-axis represents the attack types, and the y-axis shows the signal strength. The width of each violin indicates the density of data points, highlighting the distribution and central tendency of signal strength for each attack type.

Fig d: Throughput by Signal Strength

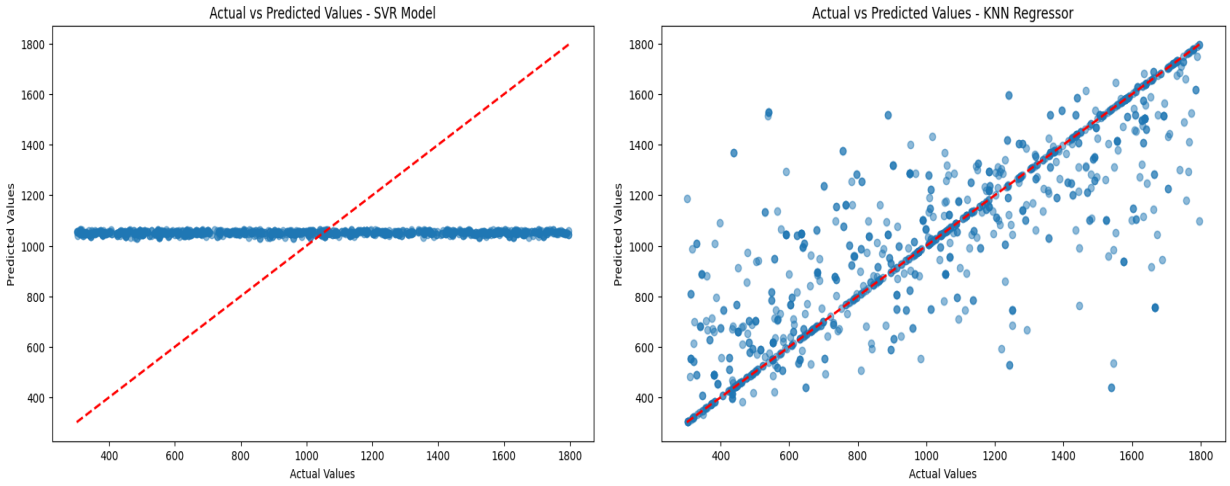
- This box plot shows the throughput (in Mbps) across different signal strength levels (encoded as 0, 1, and 2). The x-axis represents signal strength, and the y-axis shows throughput. The boxes and whiskers summarize the median, quartiles, and range of throughput values, indicating how throughput varies with signal strength.

Fig e: Throughput vs Latency

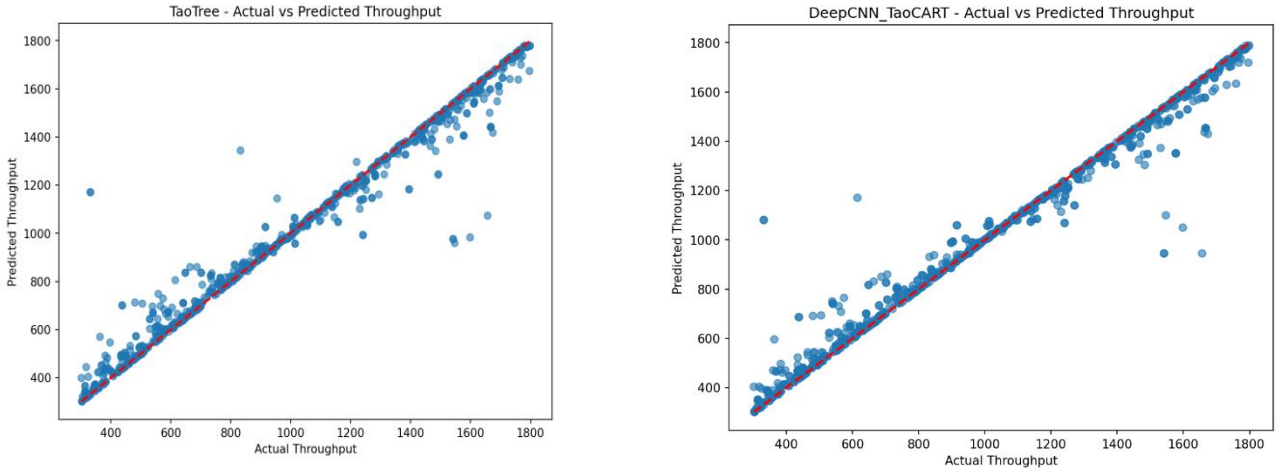
- This scatter plot examines the relationship between throughput (in Mbps) and latency (in milliseconds). The x-axis represents latency, and the y-axis represents throughput. The scattered points suggest a pattern or correlation between these two metrics, with a higher density of points around certain latency and throughput combinations.

Fig f: Correlation Heatmap with Throughput

- This heatmap displays the correlation coefficients between throughput and various other metrics (e.g., Slice ID, Slice Bandwidth, Packet Loss Rate). The x-axis and y-axis list the metrics, with color intensity (from blue to red) indicating the strength and direction of the correlation, ranging from -1 to 1. Stronger correlations are shown in warmer colors.



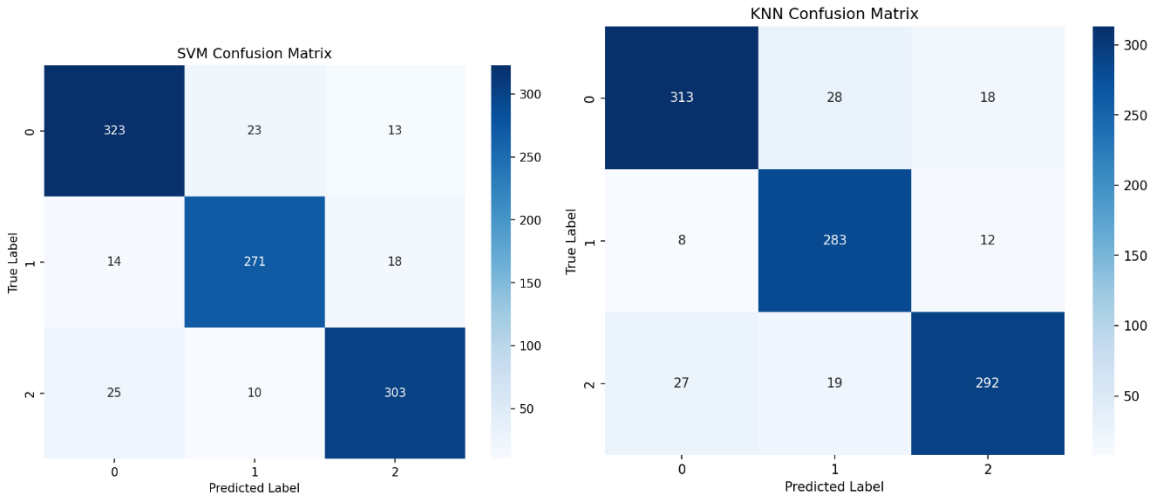
(a) (b)



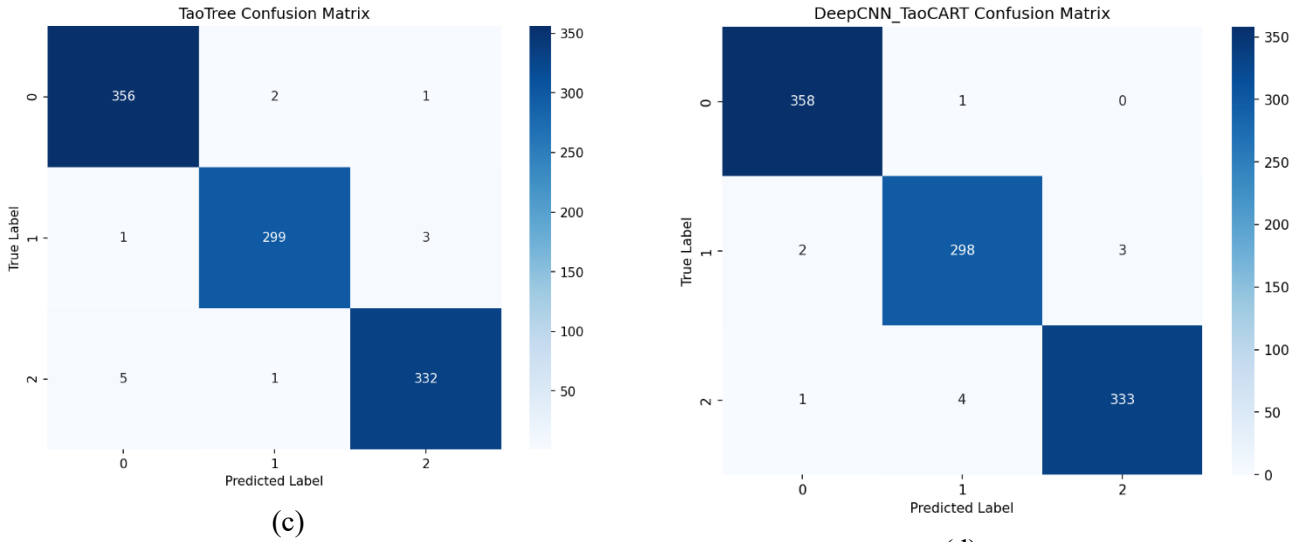
(c) (d)

Fig. 5: Scatter plot of actual vs predictions obtained using (a) SVM model.
(b) KNN model. (c) Custom TAO Tree Regressor, (d) Deep CNN – TAO Tree Regression

The scatter plots in Fig. 5 illustrate the relationship between actual and predicted values for four regression models: (a) SVM, (b) KNN, (c) Custom TAO Tree Regressor, and (d) Deep CNN–TAO Tree Regression. In the SVM plot, the predicted values appear nearly constant, forming a flat trend that indicates poor predictive capability and an inability to capture variations in the actual data. In contrast, the KNN model shows predictions distributed closely around the diagonal reference line, demonstrating strong agreement between actual and predicted values and indicating better regression performance. The Custom TAO Tree Regressor further improves the prediction alignment, showing reduced dispersion and better adaptability to nonlinear patterns in the dataset. Among all models, the Deep CNN–TAO Tree Regression model exhibits the closest clustering of points along the diagonal line, highlighting its superior predictive performance, higher accuracy, and stronger generalization capability compared to the other approaches.



(a) (b)



(c) (d)

Fig. 6: Confusion matrix obtained using (a) SVM model.
(b) KNN model. (c) Custom TAO Tree Classifier, (d) Deep CNN – TAO Tree Model.

The confusion matrices in Fig. 6 present the classification performance of four models: (a) SVM, (b) KNN, (c) Custom TAO Tree Classifier, and (d) Deep CNN–TAO Tree Model for detecting DDoS, Eavesdropping, and Spoofing attacks. The SVM confusion matrix demonstrates weak classification capability, with a large number of misclassifications among the attack categories. For instance, many Spoofing samples are incorrectly classified as Eavesdropping, indicating poor class separation and low detection reliability. The KNN model achieves improved classification accuracy, correctly identifying a higher number of Eavesdropping instances; however, it still exhibits notable confusion in distinguishing DDoS attacks from other classes. The Custom TAO Tree Classifier significantly enhances performance, producing a high number of correct predictions for all three attack classes while minimizing classification errors. Among all approaches, the Deep CNN–TAO Tree Model achieves the best overall classification results, with the majority of samples accurately classified along the diagonal of the confusion matrix. This demonstrates its superior capability in learning complex attack patterns, reducing false predictions, and improving cybersecurity threat detection accuracy.

Table 1: Performance evaluation obtained using existing SVM, KNN regressors and proposed TAO Tree regression models.

Model/Metric	MAE	MSE	RMSE	R2-score
SVR Model	370.871	18.8609	431.828	0.009
KNN Model	133.998	49133.987	221.662	0.739
Custom TAO Tree Regressor	0.0297	5787.744	76.007	0.969
Deep CNN-TAO Tree Regression	27.475	5288.521	72.722	0.972

Table 1 presents performance metrics for four different regression models evaluated on a dataset. The metrics included are Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R2 Score.

The models listed are:

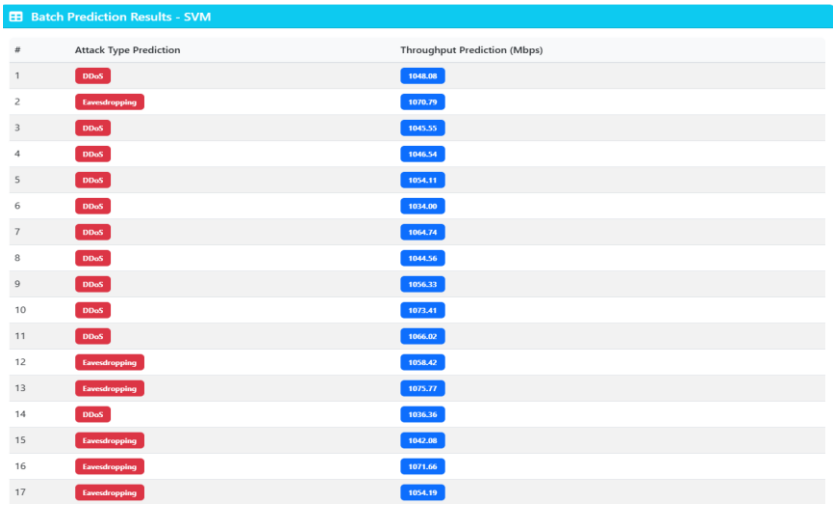
- **SVR Model**: MAE is 370.851, MSE is 18.8609, RMSE is 431.828, and R2 Score is -0.009, indicating poor predictive performance with a negative R2 score suggesting the model is not better than a mean baseline.
- **KNN Regressor**: MAE is 133.998, MSE is 49133.987, RMSE is 221.662, and R2 Score is 0.739, indicating a significant improvement in predictive accuracy and a substantial positive R2 score.
- **Custom TAO Tree Regressor**: MAE is 29.042, MSE is 5787.744, RMSE is 76.007, and R2 Score is 0.969, demonstrating the best performance among the models with the lowest errors and a very high R2 score, suggesting excellent predictive capability.
- **Deep CNN-TAO Tree Regressor**: MAE is 27.475, MSE is 5288.521, RMSE is 72.722, and R2 Score is 0.972, demonstrating the best performance among the models with the lowest errors and a very high R2 score, suggesting excellent predictive capability.

Table 2: Performance evaluation obtained using existing SVM, LR, KNN regressors and proposed TAO Tree classification models.

Algorithm	Accuracy	Precision	Recall	F1-Score
SVC Model	89.7%	89.7%	89.7%	89.7%
KNN Model	88.9%	88.907%	88.85%	88.876%
Custom TAO Tree Classifier	98.7%	98.7%	98.7%	98.7%
Deep CNN-TAO Tree Model	98.9%	98.9%	98.9%	98.9%

Table 2 presents performance metrics for four different classification algorithms evaluated on a dataset. The metrics included are Accuracy, Precision, Recall, and F1-Score.

- **SVC Model**: Accuracy is 89.7%, Precision is 89.7%, Recall is 89.7%, and F1-Score is 89.7%, indicating relatively average performance across all metrics, suggesting limited ability to correctly classify instances.
- **KNN Classifier**: Accuracy is 88.9%, Precision is 88.907%, Recall is 88.857%, and F1-Score is 88.876%, demonstrating strong performance with high values across all metrics, indicating effective classification.
- **Custom TAO Tree Classifier**: Accuracy is 98.7%, Precision is 98.7%, Recall is 98.7%, and F1-Score is 98.7%, showcasing the best performance among the algorithms with near-perfect scores, suggesting excellent classification capability.
- **Deep CNN-TAO Tree Classifier**: Accuracy is 98.9%, Precision is 98.9%, Recall is 98.9%, and F1-Score is 98.9%, showcasing the best performance among the algorithms with near-perfect scores, suggesting excellent classification capability.



Batch Prediction Results - SVM

#	Attack Type Prediction	Throughput Prediction (Mbps)
1	DDoS	1048.08
2	Eavesdropping	1070.79
3	DDoS	1045.55
4	DDoS	1046.54
5	DDoS	1054.11
6	DDoS	1034.00
7	DDoS	1064.74
8	DDoS	1044.56
9	DDoS	1056.33
10	DDoS	1073.41
11	DDoS	1066.02
12	Eavesdropping	1058.42
13	Eavesdropping	1075.77
14	DDoS	1036.36
15	Eavesdropping	1042.08
16	Eavesdropping	1071.66
17	Eavesdropping	1054.19

Fig. 7: Predictions on test data using proposed TAO model

Fig. 7 shows the batch prediction output generated by the system for test data inputs. The results are presented in a tabular format where each record includes the predicted attack type and corresponding throughput value. The predictions are produced after applying consistent preprocessing steps, including encoding and scaling, ensuring accuracy and uniformity across all inputs. This output provides a clear representation of how different network conditions are classified and how throughput is estimated, supporting effective analysis and validation of the system's performance.

5. Conclusion

This study presents a web-based intelligent network monitoring framework that performs both cyberattack classification and network throughput prediction using machine learning and deep learning techniques. The system integrates SVC, KNN, Custom TAO Tree, and Deep CNN–TAO Tree models to analyze network traffic efficiently. The implementation includes dataset preprocessing, exploratory data analysis, model training, evaluation, and real-time prediction through a Flask-based interface. Techniques such as label encoding, feature scaling, and SMOTE balancing were used to improve data quality and model accuracy. Experimental results show that traditional models such as SVC and KNN achieved moderate performance, while the proposed TAO-based models significantly improved prediction capability. The Custom TAO Tree Classifier achieved 98.7% classification accuracy, while the Deep CNN–TAO Tree Model achieved the highest accuracy of 98.9% with improved precision, recall, and F1-score. Similarly, in throughput prediction, the Deep CNN–TAO Tree Regression model achieved the best performance with an R^2-score of 0.972 and lower prediction errors compared to other models. The results demonstrate the effectiveness of combining deep learning with TAO Tree techniques for handling complex nonlinear network data. Overall, the proposed framework provides an efficient, scalable, and intelligent solution for enhancing both network security and performance monitoring in modern communication environments.

REFERENCES

[1] Kaur, N.; Gupta, L. Securing the 6G–IoT Environment: A Framework for Enhancing Transparency in Artificial Intelligence Decision-Making Through Explainable Artificial Intelligence. Sensors 2025, 25, 854. https://doi.org/10.3390/s25030854.

[2] Nassreddine, G.; Nassereddine, M.; Al-Khatib, O. Ensemble Learning for Network Intrusion Detection Based on Correlation and Embedded Feature Selection Techniques. Computers 2025, 14, 82. https://doi.org/10.3390/computers14030082.

[3] Puspitasari, A.A.; An, T.T.; Alsharif, M.H.; Lee, B.M. Emerging Technologies for 6G Communication Networks: Machine Learning Approaches. Sensors 2023, 23, 7709. https://doi.org/10.3390/s23187709.

[4] Damaševičius, R.; Venčkauskas, A.; Toldinas, J.; Grigaliūnas, Š. Ensemble-Based Classification Using Neural Networks and Machine Learning Models for Windows PE Malware Detection. Electronics 2021, 10, 485. https://doi.org/10.3390/electronics10040485.

[5] Okere, E.E.; Balyan, V. Sixth Generation Enabling Technologies and Machine Learning Intersection: A Performance Optimization Perspective. Future Internet 2025, 17, 50. https://doi.org/10.3390/fi17020050.

[6] Rekkas, V.P.; Sotiroudis, S.; Sarigiannidis, P.; Wan, S.; Karagiannidis, G.K.; Goudos, S.K. Machine Learning in Beyond 5G/6G Networks—State-of-the-Art and Future Trends. Electronics 2021, 10, 2786. https://doi.org/10.3390/electronics10222786.

[7] Mahmoud, H.; Ismail, T.; Baiyekusi, T.; Idrissi, M. Advanced Security Framework for 6G Networks: Integrating Deep Learning and Physical Layer Security. Network 2024, 4, 453-467. https://doi.org/10.3390/network4040023.

[8] Zahid, M.U.; Nisar, M.D.; Fazil, A.; Ryu, J.; Shah, M.H. Composite Ensemble Learning Framework for Passive Drone Radio Frequency Fingerprinting in Sixth-Generation Networks. Sensors 2024, 24, 5618. https://doi.org/10.3390/s24175618.

[9] Rzym, G.; Masny, A.; Chołda, P. Dynamic Telemetry and Deep Neural Networks for Anomaly Detection in 6G Software-Defined Networks. Electronics 2024, 13, 382. https://doi.org/10.3390/electronics13020382.

[10] Saeed, M.M.; Saeed, R.A.; Abdelhaq, M.; Alsaqour, R.; Hasan, M.K.; Mokhtar, R.A. Anomaly Detection in 6G Networks Using Machine Learning Methods. Electronics 2023, 12, 3300. https://doi.org/10.3390/electronics12153300.

[11] Ismail, W.N. A Novel Metaheuristic-Based Methodology for Attack Detection in Wireless Communication Networks. Mathematics 2025, 13, 1736. https://doi.org/10.3390/math13111736.