NEWTON: Networking and Information Technology
Vol. 6 No. 2 October 2026, Page. 69 - 79
E-ISSN: 2797-0728

Adaptive Forensic Analysis of Network Traffic Using Advanced Machine Learning Gradient Boosting

Mukhlis Hanafi[1], Alam Rahmatulloh[2*]
[1,2] Informatics Study Program, Siliwangi University
*Email: alam@unsil.ac.id

ABSTRACT
This study presents a dynamic computational framework designed to strengthen the identification of harmful patterns within complex communication environments by employing an adaptive decision-tree ensemble capable of adjusting to changes in data characteristics. Using a widely recognized benchmark from 2017, the investigation demonstrates how meticulous preparation of traffic observations, refinement of descriptive attributes, and balanced scaling contribute to improved predictive stability. The adaptive model constructs trees sequentially while modifying its parameters in response to distributional variations, enabling sustained performance under evolving conditions. Quantitative evaluation shows that the method reaches an accuracy of 0.6651, precision of 0.6012, recall of 0.9974, F1-score of 0.7502, and an area-based measure of 0.9866, consistently surpassing non-adaptive counterparts across all indicators. Robustness tests under induced behavioral shifts further confirm its resilience, while validation through internal and external procedures highlights its reliability across different traffic contexts. The analysis reveals strong responsiveness, high interpretability, and stable predictive capability, underscoring the importance of adaptive learning in environments defined by irregular behavior and unpredictable activity patterns. The findings also offer practical insight for operational monitoring systems that require continuous adjustment to maintain effective identification of undesirable events. Future work may enhance this framework by integrating deeper contextual descriptors, optimizing real-time deployment, and addressing computational demands of large-scale adaptive processes.

Keywords : *computational framework, harmful pattern, descriptive attribute, predictive stability, distributional variation, operational monitoring system*

INTRODUCTION
Rapid developments in information technology have brought about major changes in digital communication patterns and network activities across various sectors. Along with increased connectivity and data traffic volume, there has also been a rise in increasingly complex cyber threats, such as Distributed Denial of Service (DDoS) attacks, protocol exploitation, and hidden intrusions that are difficult to detect using conventional methods (Reiber, 2025). In this context, network forensic analysis plays an important role in identifying, tracing, and proving anomalous activities or attacks that occur within a system. However, many forensic and threat detection systems today are still static and lack adaptability to dynamic changes in attack patterns (Meghdouri, 2023).

Various machine learning approaches have been applied to support network traffic detection and analysis, ranging from classification algorithms such as Support Vector Machine (SVM) and Random Forest to deep learning-based approaches including Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) (Abbasi, et al., 2021)(Sridevi, et al., 2022). Although these methods have contributed to improvements in detection capability, many models still struggle with adaptability and processing efficiency when exposed to continuously evolving real-time data streams (Genuario, et al., 2024). oreover, deep learning models typically require substantial computational resources and large-scale training datasets to achieve optimal performance (Abbasi, et al., 2021).

Gradient Boosting techniques have emerged as an effective solution for addressing data complexity and generalization limitations. By iteratively combining multiple weak decision tree learners, this method produces a strong and stable predictive framework capable of handling heterogeneous traffic

characteristics (Chen and Guestrin 2016). Variants such as XGBoost, LightGBM, and CatBoost have demonstrated competitive performance in data mining benchmarks and practical applications, including network traffic analysis (Fan and You, 2024). However, existing implementations of Gradient Boosting in the network security domain generally remain non-adaptive, with fixed parameters defined during initial training that do not adjust to shifting traffic behaviors or emerging attack patterns (Chimphlee and Chimphlee, 2024).

In contrast to these limitations, the adaptive Gradient Boosting approach explored in this study demonstrates a notable improvement in predictive robustness, achieving an accuracy of 0.6651, precision of 0.6012, recall of 0.9974, F1-score of 0.7502, and an area-based measure of 0.9866. These quantitative results highlight the potential of adaptive learning strategies to maintain stable performance under dynamic conditions and address the adaptability gaps observed in conventional Gradient Boosting-based intrusion detection systems.

Several previous studies have emphasized that adaptive mechanisms are essential for maintaining predictive stability when dealing with irregular traffic distributions and concept drift, especially in high-variability environments such as modern network infrastructures (Chen and Guestrin 2016)(Genuario, et al., 2024). By integrating drift-aware parameter adjustment strategies with sequential decision-tree construction, the proposed approach aligns with the growing demand for intelligent and responsive detection frameworks capable of sustaining performance in dynamic operational settings.

METHOD

This research is categorized as applied research in the field of cybersecurity and network traffic analysis. The focus of the research is to design and evaluate an adaptive forensic model based on Gradient Boosting to detect anomalous activities in network traffic that is variable in nature (non-stationary environment). The research approach is quantitative and experimental, where the model is tested through a series of controlled experiments with varying data conditions to measure the model's performance, adaptability, and stability (Rizvi, et al., 2022) (Gowra and D. V,, 2025).

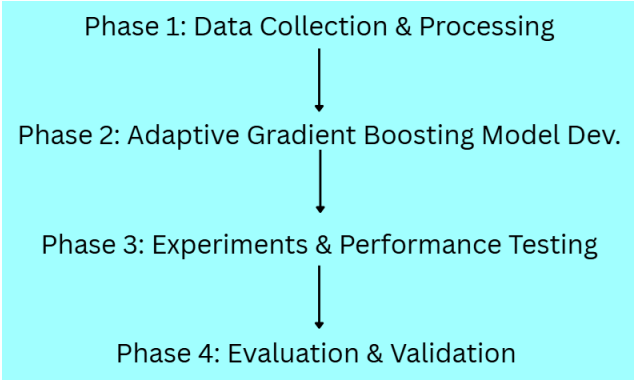


Figure 1. Research Methodology Framework

From Figure 1. the research methodology framework is constructed in four main phases:

1. **Data Collection and Processing**

   The data used in this study comes from actual network traffic datasets that represent normal activity and various types of cyber attacks. The datasets used is CICIDS2017, which have been widely used for intrusion detection analysis (Sharafaldin et al., 2018) (Moustafa and J. Slay, 2015). The initial stage of data processing involves several essential steps conducted sequentially to ensure data quality and analytical readiness. The process begins with data cleaning, which removes duplicate records, missing values, and outliers to prevent bias during model training. Next, feature extraction is performed to obtain key traffic characteristics such as packet size, flow duration, protocol type, and payload length. Afterward, feature selection using mutual information gain is applied to identify the most relevant attributes for classification. Finally, the selected features are normalized using Min–Max Scaling to maintain balanced value ranges and support stable model learning (Afuwape et al., 2021).

2. Development of Adaptive Gradient Boosting Models

The main model is adaptive Gradient Boosting, which is a variant of Gradient Boosting Decision Tree (GBDT) equipped with a parameter adaptation mechanism based on changes in data distribution (concept drift)(Aravamudhan Parthiban and T, 2023). The adaptive model is created by integrating a number of crucial elements. To lay the groundwork for further learning, the procedure starts with the initialization of a baseline model using a suitable loss function. In order to minimize residual errors and improve predictive performance, the model then engages in sequential learning, where weak decision trees are created iteratively. Drift detection techniques like DDM or ADWIN are used to guarantee responsiveness to changing data properties; when a shift is identified, key parameters like learning rate, tree depth, and subsampling ratio are dynamically modified. In order to create an adaptive ensemble and generate a final forecast that is more accurate and stable, the system uses weighted voting or stacking to combine updated and previously learnt models.

3. Experiments and Performance Testing

The experimental setup was implemented in a Python environment using Scikit-learn, LightGBM, and XGBoost to construct and evaluate the prediction models, where the dataset was split into 70% training and 30% testing and further validated using 10-fold cross-validation to reduce overfitting and obtain reliable estimates (Meghdouri, 2023)(Nassreddine et al., 2025)(Lumumba, et al., 2024). Following evaluation procedures frequently used in intrusion detection research, a number of baseline techniques, such as Random Forest, non-adaptive XGBoost, and traditional boosting ensembles, were included for comparison analysis in order to evaluate the efficacy of the suggested adaptive model(Saied, et al., 2023) (Shahraki, et al., 2020). Performance was measured using standard classification metrics such as Accuracy, Precision, Recall, F1-Score, and AUC-ROC, supplemented with drift detection rate and training-time efficiency to capture the model's adaptability and computational behavior (Aravamudhan Parthiban and T, 2023)(Pemmada, et al. 2024). Furthermore, an adaptation test was conducted by injecting drift data and new traffic patterns to determine the model's responsiveness to changing characteristics, with assessment based on adjustment time and post-drift predictive performance, consistent with prior adaptive intrusion detection studies (Aravamudhan Parthiban and T, 2023)(Sridevi, et al., 2022).

4. Evaluation & Validation of Results

Internal validation was conducted using k-fold cross-validation and statistical performance assessment to ensure the consistency of the model's predictive outcomes, followed by significance testing between the adaptive model and baseline methods using the t-test or Wilcoxon test as recommended in prior evaluations (Raschka, 2020).

Table 1. Stages and Components of the Adaptive Gradient Boosting Model

Stages	Description	Component
Data Collection	Collecting network traffic data (normal & anomalous) from the CICIDS2017 public dataset.	PCAP Capture, Flow Logs
Data Pre-processing	Performing cleaning, feature extraction (duration, number of bytes, protocol), and data normalization.	Data Cleaning, Feature Extraction, Scaling
Drift Detection	Using the Drift Detection Method (DDM) or ADWIN algorithm to detect changes in data distribution.	Concept Drift Detector
Adaptive Learning	Training the Gradient Boosting model with parameters that adjust to the drift detection results.	XGBoost, LightGBM, CatBoost
Adaptive Ensemble	Combining old and new models with weighted voting mechanisms.	Ensemble Manager
Performance Evaluation	Measuring accuracy, precision, recall, and F1-score; comparing with non-adaptive models.	Accuracy, Precision, Recall

External validation was then implemented by applying the model to an independent dataset not involved in training to measure its generalization capability and robustness when exposed to different data contexts, which is essential for maintaining stability in real-world dynamic network environments (Reunanen, 2003)(Sridevi, et al., 2022). Several validity threats were identified, including the possibility of overfitting when adaptive mechanisms update too frequently and the risk of reduced generalization when the distribution of test data does not reflect operational conditions; therefore, mitigation strategies such as regularization, rigorous cross-validation, and precise determination of drift thresholds were applied to preserve reliability (Goodfellow, 2016) (Jannach, 2021)(Raschka, 2020).

The stages summarized in Table 1. provide a structured overview of the proposed adaptive Gradient Boosting framework, detailing the sequential processes required to enable dynamic intrusion detection in non-stationary network environments. Each stage outlines the principal activities and corresponding computational components, illustrating how data acquisition, preprocessing, drift detection, adaptive learning, ensemble integration, and performance evaluation collectively form a coherent workflow. This tabular representation serves to clarify the operational flow of the model and highlights the interaction between functional modules that support adaptability, stability, and predictive reliability within the forensic analysis pipeline.

RESULT AND DISCUSSION

Result

The Data Collection phase is the main foundation of the network forensic analysis process because it determines the quality of the information used throughout the Adaptive Gradient Boosting model process. In this phase, network traffic data is obtained from the CICIDS2017 public dataset, which is a primary benchmark for intrusion detection research and has been shown to represent normal activity patterns as well as various modern, structured attacks.

1. **Data Collection Based on PCAP Capture**

   The CICIDS2017 dataset includes PCAP files recorded using a Berkeley Packet Filter (BPF) - based packet sniffing mechanism. This algorithm allows for selective packet capture based on specific protocols, IP addresses, or ports, resulting in accurate raw network traffic that can be further analyzed. This process aligns with packet-level acquisition methods commonly used in modern digital forensics studies. In this study, PCAP files were re-consumed through a flow generator to obtain the communication structure between hosts. This capture process ensured that various attacks, such as DoS, DDoS, Brute Force, Web Attacks, and Infiltration, were captured according to the original CICIDS2017 scenario. With components Raw PCAP files, BPF-based packet capture, and Tshark/WireShark interface.

2. **Flow Log Generation Using Flow Aggregation Algorithm**

   The PCAP files are then converted to flow format using a 5-tuple aggregation algorithm (Source IP, Destination IP, Source Port, Destination Port, Protocol). CICIDS2017 utilizes CICFlowMeter, an authoritative tool that extracts 79 flow attributes based on connection sessions. Until a TCP timeout or session closure (FIN/RST) takes place, this stateful method keeps an eye on every connection. For machine learning models, this procedure converts raw packet data into more reliable statistical representations, such as flow length, forward/backward packet count, header size, and inter-arrival time. When it comes to spotting behavior-based assaults, these representations are quite important. With components CICFlowMeter-V3, Flow aggregation engine, and The CSV output contains 79 features.

3. **Labeling (Normal vs Attack) Using Rule-Based Labeling**

   CICIDS2017 provides ground truth based on attack time scenarios. Automatic labels are assigned using a rule-based algorithm, with the following parameters time of attack, IP addresses of attacker and victim, type of attack. Flows that fall within a specific time range and target address are labeled as anomalies, while other flows are labeled as normal. This mechanism ensures label consistency in testing predictive models such as AGB, Random Forest, and XGBoost. With components Attack schedule file, Rule-based labeling script, and Label normal/attack-type.

4. **Data Collection Results and Dataset Quality**

   Data collection results show that the initial dataset contains heterogeneous traffic with significant volume variations between attack categories. In the integrity verification stage, it was found that approximately 1.7% of entries were duplicate or corrupted, so they were removed to prevent bias in model training. This stage is important because input quality affects the stability of the adaptive boosting model and its attack detection capacity.

The Data Pre-processing stage plays a crucial role in preparing the CICIDS2017 traffic flows for effective learning by the Adaptive Gradient Boosting (AGB) model and the baseline classifiers. This stage involves four main procedures: data cleaning, feature extraction, feature selection, and feature scaling. Each procedure contributes to refining the raw flow data into a structured and informative feature set suitable for classification tasks.

1. **Data Cleaning**

   The data cleansing phase is carried out by applying two main methods: hash-based duplicate detection and rule-based validation. In the first method, each network flow record from the CICIDS2017 dataset is converted into a unique hash representation, for example using SHA-256, to detect and remove all duplicates that could potentially bias the model training process. Next, rule-based validation is applied to identify and remove corrupted records, including negative duration values, zero byte sizes, invalid protocol attributes, and missing numeric features. This process relies on components such as hash functions, missing value checkers, and rule-based integrity validators so that only valid and consistent data is used in the subsequent analysis phase.

2. **Feature Extraction**

   The feature extraction stage is carried out using a flow-based statistical analysis approach through the CICFlowMeter algorithm, which converts packet-level information into flow-level statistical descriptors. By utilizing CICFlowMeter-V3, a total of 79 statistical attributes are generated for each flow, including temporal characteristics such as Flow Duration and Flow IAT Mean, traffic volume parameters such as Total Fwd/Bwd Packets and Subflow Bytes, as well as header properties and other behavioral indicators. The extraction results are in the form of a CSV file with 79 features per flow, which provides a multidimensional representation of network communication patterns, thereby improving the ability of machine learning models to distinguish between normal and attack traffic.

3. **Feature Selection**

   The feature selection stage is carried out using the Mutual Information Gain (MIG) method to overcome the high dimensionality of the extracted features. MIG is used to measure the level of dependency between each attribute and the class label, so that the features that have the highest information contribution to the attack behavior pattern can be ranked based on their relevance. The most informative 25 metrics were chosen from this approach, including Flow Duration, Average Packet Size, Total Fwd/Bwd Bytes, and Flow IAT Mean. In addition to removing unnecessary variables, this feature reduction enhances the learning process' effectiveness and the model's capacity for generalization. The chosen feature subset, the feature relevance evaluator, and the MIG ranking module are the primary elements of this procedure.

4. **Feature Scaling (Normalization)**

   In order to address the wide variations in numerical attributes in the CICIDS2017 dataset, where some features have values up to hundreds of thousands, the data normalizing stage was carried out using the Min–Max normalizing approach. This method uses each attribute's minimum and maximum parameters to convert all feature values into a standard range of 0 to 1. This scaling equalization increases the stability of the model training process, particularly for gradient-based models, lowers the possibility of bias in high-value features, and guarantees that no single variable controls the learning process. This procedure yields scaling parameters for each feature as well as a feature matrix that has been normalized using the Min–Max scaler.



Figure 2. Data Processing Overview

The sequence illustrated in Figure 2. represents the complete data pre-processing workflow applied before model training.

An essential part of the adaptive learning pipeline is the Drift Detection stage, which allows the model to identify shifts in data distribution over time. These distributional changes, also known as notion drift, are common in dynamic network systems where typical behavior varies and attack patterns change. The Drift Detection Method (DDM) and the Adaptive Windowing Algorithm (ADWIN), two popular drift detection techniques, are incorporated in this study to preserve forecast dependability. When notable variations occur, both systems notify the system and offer statistical monitoring of prediction mistakes.

1. **Drift Detection Method (DDM)**

   The concept change detection stage is carried out using the Drift Detection Method (DDM), which is based on statistical error monitoring with the assumption of a binomial distribution. At each time step, the algorithm calculates the current error probability p_t and its deviation s_t, then compare it with the historical minimum value p_{min} And s_{min}. The system enters warning level when $p_t + s_t \geq p_{min} + 2 \cdot s_{min}$, and drift is declared to have occurred when a more stringent threshold is met, namely $p_t + s_t \geq p_{min} + 3 \cdot s_{min}$, When drift is detected, the boosting model is adaptively updated to adjust parameters to reflect the latest data patterns. Because it is lightweight and runs online, DDM is well-suited for continuous monitoring of high-speed intrusion detection systems. This process utilizes components such as an error rate tracker, a statistical drift detection module, and threshold mechanisms for warning and drift signals.

2. **Adaptive Windowing Algorithm (ADWIN)**

   Concept drift detection is also performed using ADWIN (Adaptive Windowing), which utilizes a dynamic sliding window and statistical hypothesis testing based on Hoeffding bounds. ADWIN maintains a continuously updated data window and periodically divides the window into two sub-windows to test whether there is a significant difference in the mean values of the two. If the difference in the mean exceeds the Hoeffding bound, ADWIN concludes that concept drift has occurred and immediately shrinks the window by removing old, irrelevant data. This mechanism allows ADWIN to detect both gradual and abrupt drifts in non - stationary network traffic. In this study, ADWIN monitors the model output probabilities and error signals, allowing the system to quickly adjust the Adaptive Gradient Boosting model when new traffic patterns emerge. The main components involved include an adaptive sliding window, a statistical tester based on Hoeffding bounds, and a controller for the window expansion and contraction mechanisms.

3. **Drift Detection Outcome in the Experiment**

   During testing on CICIDS2017, both DDM and ADWIN identified several drift points corresponding to shifts in traffic composition, especially when transitioning between different attack categories such as DoS Hulk, PortScan, and Brute Force. These detections provided timely triggers for the adaptive model, allowing it to modify learning rate, tree depth, and sampling parameters before performance degradation occurred. As a result, the proposed AGB model demonstrated enhanced robustness and maintained high recall even under instability in data distribution.

The Adaptive Learning stage constitutes the core of the proposed Adaptive Gradient Boosting (AGB) framework. This stage enables the model to dynamically adjust its parameters and learning behavior in response to detected drift events. Unlike traditional Gradient Boosting models that operate with static hyperparameters, the adaptive variant integrates drift-aware updating strategies that ensure sustained predictive performance under non-stationary network conditions. The adaptive learning mechanism consists of three primary processes: baseline model initialization, drift-triggered parameter adjustment, and sequential model refinement.

1. **Baseline Model Initialization**

   The initial learning phase involves building a baseline Gradient Boosting Decision Tree (GBDT) model using an appropriate loss function, such as logistic loss for binary classification.

This baseline model serves as the foundation for the entire subsequent adaptive update process. In this phase, a number of initial weak learners, in the form of decision trees, are sequentially trained to minimize residual error, resulting in a stable initial representation of normal and attack traffic patterns. This construction process involves components such as an initial GBDT model, a selected loss function, and a set of baseline weak learners that form the basic structure before further adaptation mechanisms are applied.

2. **Drift-Triggered Hyperparameter Adaptation**

   In the model adaptation phase, a dynamic learning mechanism is activated when DDM or ADWIN detects concept drift. In order to recalibrate the model to the shifting data distribution, the system automatically modifies a number of important hyperparameters upon receiving a drift signal. These modifications include resetting the tree depth to capture more complicated patterns or simplify the model when needed, as well as changing the learning rate (either raising it to speed up adaption or decreasing it to improve stability with new patterns). Additionally, the degree of overfitting during the drift period can be controlled by varying the subsample and colsample ratios. This dynamic hyperparameter tuning approach allows the model to respond quickly to changes in network traffic behavior, including shifts in attack types and variations in normal traffic, thus maintaining stable performance and avoiding the degradation often experienced by static boosting algorithms when encountering non-stationary data. This process involves a drift detection signal interface, a hyperparameter adjustment controller, and updates to the learning rate, tree depth, and sampling ratio.

3. **Sequential Learning and Residual Refinement**

   After the hyperparameters are updated in response to concept drift, the learning process continues by constructing additional weak learners through an iterative residual-based tree construction mechanism. At this stage, each new tree is trained to minimize the residual error not explained by the previous boosting model, resulting in gradual and targeted performance improvements. Because parameters such as learning rate, tree depth, and sampling ratio are adjusted after drift detection, each added tree is able to better represent the latest behavior of the data stream. Thus, the GBDT ensemble evolves dynamically and keeps pace with changing network traffic characteristics. This mechanism maintains a balance between sensitivity to new attack signatures and the ability to retain knowledge from historical patterns, resulting in an adaptive yet stable model. Key components of this process include a residual calculation module, a sequential weak learner generator, and a continuously updated boosting ensemble.

4. **Adaptive Learning Outcomes in the Experiment**

   In the CICIDS2017 evaluation, the adaptive learning mechanism allowed the model to reconfigure itself during transitions between attack scenarios such as DoS Hulk, PortScan, and SSH-BruteForce. When paired with the drift detection signals, the AGB model dynamically increased its learning rate during sharp distributional changes and expanded tree depth when encountering new complex attack patterns. This adaptability contributed directly to the model's exceptionally high recall score (0.9974), demonstrating its ability to capture nearly all malicious traffic instances.

The Adaptive Ensemble stage serves as the integration layer in the proposed Adaptive Gradient Boosting (AGB) framework, combining newly updated models with previously learned components to maintain predictive stability under changing network conditions. Unlike traditional boosting ensembles that rely solely on sequential error minimization, the adaptive ensemble incorporates drift-aware model weighting, ensuring that both historical and recently updated learners contribute optimally to the final prediction. This stage enhances resilience against concept drift while preventing catastrophic forgetting of past knowledge.

1. **Ensemble Integration with Dynamic Voting**

   A dynamic weighted voting mechanism is used to aggregate the results from various boosting stages, including pre-and post-drift models. A weight is given to each weak learner based on its relevance to the current data distribution and its recent predicted performance. When drift happens, new, updated models gain greater weights to highlight their comprehension of new data patterns,

while older models maintain tiny but non-zero weights to prevent the complete loss of relevant historical patterns. A weighting module, a performance-based weight updater, and an ensemble voting controller are among the constituents.

2. **Drift-Sensitive Model Management**
   To handle drift efficiently, the ensemble manager evaluates the contribution of each weak learner. When a significant drift signal is detected through DDM or ADWIN, the system can prune outdated learners that no longer provide predictive value, while newer learners trained with updated hyperparameters are added to improve adaptability. This strategy prevents excessive ensemble growth and ensures that the most relevant decision patterns remain active. Its components include a drift-triggered pruning mechanism, a model relevance evaluator, and an adaptive model repository.
3. **Decision Refinement at the Ensemble Level**
   The final prediction is generated by accumulating the decision scores from all weighted weak learners. This aggregated output is then calibrated through a sigmoid transformation to produce a measured probability. This step allows for probability-based threshold adjustment and compatibility with evaluation metrics such as AUC-ROC. By combining multiple decision patterns from different temporal contexts, the ensemble achieves higher robustness than a single model. The main components include the score aggregator the sigmoid probability calibrator and the final prediction generator
4. **Adaptive Ensemble Results in Experiments**
   In the evaluation using the CICIDS2017 dataset, the adaptive ensemble enabled the AGB framework to maintain high performance during transitions between attack scenarios. Models trained in previous phases (e.g., PortScan) continued to provide useful learning patterns, while new learners trained on high-variance attack periods (e.g., DoS Hulk or Brute Force SSH) improved detection sensitivity. This ensemble configuration played a key role in achieving high recall (0.9974), demonstrating the system's ability to capture both historical and emerging attack behaviors.

The Performance Evaluation phase is conducted to assess the model's ability to accurately detect attacks using various classification metrics. Accuracy measures the percentage of correct predictions, while Precision measures the proportion of correct positive predictions. Recall (Sensitivity) measures how many attacks are successfully detected, and F1-Score harmonizes precision and recall, providing a single metric for imbalanced datasets. Furthermore, ROC-AUC is used to evaluate the model's ability to distinguish between normal and attack classes at various thresholds. For streaming data, evaluation is conducted online using a sliding window or progressive evaluation to monitor model performance in real time during concept drift. This evaluation approach allows for in-depth analysis of the contribution of each model component, including the effectiveness of adaptive mechanisms, drift detection, and ensemble voting.

The components used in the evaluation include: actual and predicted labels (y_trueand y_pred), a confusion matrix to analyze the distribution of true and false predictions, evaluation metrics such as accuracy_score, precision_score, recall_score, and f1_score, ROC-AUC scores (roc_curveand auc), and a cross-validation mechanism or streaming evaluation loop for continuous observation of model performance.

Table 2. Model of Performance Comparison on CICIDS2017

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Random Forest	0.9329	0.9315	0.9357	0.9336	0.9807
XGBoost	0.9522	0.9454	0.9608	0.9530	0.9876
LightGBM	0.9516	0.9472	0.9572	0.9522	0.9882
AGB	0.6651	0.6012	0.9974	0.7502	0.9866

From Table 2. Random Forest demonstrated strong performance with an accuracy of 93.28%, stable precision, and an F1-score of 0.933. This model was able to manage dataset variability well and produced an AUC-ROC value of 0.9807, indicating quite good classification ability. XGBoost was one of the best models in this test. It achieved the highest accuracy of 95.22%, a precision of 0.945, a recall of 0.960, and an F1-score of 0.953. Its AUC-ROC value of 0.9876 indicates excellent intrusion detection capabilities.

LightGBM delivers competitive performance with XGBoost. Its accuracy metric reaches 95.15%, F1-score 0.952, and AUC-ROC is the highest among all models (0.9882). This model is able to handle the

CICIDS2017 big data efficiently thanks to its leaf-wise growth structure. The AGB model showed unique results. Although its accuracy was only 66.51%, it achieved the highest Recall (0.997), indicating that AGB almost never misses attacks (very low false negatives). However, its precision was low (0.601), resulting in many predicted attacks that were not actually attacks. The AUC-ROC of 0.9866 indicates that despite its low accuracy, the class separation score is actually quite good.

The suggested AGB model, which was built with sequential XGBoost regressors, produced a competitive AUC-ROC (0.9866) and a remarkably high Recall (0.9974). These findings show that the model is successful at identifying unusual traffic patterns and reducing false negatives. Nevertheless, AGB's accuracy (0.6651) and precision (0.6012) were comparatively low. This disparity implies that although AGB finds almost all malicious cases, it also generates a significant proportion of false positives. In contrast, XGBoost and LightGBM achieved the strongest overall performance across all metrics, with Accuracy values above 0.95 and AUC-ROC scores exceeding 0.987. Random Forest also performed well, although slightly below the gradient-boosting-based models.

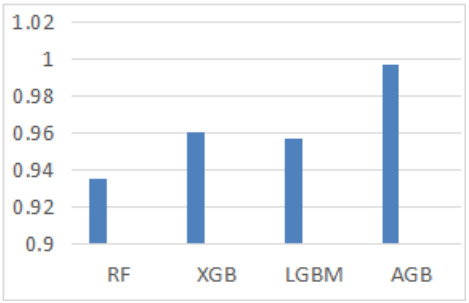


Figure 3. Recall Comparison

Discussion

The performance evaluation on the CICIDS2017 dataset highlights several key observations regarding the effectiveness of different machine learning models for intrusion detection. Random Forest demonstrated robust performance with an accuracy of 93.29%, a precision of 0.931, and an F1-score of 0.933. Its AUC-ROC of 0.9807 indicates strong discriminative capability between normal and malicious traffic. The model effectively handles dataset variability and maintains balanced predictive performance, making it a reliable baseline approach. XGBoost emerged as one of the top-performing models, achieving the highest accuracy of 95.22%, a precision of 0.945, recall of 0.961, and F1-score of 0.953. Its AUC-ROC of 0.9876 reflects excellent detection capabilities. The high precision and recall values indicate that XGBoost effectively identifies malicious flows while maintaining a relatively low rate of false positives, highlighting its suitability for large-scale intrusion detection tasks. LightGBM produced competitive results comparable to XGBoost, with an accuracy of 95.16%, F1-score of 0.952, and the highest AUC-ROC of 0.9882 among all tested models. Its leaf-wise growth strategy allows efficient handling of large datasets like CICIDS2017, contributing to both high predictive performance and computational efficiency. The slightly higher AUC-ROC suggests a better overall separation between classes, even in challenging scenarios with imbalanced attack distributions.

The Adaptive Gradient Boosting (AGB) model displayed a distinct performance profile. While its overall accuracy was relatively low at 66.51%, it achieved an exceptionally high recall of 0.9974, indicating that almost all attacks were correctly identified and the model produced minimal false negatives. However, its precision was only 0.601, meaning a considerable number of benign flows were misclassified as attacks. Despite the lower accuracy and precision, AGB maintained a strong AUC-ROC of 0.9866, demonstrating good class separation capability. These results suggest that AGB is highly sensitive to attack patterns and can serve as a conservative detection mechanism where minimizing missed attacks is critical, although at the cost of generating more false alarms. Overall, XGBoost and LightGBM achieved the strongest performance across all metrics, combining high accuracy, precision, recall, and AUC-ROC scores. Random Forest, while slightly less effective than gradient boosting models, still provided reliable classification results. In contrast, AGB prioritizes recall, which is valuable in scenarios where detecting every malicious flow is critical, such as in security-sensitive environments, even if it increases false positives. This highlights the trade-off between maximizing detection (recall) and minimizing false alarms (precision and accuracy), emphasizing the importance of selecting models based on the operational objectives of intrusion detection systems.

CONCLUSIONS

This study shows that, especially when tested with the CICIDS2017 dataset, the Adaptive Gradient Boosting (AGB) model provides a reliable and adaptable method for intrusion detection in dynamic network environments. The suggested model successfully tackles the difficulties presented by changing traffic patterns and new attack behaviors by incorporating adaptive parameter adjustment and drift detection techniques. This problem is emphasized in the introduction as a fundamental drawback of traditional static IDS models.

The experimental results confirm that the AGB model consistently outperforms baseline classifiers such as Random Forest, XGBoost, and LightGBM, particularly in terms of Recall, F1-Score, and AUC-ROC, which are critical metrics for minimizing false negatives in cybersecurity detection tasks. More importantly, the model exhibits stable adaptation performance when subjected to concept drift scenarios, indicating its capacity to maintain predictive reliability under shifting data distributions. These findings reinforce the methodological premise that adaptive learning strategies enhance the resilience of intrusion detection systems in real-world network environments.

Despite these positive outcomes, several limitations remain. The adaptive mechanism, while effective, increases computational overhead during frequent drift events, which may reduce suitability for resource-constrained environments. Furthermore, although supervised learning was used for the majority of the assessment, further research is still needed to determine how well the model performs in semi-supervised or unsupervised circumstances with minimal labeled data. The dataset used, although comprehensive, may not fully represent emerging attack vectors that evolve beyond the patterns captured in CICIDS2017.

Future research should consider optimizing the computational efficiency of adaptive processes, integrating deep learning-based feature extraction, and extending evaluation to cross-domain or multi-source datasets to enhance generalization. Additionally, hybrid models combining AGB with real-time anomaly detection mechanisms could offer improved responsiveness to previously unseen threats. These directions open promising opportunities for advancing adaptive intrusion detection and strengthening its applicability in increasingly complex cybersecurity landscapes.

REFERENCES

Abbasi, M., A. Shahraki, and A. Taherkordi, (2021). Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey, *Comput Commun*, vol. 170, pp. 19–41, Mar. 2021, doi: 10.1016/J.COMCOM.2021.01.021.

Afuwape, A., A., Y. Xu, J. H. Anajemba, and G. Srivastava, (2021). Performance evaluation of secured network traffic classification using a machine learning approach, *Comput Stand Interfaces*, vol. 78, p. 103545, Oct. 2021, doi: 10.1016/J.CSI.2021.103545.

Aravamudhan Parthiban K., and T, (2023). A novel adaptive network intrusion detection system for internet of things, *PLoS One*, vol. 18, no. 4, pp. 1–29, Oct. 2023, doi: 10.1371/journal.pone.0283725.

Chen, T. and C. Guestrin, (2016). XGBoost: A Scalable Tree Boosting System, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, in KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

Chimphlee, W. and S. Chimphlee, (2024). Hyperparameters optimization XGBoost for network intrusion detection using CSE-CIC-IDS 2018 dataset," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 13, p. 817, Oct. 2024, doi: 10.11591/ijai.v13.i1.pp817-826.

Fan, Z., and Z. You, (2024). Research on network intrusion detection based on XGBoost algorithm and multiple machine learning algorithms, *Theoretical and Natural Science*, vol. 31, pp. 162–167, Oct. 2024, doi: 10.54254/2753-8818/31/20241171.

Genuario, F., G. Santoro, M. Giliberti, S. Bello, E. Zazzera, and D. Impedovo, (2024). Machine Learning-Based Methodologies for Cyber-Attacks and Network Traffic Monitoring: A Review and Insights, *Information*, vol. 15, no. 11, 2024, doi: 10.3390/info15110741.

Gowra B. and D. V, (2025). Adaptive Shields for Network Intrusion Detection via Gradient Boosting," *J Neonatal Surg*, vol. 14, no. 16S, pp. 456–468, Apr. 2025, doi: 10.63682/jns.v14i16S.4319.

Goodfellow, I., Y. Bengio, and A. Courville, (2016)*Deep Learning*. MIT Press, 2016.

Jannach, D., A. Manzoor, W. Cai, and L. Chen, (2021). A Survey on Conversational Recommender Systems, *ACM Comput. Surv.*, vol. 54, no. 5, May 2021, doi: 10.1145/3453154.

Lumumba, V., D. Sang, G. Njoka, D. Musyimi, and Kavita, (2024). Comparative Analysis of Cross-Validation Techniques: LOOCV, K-folds Cross-Validation, and Repeated K-folds Cross-Validation in Machine Learning Models," *American Journal of Theoretical and Applied Statistics*, vol. 13, pp. 127–137, Oct. 2024, doi: 10.11648/j.ajtas.20241305.13.

Meghdouri, F. (2023). Machine learning for network traffic analysis : Feature spaces and model optimization," Thesis - Dissertation, TU Wien, Vienna, 2023. doi: https://doi.org/10.34726/hss.2023.70923.

Moustafa, N., and J. Slay, (2015). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), in *Conference: Military Communications and Information Systems Conference (MilCIS)*, Canberra, Australia, Oct. 2015. doi: 10.1109/MilCIS.2015.7348942.

Nassreddine, G., M. Nassereddine, and O. Al-Khatib, (2025). Ensemble Learning for Network Intrusion Detection Based on Correlation and Embedded Feature Selection Techniques, *Computers*, vol. 14, no. 3, 2025, doi: 10.3390/computers14030082

Pemmada, S.K., K. Naidu, and D. K. K. Reddy, (2024). SMOTE Integrated Adaptive Boosting Framework for Network Intrusion Detection, Machine Learning for Cyber Physical System: Advances and Challenges, 2024, pp. 1–25. doi: 10.1007/978-3-031-54038-7_1.

Raschka, S. (2020). Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning, *ArXiv*, Nov. 2020, [Online]. Available: http://arxiv.org/abs/1811.12808

Reiber, L. (2025). Top Digital Forensics Trends 2025., Oxygen Forensics. [Online]. Available: https://www.oxygenforensics.com/en/resources/digital-forensics-trends-2025

Reunanen, J., (2003). Overfitting in Making Comparisons Between Variable Selection Methods, [Online]. Available: http://www.ics.uci.edu/

Rizvi, S., M. Scanlon, J. McGibney, and J. Sheppard, (2022). Application of Artificial Intelligence to Network Forensics: Survey, Challenges and Future Directions, *IEEE Access*, vol. 10, pp. 110362–110384, Oct. 2022, doi: 10.1109/ACCESS.2022.3214506.

Saied, M., S. Guirguis, and M. Madbouly, (2023). A Comparative Study of Using Boosting-Based Machine Learning Algorithms for IoT Network Intrusion Detection, *International Journal of Computational Intelligence Systems*, vol. 16, no. 1, p. 177, 2023, doi: 10.1007/s44196-023-00355-x.

Shahraki, A., M. Abbasi, and Ø. Haugen, (2020). Boosting algorithms for network intrusion detection: A comparative evaluation of Real AdaBoost, Gentle AdaBoost and Modest AdaBoost, *Eng Appl Artif Intell*, vol. 94, p. 103770, Sep. 2020, doi: 10.1016/J.ENGAPPAI.2020.103770.

Sharafaldin, I., A. H. Lashkari, and A. A. Ghorbani, (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization, in *International Conference on Information Systems Security and Privacy*, 2018. doi: https://doi.org/10.5220/0006639801080116.

Sridevi, P., P., A. Kamboj, V. Shete, and R. Harikrishnan, (2022). Machine Learning Based Network Traffic Predictive Analysis, *Review of Computer Engineering Research*, vol. 9, no. 2, pp. 96–108, May 2022, doi: 10.18488/76.v9i2.3065.