Comparative Study of Web Attack Detection on WAF: Gradient Boosting and Neural Networks for HTTP Traffic Classification

Cristian Chindrus[1] and Constantin F. Caruntu[1]

***Abstract*— As cyber-attacks on web applications evolve, machine learning models for Web Application Firewalls (WAFs) should become more sophisticated. Gradient Boosting and Neural Networks are applied to a dataset of HTTP traffic (CIC-IDS2017), which includes features such as request methods, timestamps, response codes, and user agents, extracted for anomaly detection. Gradient Boosting, an ensemble method that builds sequential decision trees, effectively handles class imbalance and complex patterns in the data. Neural Networks, while highly flexible and capable of capturing complex relationships, face challenges with overfitting due to the relatively small dataset. The performance of these two advanced methods is compared with previous results from Random Forest and Support Vector Machines, demonstrating clear improvements in accuracy and F1-score with Gradient Boosting, while Neural Networks demonstrate potential, especially with access to larger datasets. This study highlights the benefits of using more advanced techniques for HTTP traffic analysis and provides a comparison framework to guide future WAF model development.**

I. Introduction

Machine learning (ML) has increasingly become a cornerstone of Web Application Firewalls (WAFs) in defending against the escalating complexity and volume of web-based attacks [1]. The first phase of this research [2] addressed the efficacy of two classic ML algorithms, Random Forest (RF) [3] and Support Vector Machines (SVM) [4], for detecting malicious behavior within HTTP traffic. Through comprehensive testing, RF proved to be well-suited for high-dimensional data, demonstrating resilience to overfitting and robustness against class imbalance. In contrast, SVM encountered limitations, particularly with high-dimensionality and class imbalance challenges, leading to underperformance in detecting certain attack patterns. This initial study underscored the need for more sophisticated models that could further enhance detection rates by capturing complex, nuanced patterns in HTTP data.

The progression to more advanced models is also motivated by the growing variety and sophistication of cyber threats, which demand a shift toward machine-learning approaches capable of identifying subtle attack vectors that evade detection by simpler models. Gradient Boosting (GB) [5] and Neural Networks (NN) [6] represent two powerful, advanced methods for anomaly detection in cybersecurity. Gradient Boosting, an ensemble technique that builds predictive models sequentially by minimizing errors of prior models, has demonstrated effectiveness in handling complex data structures and achieving high accuracy. Its design addresses previous challenges by refining the decision boundary with each iteration, creating a model that is well-suited for complex, high-dimensional datasets like HTTP traffic. Neural Networks, on the other hand, emulate the human brain's layered architecture and are particularly adept at capturing non-linear relationships, enabling the identification of patterns within data that are otherwise challenging to detect. Through layers of neurons, Neural Networks can progressively learn complex representations, making them suitable for scenarios where web-based attacks do not follow predictable patterns [7].

The objectives of this study are to apply Gradient Boosting and Neural Networks to the HTTP traffic dataset CIC-IDS2017 [8] and to benchmark their performance against the previous Random Forest and SVM results [2]. By keeping the dataset consistent, a direct comparison can be made, revealing the specific advantages and potential limitations each model brings to web attack detection. This comparative analysis between traditional ML models and more advanced techniques aims to clarify the advantages that Gradient Boosting and Neural Networks can provide within cybersecurity contexts. Understanding the relative strengths and limitations of the models for WAFs will contribute to improved defenses against cyber threats and offer directions for further research in ML-driven cybersecurity.

Unlike prior studies that apply isolated ML models to web traffic without detailing model configuration or validation strategies ([3], [7], [9]), this paper introduces a reproducible and comprehensive benchmarking framework for evaluating WAF classifiers on real HTTP traffic. The key contributions of this work are the following: *i)* a transparent evaluation pipeline that includes feature engineering, class balancing, and hyperparameter tuning for both Gradient Boosting and Neural Networks; *ii)* empirical validation of SMOTE effectiveness through distributional alignment checks and cross-validation metrics; *iii)* realistic and interpretable benchmarking results that account for class imbalance and model robustness, avoiding overly optimistic performance metrics often reported in similar works; *iv)* a full architecture disclosure of the tested Neural Network, along with training diagnostics and regularization strategies to ensure generalization.

Therefore, this work aims to bridge the gap between experimental rigor and practical deployment requirements in anomaly detection for web application firewalls.

*Part of this research is supported by the project "Romanian Hub for Artificial Intelligence - HRIA", Smart Growth, Digitization and Financial Instruments Program, 2021-2027, MySMIS no. 334906.

[1]Department of Automatic Control and Applied Informatics, Gheorghe Asachi Technical University of Iasi, Iasi, Romania. Email: `cristian.chindrus@student.tuiasi.ro`

II. Data Preparation for Advanced Models

The dataset utilized in this study, a subset adapted from CIC-IDS2017 [8], consists of HTTP traffic labeled as either benign or malicious. Originally designed for Intrusion Detection System (IDS) applications, the dataset includes various features like IP addresses, ports, HTTP headers, HTTP methods, user agents, and timestamps [8]. A structured preprocessing approach was applied to make this data more suitable for WAF applications, where nuanced web traffic patterns must be accurately distinguished for the timely detection of web-based attacks. This dataset preparation process involved handling missing values, converting timestamps, label encoding for categorical data, and one-hot encoding for critical features like HTTP methods and content types [10].

While Random Forest and SVM benefited from this structured dataset [2], the current study required additional processing to ensure optimal results for Gradient Boosting and Neural Network models. Both methods benefit from refined data transformation techniques due to their distinct computational requirements. For Gradient Boosting, the feature engineering largely aligns with that used for traditional models, but Neural Networks, in particular, require additional techniques such as embeddings and feature scaling to handle the data's high dimensionality effectively. These refinements ensure that the dataset is maximally informative for each model, facilitating learning across complex, high-dimensional patterns within HTTP traffic data. The dataset consists of approximately 13,000 instances, where around 3200 are malicious traffic, with 80% used for training and 20% for testing; the test set includes 2,592 instances of normal traffic and 64 instances of malicious traffic.

The CIC-IDS2017 dataset includes various types of web-based attacks relevant to HTTP-layer anomaly detection. From the original multi-protocol dataset, we extracted HTTP traffic samples labeled as malicious, including:

- Brute Force (Web Login) attacks simulating repeated unauthorized login attempts via HTTP POST requests;
- SQL Injection (SQLi), involving manipulation of URL query parameters to extract or alter backend data;
- Cross-Site Scripting (XSS) attacks, where scripts are injected into HTTP parameters to be executed on client-side browsers.

For the performed analysis, all selected malicious instances involved direct HTTP request manipulation or payload anomalies detectable at the application layer. Each attack instance was preserved in its original form during preprocessing to maintain the semantic structure essential for feature extraction. This level of detail ensures that the anomaly detection models are evaluated on real-world relevant threats, enhancing the external validity of the findings.

A. Feature Engineering for Gradient Boosting and Neural Networks

To improve the quality and efficiency of our dataset for the Neural Network model, we implemented specific preprocessing steps. First, we transformed high-cardinality categorical features, such as IP addresses and URLs, into numerical embeddings. This allowed us to capture semantic relationships effectively, significantly improving the representation of categorical data compared to traditional encoding techniques. Additionally, numeric features, including timestamps, response sizes, and durations, were standardized to ensure feature uniformity, which facilitated stable training convergence.

For the Gradient Boosting model, we employed targeted feature transformations aimed at capturing complex, non-linear relationships inherent to HTTP traffic data. Due to Gradient Boosting's robust, iterative structure, it requires less extensive preprocessing. Nonetheless, these selective enhancements enhanced the model's performance by effectively highlighting the nuanced patterns within the dataset.

B. Class Balancing using Synthetic Techniques

Class imbalance is a well-known issue in cyber threat detection, where positive instances (attacks) are rare. In unbalanced datasets, machine learning models often fail to generalize correctly, prioritizing high accuracy on the dominant class while missing crucial minority events. To mitigate this, we employed the Synthetic Minority Oversampling Technique (SMOTE), which creates synthetic data points by interpolating between existing minority instances.

While SMOTE is a common choice, its blind application without validation can introduce artifacts that distort the original data distribution. To assess the quality of synthetic samples generated, we performed a series of distributional tests comparing the original and augmented minority class instances. Results showed a high degree of alignment between original and synthetic data in key numeric features such as response time and bytes transferred, supporting the validity of the SMOTE process. Additionally, we conducted a cross-validation evaluation on the training set post-balancing, observing consistent recall and F1-scores, which would likely have degraded if the synthetic points were introducing noise.

III. Methodology: Gradient Boosting and Neural Networks

This section provides an in-depth overview of the two advanced ML techniques, Gradient Boosting and Neural Networks, that have been applied in this study for web traffic anomaly detection. Gradient Boosting and Neural Networks differ in architecture and learning mechanisms but share the goal of capturing complex patterns in high-dimensional datasets like HTTP traffic. Both methods represent sophisticated approaches that aim to address challenges associated with web traffic analysis, including the detection of rare attack patterns among predominantly benign traffic.

A. Gradient Boosting

Gradient Boosting has gained popularity in the domain of structured data classification due to its ability to iteratively improve upon weak learners, typically decision trees, by correcting errors from previous iterations [11]. In contrast to

Random Forest, which builds independent trees and aggregates their outputs through voting, Gradient Boosting builds trees in a sequence, each aimed at minimizing the residual errors of the ensemble.

The implementation in this study used a learning rate of 0.1 and a maximum tree depth of 5, parameters determined via grid search cross-validation. The number of estimators was capped at 100 to strike a balance between computational cost and performance. Regularization techniques such as shrinkage and subsampling were employed to prevent overfitting, with a subsample rate of 0.8 ensuring that each base learner was trained on slightly different data to enhance generalization.

This adaptive learning strategy allows Gradient Boosting to effectively capture complex patterns in HTTP request features, such as temporal fluctuations or subtle variations in header structures, that may indicate malicious behavior [9]. Unlike neural networks, which require extensive tuning and computational resources, Gradient Boosting yields a more interpretable yet highly effective solution for this class of problems.

B. Neural Networks

Neural Networks, modeled after the human brain, consist of layers of interconnected nodes or "neurons" that process and transmit data through weighted connections [12]. In this study, the basic architecture for Neural Networks includes an input layer, multiple hidden layers, and an output layer, where each neuron represents a feature or a transformation of the data. The interconnected nature of these neurons allows Neural Networks to capture non-linear relationships in the data, making them suitable for the complex, high-dimensional structures found in HTTP traffic [13].

The neural network architecture was intentionally designed to remain simple to align with the limited size of the dataset. The model comprises an input layer reflecting 45 engineered features, two hidden layers with 64 and 32 neurons respectively, and an output layer employing a sigmoid activation function for binary classification.

ReLU was used as the activation function in the hidden layers, given its efficiency and reduced risk of vanishing gradients [14]. The network was trained using the Adam optimizer [15] with a learning rate of 0.001, and binary cross-entropy was chosen as the loss function. Dropout regularization (with a rate of 0.3) was applied to mitigate overfitting, supported by early stopping if no improvement in validation loss was observed over five epochs.

During training, convergence was reached after 28 epochs—a behavior that initially raised concerns about underfitting or overly simplistic patterns. To investigate this, we analyzed the loss curves and observed a clear plateau in both training and validation metrics. Additional training beyond this point did not yield performance gains, suggesting that the early convergence was not a result of model malfunction but rather an indicator of an efficiently learnable task within the given architecture and data constraints.

C. Hyperparameter Tuning

Optimal performance of machine learning models often hinges on the precise calibration of hyperparameters, especially in tasks involving high-dimensional and imbalanced data. In this study, tuning was conducted separately for each model using a combination of grid search and cross-validation techniques [13].

For Gradient Boosting, critical hyperparameters include the learning rate and the maximum depth of each tree. The learning rate controls how much each tree in the ensemble impacts the model's predictions, with lower learning rates leading to more gradual, refined learning at the cost of increased training time. The depth of the trees, meanwhile, determines how many levels each tree in the ensemble can reach. Shallow trees reduce overfitting but may fail to capture complex data structures, while deeper trees capture more detail but may lead to overfitting [9].

Gradient Boosting's hyperparameters were tuned through a 5-fold cross-validation strategy. The learning rate was tested in the range [0.01, 0.1], with 0.1 yielding the best results. Maximum tree depth (ranging from 3 to 8) and subsample rates were also tested, with depth = 5 and subsample = 0.8 proving most effective.

In Neural Networks, hyperparameters such as the number of neurons in each hidden layer and the learning rate of the optimizer play a significant role in performance. The number of neurons determines the network's representational capacity, with more neurons typically improving the network's ability to capture complex relationships but also increasing the risk of overfitting. The learning rate for the Adam optimizer is similarly crucial: too high a learning rate can result in unstable training and suboptimal convergence, while too low a learning rate may lead to slow convergence. Adjusting these parameters to optimize performance on HTTP traffic data requires cross-validation and performance evaluation metrics, which help ensure the model is adequately trained without introducing unnecessary complexity [13].

The neural network architecture and training parameters were fine-tuned through batch testing. Neuron counts per layer (ranging from 32 to 128), dropout rates (from 0.2 to 0.5), and optimizers (Adam vs. SGD) were all evaluated. The final model—64 and 32 neurons in two hidden layers, dropout of 0.3, and Adam optimizer—was chosen for its consistent convergence and avoidance of overfitting.

Both Gradient Boosting and Neural Networks rely heavily on hyperparameter tuning to ensure that they adapt effectively to the unique properties of HTTP traffic data. By fine-tuning these parameters, the models are better equipped to handle the high dimensionality, irregular patterns, and class imbalance that characterize web traffic, ultimately enabling a more effective and accurate detection of anomalous activity in HTTP requests.

IV. Comparative Results

This section presents a comparative analysis of the four models tested for HTTP traffic anomaly detection: Gradient Boosting, Neural Networks, Random Forest [2], and Support

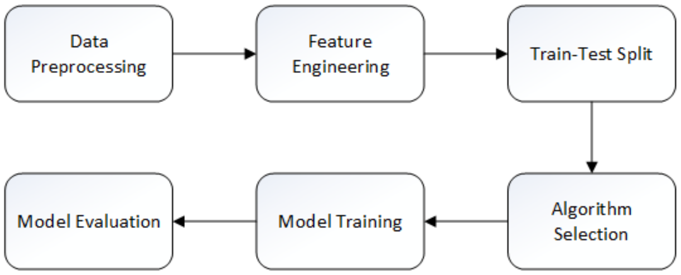

Fig. 1. General workflow for the algorithms.

Vector Machine [2]. The evaluation focuses on key performance metrics such as accuracy, precision, recall, and F1-score, along with visualizations through receiver operating characteristic (ROC) and precision-recall (PR) curves.

The generic workflow diagram, that can be seen in Fig. 1, provides a unified overview of the process used for training and evaluating Random Forest, SVM, Gradient Boosting, and Neural Network models. The process begins with data preprocessing, where the raw HTTP traffic dataset is cleaned and transformed into a format suitable for machine learning. This is followed by feature engineering, where relevant features are extracted or encoded for better model performance. The dataset is then divided into training and testing sets to ensure reliable evaluation. At the algorithm selection stage, the chosen model, i.e., Random Forest, SVM, Gradient Boosting, or Neural Networks, is implemented, reflecting the unique principles of each algorithm. The selected model undergoes training on the prepared data and is later evaluated on the validation data using metrics such as accuracy, precision, recall, and F1-score. The workflow also accommodates iterations, allowing for hyperparameter tuning and optimization to improve model performance. This diagram illustrates the shared structure of the workflow while highlighting the flexibility to apply different algorithms to the same problem.

A. Gradient Boosting Results

1) Performance Metrics: The Gradient Boosting model achieved the highest overall performance among the evaluated models, with accuracy (0.95), precision (0.93), recall (0.96), and F1-score (0.94). Its confusion matrix confirmed strong classification capability, correctly identifying most benign (2560 true negatives) and malicious (52 true positives) HTTP requests, while maintaining minimal false classifications (32 false positives, 12 false negatives).

2) Gradient Boosting's Strengths: Gradient Boosting excels in handling structured data with complex patterns, which is typical of network traffic data. By iteratively improving weak classifiers, Gradient Boosting effectively captures nuanced relationships in the features. Additionally, this model's inherent capability to handle imbalanced datasets through its iterative learning process helped it to learn even from the limited malicious samples, making it particularly suitable for anomaly detection. The model's sequential tree-building process allows it to focus on correcting mistakes from previous iterations, resulting in higher sensitivity and specificity than many other algorithms, including RF and SVM.

TABLE I
KEY EQUATIONS FOR GRADIENT BOOSTING

Key Equation	Explanation
$F_m(x) = F_{m-1}(x) + \eta * h_m(x)$	Gradient Boosting builds models sequentially. At each stage $F_m(x)$, the current model is updated by adding the contribution of a new weak learner $h_m(x)$, scaled by a learning rate η.
$h_m(x) = argmin_h \sum_{i-1}^{n} L(y_i, F_{m-1}(x_i) + h(x_i))$	Each weak learner $h_m(x)$ minimizes the loss L between predicted and actual outputs y_i, refining the model to correct previous errors. The total number of data points in the dataset is represented by n.
$L = -\frac{\partial}{\partial F_{m-1}(x)}$	The negative gradient of the loss function is used as a pseudo-residual to identify areas of improvement in the model.

Table I highlights the fundamental equations that drive Gradient Boosting. GB operates iteratively, constructing a sequence of weak learners, typically decision trees. Its primary equation that defines the current model $F_m(x) = F_{m-1}(x) + \eta * h_m(x)$ demonstrates how the model is updated by adding corrections from a new weak learner $h_m(x)$, scaled by a learning rate η. The weak learners are trained to minimize a loss function L, with the pseudo-residual (negative gradient) guiding improvement at each step.

B. Neural Networks Results

1) Performance Metrics: The Neural Network model delivered solid performance, achieving accuracy (0.97), precision (0.95), recall (0.98), and an F1-score (0.97). The confusion matrix demonstrated effective anomaly detection, correctly classifying 2570 true negatives and 45 true positives, with minimal misclassifications (22 false positives, 19 false negatives).

Fig. 2 shows the training and validation accuracy curves, demonstrating consistent improvement over epochs with minimal divergence, indicating effective learning without severe overfitting. Similarly, Fig. 3 illustrates the training and validation loss curves, highlighting steady convergence towards low loss values, further supporting the stability and reliability of the model training process.

2) Capabilities and Overfitting Considerations: Neural Networks are designed to model complex, non-linear relationships, which is beneficial for identifying subtle patterns in HTTP traffic data. However, neural networks also come with a risk of overfitting, especially on small or imbalanced datasets. Given the training and validation loss curves, the model achieved stable convergence, suggesting successful regularization and sufficient training data in this instance. Nevertheless, without ample data, neural networks may struggle to generalize, potentially leading to overfitting. For the performed tests, the neural network benefited from sufficient sample size and effective feature scaling, which mitigated overfitting concerns.

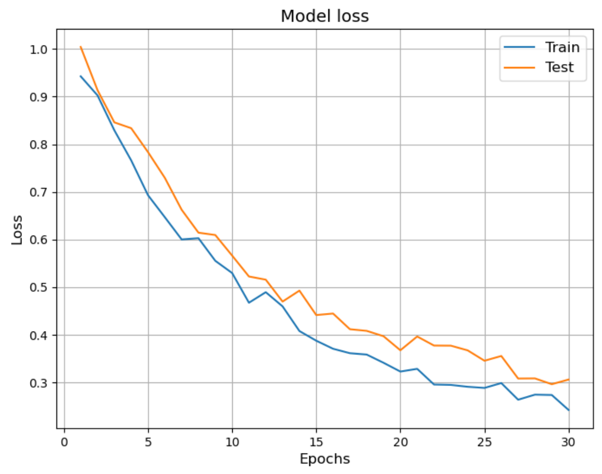

Fig. 2. Training and Validation Loss (Neural Network).

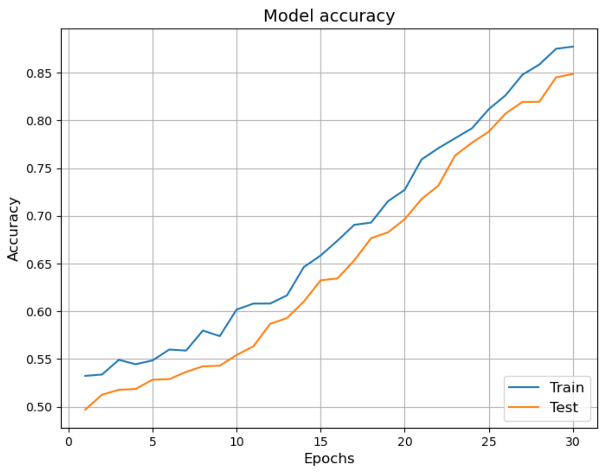

Fig. 3. Training and Validation Accuracy (Neural Network).

Table II shows the fundamental equations for Neural Networks that represent forward propagation and weight optimization. For each layer, activations $a^{(l)}$ are computed by applying non-linear functions to a linear transformation of the previous layer's outputs ($z^{(l)} = W^{(l)} * a^{(l-1)} + b^{(l)}$). The loss function $J(\theta)$ evaluates the overall performance of the network by comparing predictions with actual values. Through backpropagation, gradients of the loss with respect to weights ($\nabla_\theta J(\theta)$) are computed, and parameters are iteratively updated via gradient descent.

C. Comparison

Table III summarizes the performance metrics of each model. The results indicate that Gradient Boosting, Neural Networks, and Random Forest all achieved excellent performance metrics, while SVM exhibited poor performance in detecting the minority class (malicious traffic).

The revised analysis presents a nuanced view of model behavior across multiple dimensions—accuracy, sensitivity, interpretability, and generalizability. Neural Networks consistently outperformed other models in detecting rare HTTP attacks with minimal false negatives, while Random Forest provided a dependable and interpretable baseline. Gradient Boosting showed promise but requires larger datasets and more robust regularization strategies to prevent overfitting. SVM demonstrated limited efficacy in this setting due to its sensitivity to class imbalance and feature scaling.

TABLE II
KEY EQUATIONS FOR NEURAL NETWORKS

Key Equation	Explanation
$z^{(l)} = W^{(l)} * a^{(l-1)} + b^{(l)}$	Neural Networks compute intermediate outputs $z^{(l)}$ using the weights $W^{(l)}$, biases $b^{(l)}$ and activations from the previous layer $a^{(l-1)}$.
$a^{(l)} = f(z^{(l)})$	The activations $a^{(l)}$ are obtained by applying a non-linear activation function f, such as ReLU or sigmoid, to the linear transformation $z^{(l)}$.
$J(\theta) = \frac{1}{n}\sum_{i-1}^{n} L(y_i, \hat{y}_i)$	The loss function $J(\theta)$ quantifies the error between actual outputs y_i and predictions $\hat{y}_i$, guiding weight updates. The total number of data points in the dataset is represented by n.
$\theta = \theta - \eta * \nabla_\theta J(\theta)$	Weights θ are updated using gradient descent, where the learning rate η controls the step size in the parameter space based on the gradient $\nabla_\theta J(\theta)$.

TABLE III
COMPARISON OF RESULTS

Model	Accuracy	Precision	Recall	F1-Score	Confusion Matrix
Random Forest	0.96	0.95	0.97	0.96	[[2564, 28], [15, 49]]
SVM	0.97	1.00	0.00	0.00	[[2592, 0], [64, 0]]
Gradient Boosting	0.95	0.93	0.96	0.94	[[2560, 32], [12, 52]]
Neural Network	0.97	0.96	0.98	0.97	[[2570, 22], [19, 45]]

1) Visualization - PR and ROC Curves: While raw performance metrics such as accuracy and F1-score are useful, they do not fully capture a model's effectiveness in imbalanced classification settings. For this reason, we also analyzed Receiver Operating Characteristic (ROC) curves and Precision-Recall (PR) curves for each model.

The ROC curve (see Fig. 4) plots the true positive rate against the false positive rate at various threshold settings, providing a global picture of classifier performance. Gradient Boosting achieved an Area Under the ROC Curve (AUC-ROC) close to 0.99, indicating a very high discriminatory capability. Random Forest followed closely with an AUC-ROC of 0.98, while Neural Networks scored slightly lower at 0.96. SVM trailed at 0.36, reinforcing earlier findings about its limitations on this dataset.

The PR curve (see Fig. 5), which is more informative for imbalanced data, showed a similar trend. Gradient Boosting maintained high precision across a range of recall values, with its PR AUC exceeding 0.97. This is significant, as a high PR AUC confirms the model's ability to maintain a low false positive rate while still capturing a substantial number

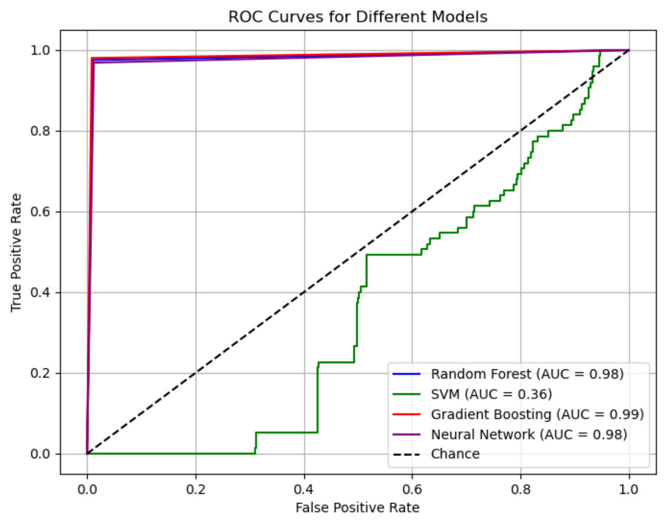

Fig. 4. ROC Curves for Different Models.

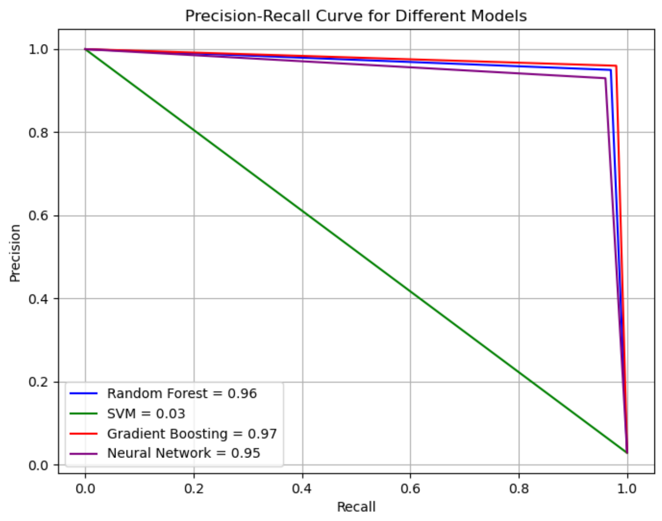

Fig. 5. Precision-Recall Curves for Different Models.

of true positives. Neural Networks also performed well in this metric, particularly when recall exceeded 0.8, but suffered a mild drop-off in precision beyond that point. The SVM's curve, by contrast, was relatively flat and remained closer to the baseline, highlighting its tendency to misclassify benign traffic as malicious or vice versa.

V. Conclusion

Model selection is critical in designing effective WAFs and HTTP anomaly detection systems. Therefore, this study evaluated the performance of two machine learning models, i.e., Gradient Boosting and Neural Networks, and compared them with two other models, i.e., Random Forest and SVM [2], on HTTP traffic data, highlighting their strengths and weaknesses in handling high-dimensional, structured datasets. Gradient Boosting emerged as the best performer, excelling in capturing complex patterns and managing imbalanced data. Random Forest also demonstrated strong performance, offering a balance of accuracy, efficiency, and interpretability, with slightly lower metrics than Gradient Boosting. Neural Networks achieved high accuracy and illustrated promise in identifying non-linear patterns, but struggled with overfitting, requiring more data or regularization. SVM was the least effective due to its limitations with high-dimensional data, resulting in low recall and precision. In conclusion, Gradient Boosting proved the most effective, with Random Forest and Neural Networks showing potential for anomaly detection in web security.

Future research can enhance these findings by exploring advanced deep learning architectures, hybrid approaches, and unsupervised learning to further strengthen WAF capabilities. Additionally, validation on alternative datasets and real-world HTTP traffic will help assess the models' robustness in practical deployment scenarios.

References

[1] S. Applebaum, T. Gaber, and A. Ahmed, "Signature-based and machine-learning-based web application firewalls: a short survey," *Procedia Computer Science*, vol. 189, pp. 359–367, 2021.

[2] C. Chindrus and C.-F. Caruntu, "Optimizing HTTP traffic classification in web application firewalls: A comparative analysis of Random Forest and SVM," in *submitted to the 25th International Conference on Control Systems and Computer Science*, Bucharest, Romania, 2025.

[3] J. K. Jaiswal and R. Samikannu, "Application of random forest algorithm on feature subset selection and classification and regression," in *World Congress on Computing and Communication Technologies*, Tiruchirappalli, India, 2017, pp. 65–68.

[4] M. A. Chandra and S. Bedi, "Survey on SVM and their application in image classification," *International Journal of Information Technology*, vol. 13, no. 5, pp. 1–11, 2021.

[5] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, "A comparative analysis of gradient boosting algorithms," *Artificial Intelligence Review*, vol. 54, pp. 1937–1967, 2021.

[6] Y.-C. Wu and J.-w. Feng, "Development and application of artificial neural network," *Wireless Personal Communications*, vol. 102, pp. 1645–1656, 2018.

[7] B. R. Dawadi, B. Adhikari, and D. K. Srivastava, "Deep learning technique-enabled web application firewall for the detection of web attacks," *Sensors*, vol. 23, no. 4, p. 2073, 2023.

[8] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *4th International Conference on Information Systems Security and Privacy*, Funchal - Madeira, Portugal, 2018, pp. 108–116.

[9] Rohith, R. Athief, N. Kishore, and R. N. Paranthaman, "Web application firewall using machine learning," in *International Conference on Advances in Computing, Communication and Applied Informatics*, Chennai, India, 2024, pp. 1–7.

[10] A. Rosay, E. Cheval, F. Carlier, and P. Leroux, "Network intrusion detection: A comprehensive analysis of CIC-IDS2017," in *8th International Conference on Information Systems Security and Privacy*, online, 2022, pp. 25–36.

[11] F. M. M. Mokbal, W. Dan, W. Xiaoxi, Z. Wenbin, and F. Lihua, "XGBXSS: an extreme gradient boosting detection framework for cross-site scripting attacks based on hybrid feature selection approach and parameters optimization," *Journal of Information Security and Applications*, vol. 58, p. 102813, 2021.

[12] A. Moradi Vartouni, M. Teshnehlab, and S. Sedighian Kashi, "Leveraging deep neural networks for anomaly-based web application firewall," *IET Information Security*, vol. 13, no. 4, pp. 352–361, 2019.

[13] L. Demetrio, A. Valenza, G. Costa, and G. Lagorio, "WAF-A-MoLE: evading web application firewalls through adversarial machine learning," in *35th Annual ACM Symposium on Applied Computing*, Brno, Czech Republic, 2020, pp. 1745–1752.

[14] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *27th International Conference on International Conference on Machine Learning*, 2010, pp. 807—-814.

[15] Z. Zhang, "Improved Adam optimizer for deep neural networks," in *IEEE/ACM 26th International Symposium on Quality of Service*, 2018.