

THREAT DETECTION IN FIREWALL LOGS USING FEATURE SELECTION AND ENSEMBLE LEARNING

NOORDAMA¹, BENFANO SOEWITO²

^{1,2}Department of Computer Science, BINUS Graduate Program-Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia
E-mail: ¹noordama@binus.ac.id, ²bsoewito@binus.edu

ABSTRACT

The firewall serves as the first line of defense in network security, responsible for monitoring and filtering data traffic. However, firewalls remain limited in accurately detecting threats, often generating false positives and false negatives, where legitimate traffic is misclassified as attacks or vice versa. Despite numerous studies that apply machine learning to firewall log analysis, the challenge of accurately distinguishing between benign and malicious activities remains. Existing methods often struggle with high rates of false positives and false negatives, particularly in complex, real-world network environments. This study aims to bridge this gap by combining feature selection and stacking-based ensemble learning, which has been underexplored in addressing the issue of misclassification in firewall logs. Our feature selection approach combines Random Forest feature importance with mutual information, while the ensemble framework employs a stacking strategy that integrates Iterative Dichotomiser 3 (ID3), Random Forest (RF), and Extreme Gradient Boosting (XGBoost), with XGBoost serving as the meta-learner. Experiments were conducted on two datasets: Palo Alto firewall logs from an Indonesian government agency and a public dataset from the UCI Machine Learning Repository, to evaluate the model's generalization performance. The results demonstrate that the ensemble learning method reduces false positives and false negatives, achieving 99.97 % accuracy on the primary dataset and 99.87 % on the secondary dataset, representing a 7.14 % reduction in misclassification compared with standalone machine learning algorithms. This work contributes to the development of a robust and efficient threat detection system suitable for deployment in both governmental and public network environments.

Keywords: *Threat Detection, Feature Selection, Ensemble Learning, Random Forest, Network Security.*

1. INTRODUCTION

Firewalls serve as the primary defense in network security, inspecting bidirectional traffic for potential threats. Nevertheless, they frequently suffer from high rates of false positives and false negatives, leading to legitimate activities being flagged as threats or genuine attacks going unnoticed. To address these deficiencies, this study integrates feature selection with a stacking-based ensemble framework to boost detection accuracy and reduce misclassification.

In Indonesia, the rapid expansion of internet use has been paralleled by escalating cyber threats, particularly within government networks where routine file downloads and web browsing heighten the risk of malware infiltration. As reported by the National Cyber and Crypto Agency (BSSN), firewalls record all network transactions in log files, which capture essential traffic details [1],[2]. However, despite this logging capability, firewalls remain vulnerable to misclassification errors false

positives can trigger unnecessary security responses, while false negatives permit real attacks to bypass defenses [3]. Prior work has explored a range of techniques to mitigate these errors in firewall log analysis. Machine learning methods, in particular, have demonstrated effectiveness at detecting intricate patterns that traditional signature or rule-based systems overlook. Yet, only a handful of studies have combined feature-selection mechanisms with stacking-based ensemble models to jointly optimize accuracy and error reduction, revealing a clear gap in the literature.

Although many studies focus on improving firewall log classification accuracy, few address the reduction of false positives and negatives through integrated feature selection and ensemble learning. The gap lies in the integration of these two approaches, which can significantly enhance both the accuracy and efficiency of detection systems. This study contributes by integrating feature importance through Random Forest and Mutual

Information with a stacking ensemble model to address these challenges, we propose a stacking ensemble that unites three algorithms: Iterative Dichotomiser 3 (ID3), Random Forest, and Extreme Gradient Boosting (XGBoost). ID3 is selected for its rapid decision-tree construction based on information gain, making it suitable for real-time log classification [4]. Random Forest offers robustness against overfitting and consistency on complex, high-dimensional data [5],[6], while XGBoost excels at iteratively correcting mispredictions, especially in imbalanced datasets [7], XGBoost can be integrated with Explainable AI (XAI) techniques to explain the results of analyses in cloud service logs.

Recent advances have demonstrated the effectiveness of deep learning for real-time log analysis. Al Jallad et al. [8] deployed a Long Short-Term Memory (LSTM) network to process streaming firewall logs, achieving reduced false positives by up to 10 % compared to conventional SVM approaches.

2. RELATED WORKS

Beyond deep learning, traditional machine learning algorithms have also been leveraged to uncover intricate patterns in firewall log data. Ertam and Kaya [9] evaluated several SVM kernels linear, polynomial, sigmoid, and RBF and found that the RBF kernel achieved the highest F1 score, while the sigmoid kernel delivered superior recall. Their work underscores the importance of kernel selection for balancing detection performance metrics.

Unsupervised methods offer an alternative by identifying anomalies without labeled data. Allagi and Rachh [10] applied K-Means clustering and Self-Organizing Feature Maps (SOFM) to firewall logs from the UCI Repository, achieving 97.2 % accuracy in distinguishing normal from abnormal traffic. This result highlights the promise of clustering techniques for large-scale anomaly detection.

Several studies have compared multiple supervised classifiers on the same Firat University log dataset (65 532 entries, 11 features). Sharma et al. [11] tested K-Nearest Neighbors, Logistic Regression, SVM, Decision Tree, and SGD, then proposed a heterogeneous stacking ensemble with Random Forest as meta-learner, which achieved 99.8 % accuracy and 91 % precision exceeding all individual models. Al-Behadili [12], using a C4.5 Decision Tree with 10-fold cross-validation, reported 99.83 % accuracy, a Kappa of 0.9972, and F-Measure of 0.998, outperforming six alternative

classifiers including SVM, ANN, and PSO. These findings indicate that while a well tuned single model can deliver near state of the art performance, ensemble methods can yield incremental gains in precision and reliability.

Balancing classification errors is critical. Dalvi et al. [13] emphasized the need to calibrate prediction thresholds in intrusion detection models such as Random Forest, Logistic Regression, and Neural Networks to manage class imbalances and achieve a balanced reduction in false positives and false negatives.

Ensemble strategies themselves have been extensively studied. Mienye and Sun [14] compared bagging, boosting, and stacking across algorithms like Random Forest, AdaBoost, Gradient Boosting, XGBoost, LightGBM, and CatBoost, finding that boosting and stacking consistently improved resilience against diverse attack types.

Integrating feature selection with adaptive classification further enhances detection. Ling and Hao [15] combined Normalized Mutual Information Feature Selection (NMIFS) with a Parallel Adaptive Quantum Genetic Algorithm (MOP-AQGA), which on NSL-KDD and CICIDS2017 datasets achieved high accuracy and low false-negative rates, particularly on smaller samples. Similarly, Kasongo and Sun [16] applied XGBoost-based feature selection on UNSW-NB15 logs and demonstrated that Decision Trees and ANNs reached 90.85 % and 84.39 % accuracy, respectively, while notably reducing false positives.

Finally, Komadina et al. [17] explored synthetic anomaly injection in power-grid firewall logs, comparing supervised and unsupervised models. Their work confirmed that supervised approaches remain more effective for detecting stealthy threats and uniquely evaluated resource usage and stability across both proprietary and public datasets.

Overall, these studies illustrate the effectiveness of machine learning and ensemble techniques in improving firewall log based threat detection. They establish a foundation for our proposed integration of targeted feature selection with stacking ensembles to deliver a system that is not only accurate but also computationally efficient and adaptable to real-world network conditions.

3. METHODOLOGY

3.1. Overview

This study was conducted systematically, starting from data collection, preprocessing, and model development using an ensemble learning approach, followed by performance evaluation. The

overall research design is illustrated in Figure. 1.

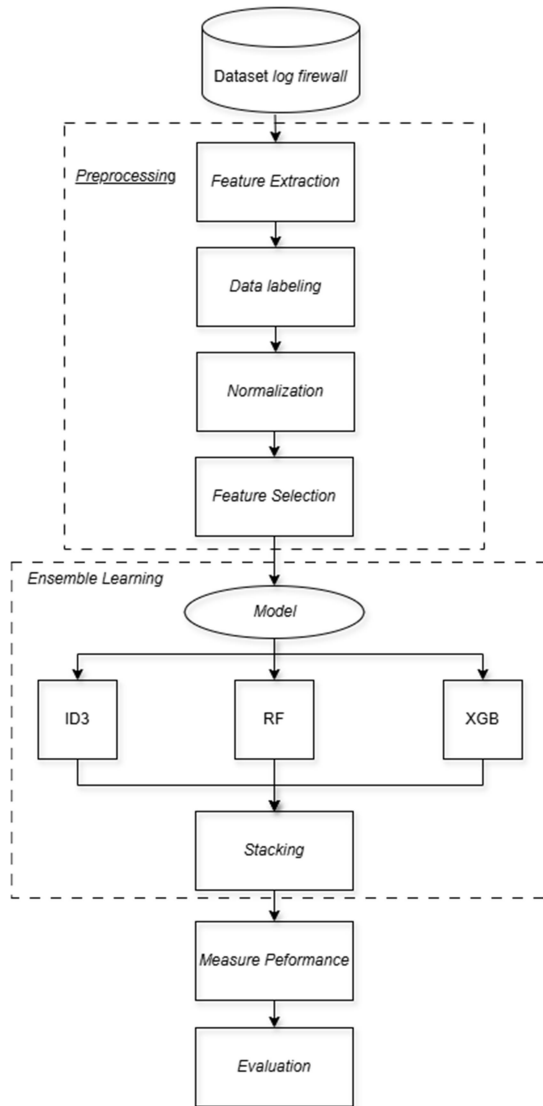


Figure. 1: Research design for threat detection based on firewall logs

As shown in the Figure. 1, The research design outlines the end-to-end process of threat detection based on firewall log analysis. The process begins with the collection of firewall log datasets, which are then subjected to a feature extraction phase to convert categorical data into numerical formats suitable for model processing.

Subsequently, data labeling and normalization are performed to ensure consistent scaling across features. Feature selection is carried out using two methods: Mutual Information (MI) and Random Forest-based Feature Importance (FI-RF). These methods are applied to identify and retain the most relevant features for the classification task.

The preprocessed data is used to train three machine learning models: Random Forest (RF), Iterative Dichotomiser 3 (ID3), and XGBoost (XGB). The predictions generated by these base models are then combined using a stacking ensemble approach, with XGBoost serving as the meta-learner to produce the final classification output.

The final stage involves evaluating the model’s performance through a set of classification metrics, followed by a comprehensive assessment to determine the overall effectiveness of the threat detection system.

3.2. Dataset

This study utilizes two sources of datasets: a primary dataset and a secondary dataset. Each dataset serves a specific role in testing and evaluating the performance of the classification model for detecting network threats based on firewall log analysis.

3.2.1. Primary dataset

The primary dataset used in this study is sourced from the firewall logs of a Palo Alto 5220 device deployed within an Indonesian government agency. This dataset contains 160,743 log entries and a total of 120 initial features, reflecting a wide range of important information regarding network traffic activity [18]. Classification labels in this dataset were determined based on three key features: threat severity, firewall action, and application risk level. The labeling process was carried out manually by two network security experts, who categorized each log entry into one of two main classes: *threat* or *benign*. The criteria for determining ground-truth labels are presented in Table 1.

Table 1: Ground-truth Labeling Criteria

Factor	Threat	Benign
Severity	Critical, High, Medium	Informational, Low
Action	block, drop, reset-server, reset-both, sinkhole	allow, alert
Risk of Apps	5 (Very high), 4 (High), 3 (Medium)	2 (Low), 1 (Very low)
Context	Direct attack activity	Normal activity or requires further investigation

Based on the criteria outlined in Table 1, the classification process was conducted in two phases. The first phase involved analyzing actual threats. In this phase, priority was given to log entries that

indicated security mitigation actions, such as drop, reset-server, or sinkhole. The primary focus was on threat categories with Critical, High, or Medium severity levels. Examples of threats included in this category are Denial of Service (DoS) attacks, brute force attacks, vulnerability exploitation, and Command and Control (C2) communication activities.

The second phase involved analyzing traffic that was considered benign. In this stage, the focus was on log entries with firewall actions such as allow or alert. This category generally includes informational events that warrant monitoring but are not immediately categorized as threats. Examples of such activities include lightweight reconnaissance, unusual TLS/SSL traffic anomalies, and SSH access attempts that do not indicate direct exploitation. The data cleaning process included the removal of irrelevant features, empty records, and duplicate entries. These refined features were then further analyzed in the feature selection stage as described in the research methodology.

3.2.2. Secondary dataset

The secondary dataset used in this study was obtained from the UCI Machine Learning Repository and is employed to assess the model's generalization capability on external data beyond the scope of the primary research environment. This dataset contains 65,532 firewall activity records with 12 key features. Classification labels in this dataset are based on the *action* feature, which reflects four types of firewall decisions: *allow*, *deny*, *drop*, and *reset-both*.

The use of this public dataset allows for the evaluation of the model's performance across a variety of heterogeneous data conditions, while also testing the model's adaptability and robustness in real-world scenarios. Detailed information about the features included in the secondary dataset is presented in Table 2.

Table 2: Feature List of the Secondary Dataset

No.	Feature	Description	Data Type
1	Source Port	Port number on the source device initiating the session.	integer
2	Destination Port	Port number on the destination device receiving the session.	integer
3	NAT Source Port	Port number after NAT is applied to the source.	integer
4	NAT Destination Port	Port number after NAT is applied to the destination.	integer
5	Action	Action taken by the firewall (e.g., allow,	category

No.	Feature	Description	Data Type
		deny).	
6	Bytes	Total bytes transferred during the session.	integer
7	Bytes Sent	Number of bytes sent from source to destination.	integer
8	Bytes Received	Number of bytes received at the source from destination.	integer
9	Packets	Total number of packets exchanged in the session.	integer
10	Elapsed Time (sec)	Duration of the session in seconds.	integer
11	pkts_sent	Number of packets sent from source.	integer
12	pkts_received	Number of packets received at source.	integer

With its clearly structured and well-organized format, this dataset is expected to enable reliable and robust evaluation of classification models, while ensuring strong generalization capabilities in detecting network threats based on firewall log analysis.

3.3. Feature Selection

Feature selection is a crucial stage in data modeling, aimed at improving computational efficiency while enhancing classification performance by selecting the most relevant features. In this study, feature selection is carried out using a combination of two approaches: Mutual Information (MI) and Random Forest-based Feature Importance (FI-RF).

The Mutual Information (MI) approach is employed to measure the level of dependency between each feature and the target variable (classification label). Features with high MI values indicate a strong informational relationship with the target label, making them more effective and valuable in improving the performance of the classification model [19].

In parallel, the Random Forest-based Feature Importance (FI-RF) method is applied to evaluate the relevance of each feature based on its contribution to reducing impurity in decision tree nodes. In this method, features that are frequently used to split tree nodes and result in impurity reduction are assigned higher importance scores. These features are then considered to play a critical role in the classification process [20].

By combining these two techniques, the feature selection process is capable of identifying features that not only contribute to the classification model but also exhibit a strong correlation with the target labels. This strategy ensures that the resulting

model achieves higher accuracy, optimal computational efficiency, and better generalization in detecting threats from firewall log data.

3.4. Ensemble Learning

This study employs an ensemble learning approach to build a classification model that is more accurate and resilient to prediction errors. The method applied is a stacking ensemble, which integrates three machine learning algorithms as base learners: Iterative Dichotomiser 3 (ID3), Random Forest (RF), and XGBoost (XGB). Additionally, the XGBoost algorithm is also utilized as the meta-learner to generate the final prediction. An overview of the stacking ensemble architecture implemented in this study is illustrated in Figure. 2.

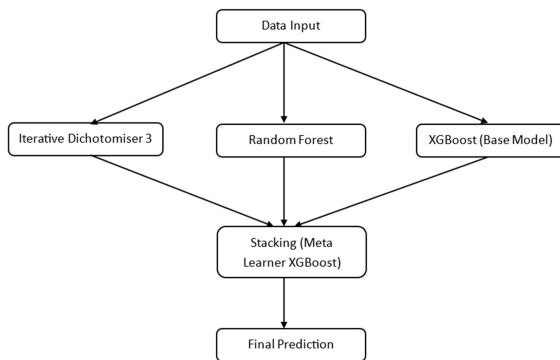


Figure 2: Architecture of the Stacking Ensemble Learning Method

The ensemble learning process begins with data input, which consists of the features selected in the earlier feature selection phase. The data is then processed in parallel by the three base algorithms:

- Iterative Dichotomiser 3 (ID3) is a decision tree-based algorithm that uses information gain as the criterion for splitting nodes. Information gain evaluates a feature's ability to reduce uncertainty or entropy during the classification process [21]. The main advantages of ID3 are its simple decision tree structure and fast classification performance.
- Random Forest (RF) is a bagging-based ensemble method that constructs multiple decision trees from randomly sampled subsets of the dataset. The final classification result is derived by aggregating the outputs of individual trees, resulting in more stable and consistent predictions and a reduced risk of overfitting on complex datasets [22].
- XGBoost (XGB) is a gradient boosting-based algorithm that incrementally corrects

classification errors at each iteration. It is well-known for its effectiveness in handling large, complex, and imbalanced datasets, and delivers highly accurate classification results [23].

The predictions generated by the three base models ID3, Random Forest, and XGBoost are then used as inputs for the next stage, where XGBoost acts as the meta-learner. At this stage, the meta-learner combines the information derived from the base learners to produce a final classification decision that is more accurate and precise than the predictions made by any single model alone.

This stacking ensemble approach is designed to leverage the unique strengths of each algorithm: the interpretability of ID3 decision trees, the prediction stability of Random Forest, and the efficiency and accuracy of XGBoost [24].

As a result, the ensemble model delivers a more robust, accurate, and efficient network threat classification system based on firewall log analysis, well-suited for a wide range of real-world scenarios.

3.5. Performance Metrics

The evaluation of model performance in this study is conducted to assess the effectiveness of the classification system and the efficiency of computing resource usage. The evaluation consists of two main aspects: classification performance and operational efficiency.

3.5.1. Classification performance evaluation

To evaluate classification quality, the following standard performance metrics are utilized:

Accuracy: Measures the proportion of correct predictions among all predictions made by the model. It provides an overall view of the model's performance.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Precision: Indicates the accuracy of positive predictions by measuring how many of the predicted positive cases are actually correct. This metric is particularly important for reducing false positives.

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

Recall: Also known as sensitivity, recall measures the model's ability to identify all actual positive cases. It is useful for minimizing false negatives.

$$Recall = \frac{TP}{TP + FN}$$

F1-Score: The harmonic mean of precision and recall. This metric is most effective in scenarios with imbalanced class distributions.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

In addition to these metrics, a confusion matrix is used for a more detailed analysis of classification errors. The structure of the confusion matrix is presented in Table 3.

Table 3. Confusion Matrix Structure

	Predicted Positive		Predicted Negative	
Actual Positive	True (TP)	Positive	False (FN)	Negative
Actual Negative	False (FP)	Positive	True (TN)	Negative

Explanation of Confusion Matrix Elements:

- True Positive (TP): Correctly predicted as positive.
- False Positive (FP): Incorrectly predicted as positive (Type I error).
- False Negative (FN): Incorrectly predicted as negative (Type II error).
- True Negative (TN): Correctly predicted as negative.

These metrics provide a comprehensive understanding of the model's ability to distinguish between benign and malicious traffic, especially in scenarios involving imbalanced datasets.

3.5.2. Evaluation of computing resource efficiency

Besides classification performance, this study also evaluates the model's computational efficiency using three key indicators: RAM usage, disk space usage, and execution time. These metrics are measured using Python scripts with the help of psutil and shutil libraries.

RAM Usage is calculated by obtaining the memory used by the process:

```
process = psutil.Process(os.getpid())
ram_usage = process.memory_info().rss /
1024**2 # in Megabytes (MB) \quad (5)
```

Disk Space Usage is measured as the difference in disk usage before and after the modeling process:

```
disk_usage_before = shutil.disk_usage("/")
# modeling process
disk_usage_after = shutil.disk_usage("/") \quad (6)
```

```
disk_usage_diff = (disk_usage_after.used -
disk_usage_before.used) / (1024 ** 2) # in MB
```

Execution Time is measured using the time taken to complete the model training and prediction:

```
start_time = time.time()
# modeling process
end_time = time.time()
execution_time = end_time - start_time # in
seconds \quad (7)
```

This dual-aspect evaluation framework ensures that the developed model not only achieves high predictive performance but also maintains efficient use of computational resources. Such consideration is especially relevant for deployment in resource-constrained environments, such as government or large-scale enterprise networks.

4. RESULT AND DISCUSSION

We evaluated the model using two datasets: the primary dataset, consisting of internal firewall logs from a government institution, and the secondary dataset obtained from the UCI Machine Learning Repository. A 70:30 split was used for training and testing, respectively. The evaluation encompasses a comparative analysis of classification performance, model generalization, and operational efficiency based on computational resource usage.

4.1. Feature Extraction, Data Labeling, and Normalization Results

During the preprocessing stage, feature extraction was performed by converting categorical attributes such as severity, action, and rule into numerical format using label encoding. This was followed by a data cleaning process that reduced the number of features from the initial 120 to 30 relevant features. Data labeling was carried out based on a combination of three main features: *severity*, *action*, and *risk of apps*. Each entry was classified into one of two labels: Threat or Benign.

Additionally, numerical data fields such as the *repeat-count* feature were normalized using min-max scaling to ensure a consistent range across all features. The results of the feature extraction process are summarized in Table 4.

Table 4: Summary of Primary Dataset Feature Extraction

Feature	Categorical	Numeric Data
Severity	critical	0
	high	1
	informational	2
	medium	3

Feature	Categorical	Numeric Data
Action	alert	0
	allow	1
	drop	2
	reset-server	3
	sinkhole	4
Feature	IP address	Numeric Data
Destination Address	10.107.1.107	4422
	10.107.25.157	4500
Feature	Date	Numeric Data
Receive Time	11/4/2024 13:10	600
	11/4/2024 13:09	599

The transformation of features into numerical data and the application of normalization techniques to the primary dataset were essential steps to prevent the dominance of certain features during model training. As a result, a clean, structured, and ready-to-use dataset was obtained for subsequent feature selection and modeling processes.

4.2. Feature Selection Results

The feature selection process, conducted using Mutual Information (MI) and Random Forest-based Feature Importance (FI-RF) methods, yielded different scores for each feature. However, several features consistently demonstrated high values across both selection methods, particularly the features thr_category and Severity.

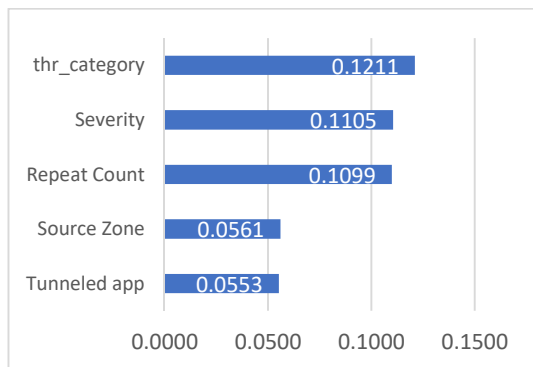


Figure 3: Top 5 Feature Importance Scores from Random Forest on Primary Data

As shown in Figure 3, the FI-RF method identified thr_category (0.1211), Severity (0.1105), and Repeat Count (0.1099) as the top-ranking features. These features contribute to the construction of decision trees in the Random Forest model. In contrast, features such as NAT Source Port and IP Protocol received very low or zero importance scores, indicating their irrelevance to the model and justifying their removal to improve computational efficiency.

Meanwhile, based on the Mutual Information (MI) method illustrated in Figure 4, features such as Outbound Interface (0.3889), thr_category (0.3776), and Threat/Content Name (0.3717) achieved the highest MI values, indicating a strong correlation with the classification target label. The Severity and Repeat Count features also ranked high in MI scores, showing consistency with the FI-RF results. Conversely, features like url_idx (0.0019) and Source Country (0.0243) received low scores, implying a weak relationship with the classification target.

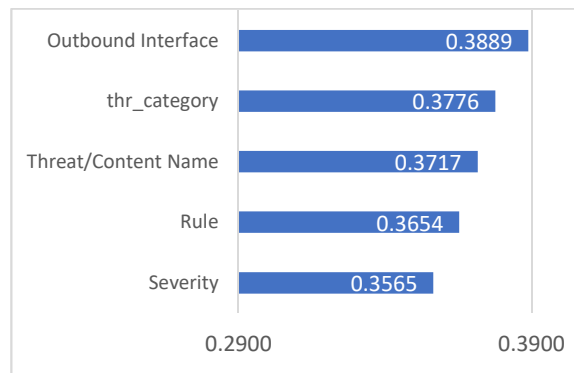


Figure 4: Top 5 Mutual Information Scores on Primary Data

Based on the results of both selection methods, a total of 30 key features were selected from the original 120 features to be used as input in the classification modeling on the primary dataset. These features are summarized in Table 5.

Table 5: Datasets and Features Used in the Classification Model

Data Source	Label	Selected Features
Primary Dataset - Palo Alto 5220 Firewall Threat Logs	Threat and Benign	Outbound Interface, thr_category, Threat/Content Name, Rule, Severity, Repeat Count, Destination address, Application, Inbound Interface, Source address, Destination Port, Tunneled app, Technology of app, Characteristic of app, Risk of app, Destination Country, Source Port, Category of app, Subcategory of app, Threat/Content Type, Source Zone, Generate Time, contentver, Receive Time, Time Logged, Session ID, High Res Timestamp, Action, Category, Flags.
Secondary Dataset - UCI Firewall Logs	allow, deny, drop, reset-	Source Port, Destination Port, NAT Source Port, NAT Destination Port, Bytes, Bytes Sent, Bytes Received, Packets,

Data Source	Label	Selected Features
	both	Elapsed Time (sec), pkts_sent, pkts_received.

In the secondary dataset, as shown in Figs. 5 and 6, the Destination Port feature achieved the highest score in both feature selection methods 0.9218 (MI) and 0.2404 (FI-RF). On the other hand, the pkts_sent feature was consistently evaluated as having minimal influence in both methods. The differences in importance rankings of other features, such as Bytes, indicate that the two selection methods complement each other in evaluating feature relevance.

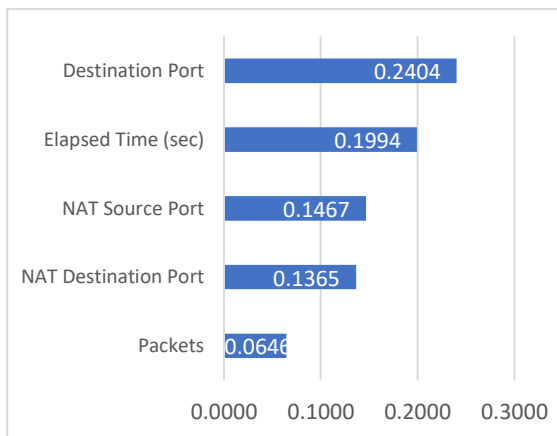


Figure 5: Top 5 Feature Importance Scores from Random Forest on Secondary Data

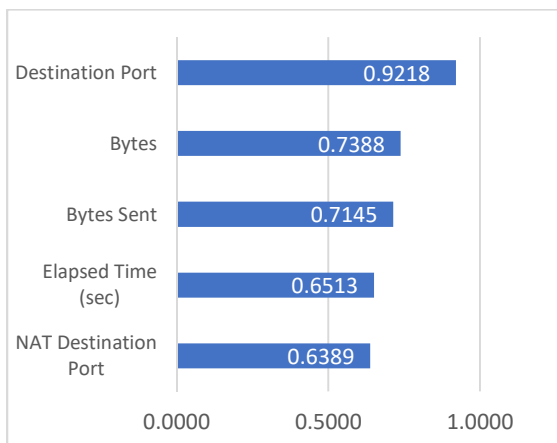


Figure 6: Top 5 Mutual Information Scores on Secondary Data

These findings support the robustness of using dual selection techniques to identify the most informative and relevant features, thus enhancing the accuracy and generalization of the threat detection model.

4.3. Evaluation of the Primary Dataset

The performance evaluation using the primary dataset involved a comparison among four methods: ID3, Random Forest, XGBoost, and the stacking ensemble approach. The evaluation results are presented in Table 6.

Table 6: Model Evaluation on Primary Dataset

Meth od	Train Time (s)	Predi ct Time (s)	RAM Usage (GB)	Disk Usage (GB)	Accu racy (%)	FP + FN
ID3	0.859	0.005	1.5	36.97	99.97	15
RF	13.59	0.146	1.5	36.97	99.97	14
XGB	1.103	0.073	1.5	36.97	99.97	14
Ense mble	82.63	0.255	1.4	36.97	99.97	13

As shown in Table 6, the stacking ensemble learning model achieved the highest accuracy and lowest total misclassification (FP + FN), although it required longer training time compared to the other models. The evaluation results demonstrate that the stacking ensemble method yielded the fewest classification errors (13 cases), outperforming the individual models in terms of predictive reliability.

4.4. Evaluation of the Secondary Dataset

Evaluation using the secondary dataset (from the UCI Repository) was conducted to assess the model's adaptability to external data. The evaluation results are presented in Table 7.

Table 7: Model Evaluation on Secondary Dataset

Meth od	Train Time (s)	Predi ct Time (s)	RAM Usage (GB)	Disk Usage (GB)	Accu racy (%)	FP + FN
ID3	0.106	0.001	1.1	36.97	99.76	48
RF	4.078	0.101	1.1	36.97	99.81	38
XGB	1.697	0.104	1.1	36.97	99.86	28
Ense mble	32.31	0.302	1.1	36.97	99.87	26

The stacking ensemble once again delivered the best performance, achieving the highest accuracy and the lowest number of misclassification errors (FP + FN). These results highlight the effectiveness of the ensemble approach in handling heterogeneous data with imbalanced class distributions, demonstrating its robustness and generalization capability across diverse network traffic scenarios.

4.5. Comparison with Previous Studies

Table 8 summarizes prior work on an 11-feature firewall log dataset (secondary dataset).

Table 8: Summary of the Related Work

Reference	Model	Features	Result
[9]	SVM	11 features	Recall = 98.5% Precision = 67.5%
[10]	SOFM	undefined features	Accuracy= 97.20%
[11]	Ensemble RF	11 features	Accuracy= 99.80%
[12]	DT	11 features	Accuracy= 99.83%
Our Method	SVM	11 features	Accuracy= 98.82%
Our Method	Stacking Ensemble	11 features	Accuracy= 99.87%

Our stacking ensemble model trained on the UCI dataset with 11 selected features achieved 99.87 % accuracy, outperforming all previous approaches in terms of both accuracy and classification stability. To further assess the robustness of our feature set, we trained a Support Vector Machine (SVM) with an RBF kernel on the same 11 selected features using 10-fold cross-validation. The SVM achieved 98.82 % accuracy, only 1.05 pp lower than our stacking ensemble, confirming that the features we selected generalize well across different classifiers.

4.6. Confusion Matrix Visualization

The confusion matrix visualization is provided to analyze the error patterns of the stacking ensemble model on both the primary and secondary datasets. As illustrated in Figure 7, for the primary dataset, which involves binary classification (*benign* vs. *threat*), the model successfully classified most instances correctly, with only a small number of false positives and false negatives.

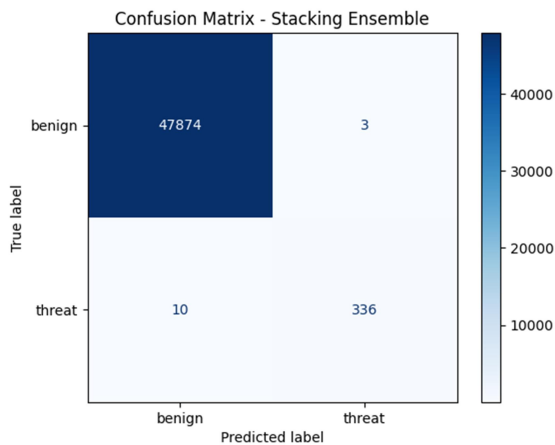


Figure 7: Confusion Matrix - Ensemble Learning on Primary Dataset

In the secondary dataset (Figure 8), which involves multi-class classification, the model performed well

on the majority classes (*allow*, *deny*, *drop*), but showed difficulty in accurately classifying the minority class (*reset-both*). This highlights the model's limitations in managing class imbalance.

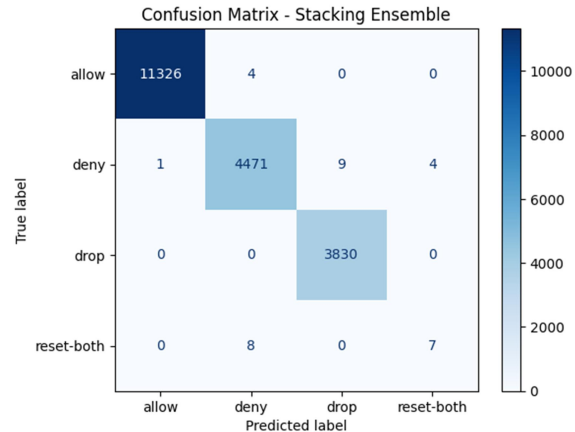


Figure 8: Confusion Matrix - Ensemble Learning on Secondary Dataset

4.7. Performance Efficiency Analysis and Limitations

The overall analysis reveals a correlation between model accuracy and resource efficiency. The ID3 model demonstrated superior performance in terms of training and prediction speed but exhibited a higher error rate compared to the other models. Both Random Forest and XGBoost provided a balanced trade-off between classification performance and computational efficiency. In contrast, the stacking ensemble model achieved the highest accuracy, yet required longer training time and higher resource consumption. Therefore, model selection can be tailored to specific requirements favoring either accuracy (with the stacking ensemble) or operational efficiency (with ID3 or XGBoost), depending on the priorities and constraints of the deployment environment.

This study has several limitations. First, the evaluation was only conducted on two historical datasets and has not been tested on real-time streaming logs. Second, no manual parameter adjustments were made for each base model. Third, the log formats tested were limited to standard firewall CSV formats and did not include Syslog or NetFlow formats. These factors may limit direct application to diverse operational environments.

5. CONCLUSION AND FUTURE WORK

This study demonstrates that the integration of ensemble learning methods utilizing the ID3,

Random Forest, and XGBoost algorithms, combined with Mutual Information (MI) and Random Forest-based Feature Importance (FI-RF) feature selection techniques, can enhance the effectiveness of classification systems for detecting network threats based on firewall log analysis. Compared to single machine learning algorithms, the proposed ensemble approach proved effective in reducing classification errors, including false positives and false negatives.

Although ensemble learning introduces a relatively longer training time due to the integration of predictions from multiple base algorithms, the improved accuracy and classification stability it offers represent major advantages. In terms of resource consumption, the ensemble approach incurs a slightly higher RAM usage than individual models, while disk space requirements are primarily influenced by the size of the dataset used.

The evaluation was conducted using two distinct datasets: a primary dataset consisting of 160,743 firewall log entries from a government institution in Indonesia, and a secondary public dataset from the UCI Machine Learning Repository containing 65,532 entries. Experimental results showed high accuracy (above 99%) for both datasets, along with stable performance and strong generalization capability when applied to external data.

The proposed method shows practical applicability in governmental cyber threat detection systems, especially those dealing with large-scale firewall log data. For future research, the incorporation of deep learning approaches or hybrid models with Autoencoders may be explored to detect more complex and dynamic threats in modern network environments.

AUTHOR CONTRIBUTORSHIP

Noordama: Conceptualization, Methodology, Software, Formal analysis, Investigation, Resources, Data Curation, Writing - Original Draft, Visualization. Benfano Soewito: Conceptualization, Methodology, Validation, Writing - Review & Editing, Supervision.

DATA AVAILABILITY

To increase the transparency of this paper, the authors have provided data sources. The data is available at the Zenodo Repository: <https://zenodo.org/records/15766671>.

REFERENCES

- [1] Badan Siber dan Sandi Negara (BSSN), "Cyber Security Challenges and Digital Resilience in Indonesia," *Telkom Digital Solution*, [White Paper], n.d. [Online]. Available: <https://www.telkomdigitalsolution.com/storage/file/magazine/bAx0CIFHbatV179uDK3GQhJeUkJDfjlsxaqHEc3Z.pdf>
- [2] M. H. Rahman, T. Islam, M. M. Rana, R. Tasnim, T. R. Mona, and M. M. Sakib, "Machine learning approach on multiclass classification of internet firewall log files," *arXiv preprint*, arXiv:2306.07997, 2023. [Online]. Available: <https://arxiv.org/abs/2306.07997>
- [3] C. Y. Ho, Y. D. Lin, Y. C. Lai, I. W. Chen, F. Y. Wang, and W. H. Tai, "False positives and false negatives from real traffic with intrusion detection/prevention systems," *International Journal of Future Computer and Communication*, vol. 1, no. 2, pp. 87–90, 2012. [Online]. Available: <https://www.ijfcc.org/papers/23-T00008.pdf>
- [4] A. Parashar, K. S. Saggi, and A. Garg, "Machine learning based framework for network intrusion detection system using stacking ensemble technique," *Indian Journal of Engineering & Materials Sciences (IJEMS)*, vol. 29, no. 4, 2022. [Online]. Available: <http://op.niscpr.res.in/index.php/IJEMS/article/download/46838/465481321>
- [5] M. Aljabri, A. A. Alahmadi, R. M. A. Mohammad, M. Aboulnour, D. M. Alomari, and S. H. Almotiri, "Classification of firewall log data using multiclass machine learning models," *Electronics*, vol. 11, no. 12, p. 1851, 2022. [Online]. Available: <https://www.mdpi.com/2079-9292/11/12/1851>. doi: 10.3390/electronics11121851
- [6] A. Devarakonda, N. Sharma, P. Saha, and S. Ramya, "Network intrusion detection: A comparative study of four classifiers using the NSL-KDD and KDD'99 datasets," *J. Phys.: Conf. Ser.*, vol. 2161, no. 1, p. 012043, 2022. doi: 10.1088/1742-6596/2161/1/012043
- [7] A. H. A. Monfared and K. Borna, "Firewall Log Classification: Leveraging Feature Selection and Stacking Ensemble," *ResearchSquare*, 2024. [Online]. Available: <https://www.researchsquare.com/article/rs-4508259/latest>
- [8] K. Al Jallad, M. Aljnidi, and M. S. Desouki, "Anomaly detection optimization using big

- data and deep learning to reduce false-positive,” *Journal of Big Data*, vol. 7, no. 1, pp. 1–12, 2020. doi: 10.1186/s40537-020-00346-1
- [9] F. Ertam and M. Kaya, “Classification of firewall log files with multiclass support vector machine,” in *Proc. 2018 6th Int. Symp. on Digital Forensic and Security (ISDFS)*, Antalya, Turkey, Mar. 2018, pp. 1–4. doi: 10.1109/ISDFS.2018.8355382
- [10] S. Allagi and R. Rachh, “Analysis of Network log data using Machine Learning,” in *Proc. 2019 IEEE 5th Int. Conf. for Convergence in Technology (I2CT)*, Pune, India, Mar. 2019, pp. 1–3. doi: 10.1109/I2CT45611.2019.9033737
- [11] D. Sharma, V. Wason, and P. Johri, “Optimized classification of firewall log data using heterogeneous ensemble techniques,” in *Proc. 2021 Int. Conf. Advance Computing and Innovative Technologies in Engineering (ICACITE)*, Mar. 2021, pp. 368–372.
- [12] H. N. K. Al-Behadili, “Decision tree for multiclass classification of firewall access,” *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 3, pp. 294–302, 2021.
- [13] S. Dalvi, G. Gressel, and K. Achuthan, “Tuning the false positive rate/false negative rate with phishing detection models,” *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 9, no. 2, pp. 7–13, 2019. [Online]. Available: <https://www.researchgate.net/publication/338571436>
- [14] I. D. Mienye and Y. Sun, “A survey of ensemble learning: Concepts, algorithms, applications, and prospects,” *IEEE Access*, vol. 10, pp. 99129–99149, 2022. doi: 10.1109/ACCESS.2022.3201435
- [15] Z. Ling and Z. J. Hao, “Intrusion detection using normalized mutual information feature selection and parallel quantum genetic algorithm,” *International Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 18, no. 1, pp. 1–24, 2022. doi: 10.4018/IJSWIS.307324
- [16] S. M. Kasongo and Y. Sun, “Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset,” *Journal of Big Data*, vol. 7, no. 1, p. 105, 2020. doi: 10.1186/s40537-020-00379-6
- [17] A. Komadina, I. Kovačević, B. Štengl, and S. Groš, “Comparative analysis of anomaly detection approaches in firewall logs: Integrating light-weight synthesis of security logs and artificially generated attack detection,” *Sensors*, vol. 24, no. 8, p. 2636, 2024. doi: 10.3390/s24082636
- [18] Palo Alto Networks, “Threat log fields,” *Palo Alto Networks Documentation*. [Online]. Available: <https://docs.paloaltonetworks.com/pan-os/9-1/pan-os-admin/monitoring/use-syslog-for-monitoring/syslog-field-descriptions/threat-log-fields>
- [19] Q. Zhang, M. Lu, S. Yu, J. Xin, and B. Chen, “An information bottleneck approach for feature selection,” *Pattern Recognition*, vol. 144, p. 111564, 2025.
- [20] K. Gajowniczek and M. Dudziński, “Influence of explanatory variable distributions on the behavior of the impurity measures used in classification tree learning,” *Entropy*, vol. 26, no. 12, p. 1020, 2024.
- [21] J. R. Quinlan, “Induction of decision trees,” *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986. doi: 10.1007/BF00116251
- [22] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. doi: 10.1023/A:1010933404324
- [23] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, San Francisco, CA, USA, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785
- [24] D. H. Wolpert, “Stacked generalization,” *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992. doi: 10.1016/S0893-6080(05)80023-1