Enhancing ML model efficiency for DDoS attack detection on web servers in resource-constrained environments

Qais Saif Qassim[1,†] and Nessibeli Askarbekova [2,†]

1 College of Computing and Information Sciences, University of Technology and Applied Sciences-Ibri, Ibri, Oman
2 International Information Technology University, 34/1 Manas St., 050000, Almaty, Kazakhstan

Abstract
Distributed Denial of Service (DDoS) attacks, particularly High Orbit Ion Cannon (HOIC) attacks, present a significant threat to web servers. This study focuses on enhancing feature selection and model efficiency to ensure accurate detection while minimizing computational resources. The study aims to demonstrate that effective DDoS detection can be achieved with fewer instances and a limited number of key features. This approach not only optimizes the computational resources required for model training but also highlights the potential for deploying efficient and scalable detection systems in real-world environments where data processing capacity may be constrained. The selected features and resampled datasets form the basis for developing and evaluating machine learning models capable of accurately identifying HOIC attacks across varying dataset sizes.

Keywords:
Feature Selection, Classification, DDoS, Intrusion Detection

1. Introduction

In an era where the internet has become integral to virtually every aspect of modern life, the security of web applications is paramount. As businesses, governments, and individuals increasingly rely on these applications for critical services, the threat landscape has evolved, with attackers developing sophisticated methods to disrupt and compromise systems. Among the large number of today's cyber threats, Distributed Denial of Service (DDoS) attacks have emerged as one of the most disruptive, capable of overwhelming web applications and rendering them inaccessible to legitimate users [1]. DDoS attacks stand out from other cyber threats due to their ability to cause widespread disruption with relatively minimal effort on the part of the attacker. By leveraging a large number of compromised devices, attackers can generate vast volumes of malicious traffic that can saturate a target's network infrastructure, leading to significant downtime, financial losses, and damage to an organization's reputation [2]. Within the spectrum of DDoS attacks, the High Orbit Ion Cannon (HOIC) represents a particularly potent threat, characterized by its ability to generate large-scale traffic surges that can evade traditional security measures [3].

The HOIC is a powerful tool used by cyber criminals, specifically for executing Distributed Denial of Service attacks. It is designed to overwhelm multiple targets simultaneously [4]. This capability makes it a more sophisticated and versatile weapon in the arsenal of cyber attackers. HOIC operates by flooding a target server with a massive volume of HTTP 'GET' and 'POST' requests. These requests are designed to drain the server's capabilities and render it unable to respond to legitimate users [3]. The tool's capacity

DTESI 2024: 9th International Conference on Digital Technologies in Education, Science and Industry, October 16–17, 2024, Almaty, Kazakhstan
* Corresponding author.
qqassim@acm.org (Q. Qassim); n.askarbekova@iitu.edu.kz (N. Askarbekova)
0000-0002-6391-5246 (Q. Qassim)

to send several requests concurrently, paired with its multi-threading capabilities, enables it to produce a large quantity of traffic in a short period of time.

One of the distinguishing characteristics of HOIC is the usage of "boosters." These are add-ons that effectiveness the tool's efficacy by customizing the attack to exploit specific vulnerabilities on the target server [4]. Boosters can also obfuscate the attack traffic, making it difficult for defense mechanisms to detect and mitigate the threat [3]. HOIC's versatility and capacity to enhance the intensity of the attack make it an intimidating tool for carrying out large-scale DDoS attacks. Despite its destructive power, HOIC is rather user-friendly, having an interface that enables even those with limited technical expertise to launch an attack. This accessibility has increased its popularity among cyber attackers. However, it should be noted that employing HOIC for harmful reasons is both unlawful and unethical. Understanding its mechanics is crucial for cybersecurity professionals to develop effective countermeasures and protect digital infrastructure from such threats [4], [5].

This paper aims to underscore the significance of DDoS attacks in the context of web application security, drawing comparisons with other forms of cyber threats. Furthermore, by analyzing a dataset sourced from Kaggle, this research seeks to identify key network traffic parameters that are indicative of HOIC attacks. By understanding these parameters, it becomes possible to develop more effective detection mechanisms, thereby enhancing the resilience of web applications against these pervasive threats.

2. Modifications

This section demonstrates some of the efforts that have been made to address the attacks on web servers and web applications. One of the key areas of focus has been the improvement of Intrusion Detection Systems (IDS), which play an important role in detecting malicious activity in real time. These systems use machine learning algorithms, anomaly detection techniques, and signature-based methods to analyze network data, identify patterns that indicate an attack, and respond quickly to reduce its damage.

Ramezany et. al. [6] have presented a machine learning-based malicious payload detection and classification mechanism based on TF-IDF feature vector construction. The study has proposed a new HTTP attack dataset composed of eight attack categories (including RCE, Injection, LFI- lFD, XSS, XXE, Open-Redirect, CRLF, and Deserialize) and clean category for non-malicious payload. The instances of the dataset have been collected using honeypot machines and classified by a domain expert. To verify the proposed model, the authors have performed a comparative study on three algorithms applied to the framework: Support Vector Machine, Random Forest, and Stochastic Gradient Descent. The experimental results showed that Random Forest outperforms other methods, given that sufficient samples are available.

Vartouni et. al. [7] proposes a machine learning-based firewall for web applications that relies on anomaly detection. The proposed approach analyzes HTTP traffic and uses an n-gram model based on character to create features from HTTP data. To minimize the problem's dimensionality, a stacked autoencoder (SAE) with various configurations is used to extract relevant features from data. Finally, to discover anomalies, an isolation forest method is used. The proposed model was tested using the CSIC 2010 dataset. The authors concluded that deep learning algorithms based on sigmoid have good generalization based on detection rate and specificity. Hashim et. al. [8] have proposed a machine learning-based detection algorithm to identify phishing websites. The proposed system also recommends a mitigation technique for different web application attacks. The proposed method utilizes three machine learning algorithms (Support Vector Machine, Random Forest and Logistic regression) and deep learning using Long Short Term Memory (LSTM). 98% detection accuracy was achieved using LSTM.

Protecting against SQL attacks, Gogoi et. al. [9] proposed an SQL injection detection method using NLP and Machine Learning. The ML algorithms are trained using normal inputs and SQLi payloads. The

normal input is generated using the probabilistic method. On the other hand, the SQLi Attack payloads are obtained from tools and manual settings. The experimental results showed that using ML and NLP techniques can improve the traditional detection approaches. Moreover, SVM classification has outperformed other machine learning algorithms. Another SQL-related attack detection system has been proposed by Joshi and Geetha [10]. The study proposed a detection method based on Naïve Bayes Machine Learning Algorithm combined with Role Based Access control mechanism. During the learning phase, the application receives the training dataset from text files and applies each data to the classifier's learning mechanism. The classifier learns using machine learning feature vectors generated by the blank separation and tokenizing method from data collected. The function of the user is also included in the feature vector, which is utilized for classification using the Role-Based Access Control mechanism. During the classification process, the application reads the test dataset from text files and applies each data to the classifier's classification procedure. Classification is performed using the generated feature vector. Precision and Recall are used in the application to examine the classification results of the Nave Bayes machine learning approach.

To detect web application injection threats, a one-class Support Vector Machine (SVM) was proposed by Zhou et. al. [11]. The authors proposed that detection of injection attacks be approached as an anomaly detection challenge. Several genuine HTTP requests are used to train a one-class SVM model during the training phase. The trained one-class SVM is used in the testing stage to determine if an HTTP request is authentic or malicious. To extract features from HTTP requests, we use the 2u-gram technique (a version of n-gram). The experimental results reveal that a one-class SVM detects web application injection attacks with a 94.04% detection rate and a 1.62% false positive rate. Another machine learning-based detection system is presented by Banerjee et. al. [12]. The authors have considered Cross-site scripting detection by looking at the URL and the JavaScript. The study proposed employing four machine learning techniques to estimate the severity of XSS threat (SVM, KNN, Random Forest, and Logistic Regression). After analyzing and evaluating the dataset features, it was determined that the Random Forest Classifier performed the best, with a False Positive Rate of 0.34.

Blind cross-site scripting detection was proposed by G. Kaur et al. [13] using machine learning. The proposed detection method investigates the presence of dangerous payloads that were likely to be stored in databases via online applications. The Linear Support Vector Machine classifier has been utilized to detect the malicious scripts. The system extracts JavaScript events which attackers usually use to insert vulnerable payloads. The extracted events were utilized as training vectors to construct the hyperplane of the SVM. The same has been used for testing purposes as well. The experimental results showed that the proposed approach could detect blind XSS attacks with 95.4% accuracy. Sharma, Zavarsky and Butakov [14] claimed that the selection of the features plays a vital role in improving the detection system's accuracy. Therefore, the study suggested a fine-tuned feature set extracted from the generic CSIC 2010 HTTP dataset. A specialized software tool has been developed to extract the required features/. The experimental results with three machine learning algorithms (J48, Naïve Bayes, OneR) demonstrate the reliability in detecting web-based attacks. The J48 decision tree algorithm was depicted to be the best-performing algorithm, with the best attack detection rate of 94.5%.

The application of machine learning in web security was not only through proposing a detection system nor improving the performance of WFA. Vulnerability identification has its share, such that several researchers offered to implement web applications' vulnerability scanners using machine learning. For example, Tommy, Sundeep and Jose [15] proposed a system which helps to automatically find and fix the vulnerabilities present in web applications using machine learning. Machine learning helps to improve the system's performance using the statistics of the previous results. The proposed method uses web crawls to reach every web application page and utilizes pre-defined payloads to test several vulnerabilities, including SQL injection and Cross-Site Scripting (XSS). Moreover, the proposed system can correct the identified vulnerabilities through a pre-defined set of instructions. The Support

vector algorithm is used to improve the efficiency of the vulnerability identification process. In contrast, the scan result of one system shall be utilized to determine the vulnerabilities of similar systems faster.

3. IDS 2018 intrusion dataset

The IDS 2018 intrusion dataset, developed by the University of New Brunswick [16], is a comprehensive and widely recognized resource in the field of cybersecurity research. This dataset was meticulously crafted to simulate a realistic and diverse range of network traffic, encompassing both normal and malicious activities. It serves as an invaluable tool for researchers and practitioners aiming to advance the development and evaluation of intrusion detection systems (IDS), including those designed to identify DDoS attacks. The dataset is structured to reflect contemporary network environments, incorporating various attack scenarios that mirror real-world threats. Among these, DDoS attacks are prominently featured, providing a robust foundation for analyzing the characteristics and behaviors associated with these disruptive activities. The IDS 2018 dataset includes detailed records of network traffic, such as IP addresses, port numbers, and protocol types, alongside annotations that distinguish between benign and malicious traffic. For this study, the focus will be on leveraging the IDS 2018 dataset to identify network traffic parameters that are critical for detecting HOIC attacks. By systematically analyzing the dataset, this research aims to uncover patterns and indicators that can enhance the detection capabilities of security systems, thereby contributing to the broader effort to safeguard web applications from DDoS threats.

4. Methodology

The analysis of the IDS 2018 intrusion dataset, particularly concerning the detection of HOIC attacks, was conducted using the Weka workbench, a powerful suite of machine learning software for data analysis and predictive modeling. The research methodology, as illustrated in Figure 1, followed a structured approach to ensure that the dataset was unbiased and that the most significant attributes for detecting HOIC attacks were accurately identified.

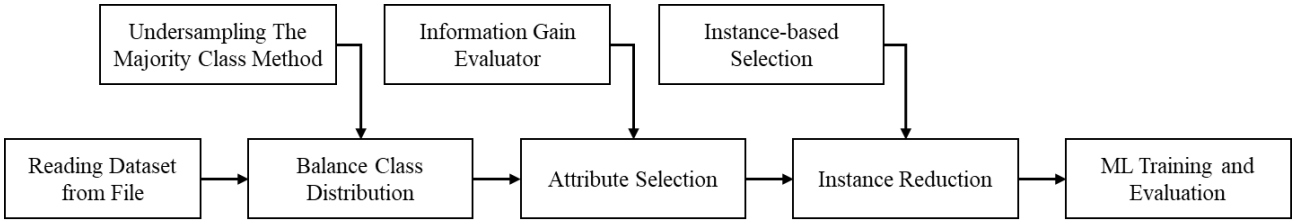

Figure 1: Research methodology phases.

4.1. Data preprocessing

Initially, the IDS 2018 dataset was filtered to isolate the network traffic instances relevant to HOIC attacks. To prevent any skewed results or biases in the analysis, the dataset was balanced by incorporating an equal number of benign instances. This balancing process was crucial to ensure that the learning algorithms could distinguish between normal and malicious traffic without being influenced by an overwhelming majority of one class. The final dataset consisted of 523,422 instances, evenly split between malicious and benign traffic, with each instance characterized by 80 attributes. Figure 2 depicts the correlation among all the 80 attributes of the dataset. As illustrated, several attributes can be eliminated due to their zero correlation with other attributes.

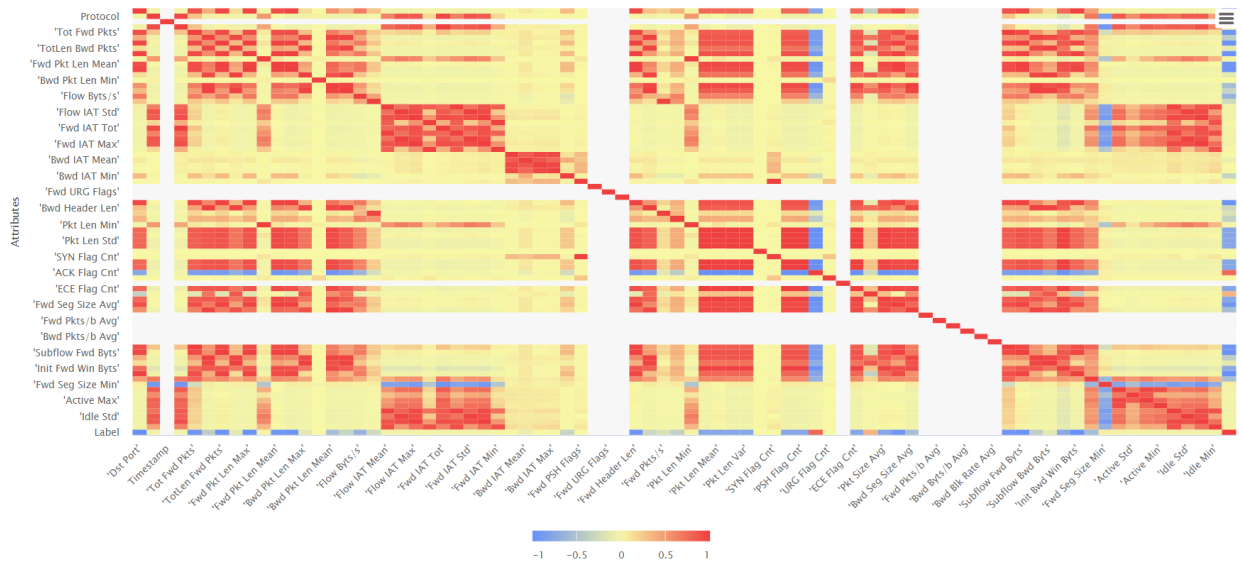

Figure 2: Correlation matrix of the dataset's attributes.

4.2. Attribute selection

To identify the most significant network traffic parameters for detecting HOIC attacks, the Information Gain (IG) was employed. Information Gain is an attribute selection technique used to reduce the number of attributes by measuring the contribution of each attribute in predicting the target variable. It is calculated by first determining the entropy, which measures the uncertainty in the dataset, and then calculating how much this uncertainty is reduced when the data is split based on a specific feature. The reduction in entropy is the Information Gain. Features are ranked by their IG scores, and those with higher scores are deemed more relevant. By setting a threshold or selecting the top-ranked features, irrelevant or redundant features are discarded, resulting in a smaller feature set that retains predictive power.

Information Gain measures the relevance of each attribute by evaluating its contribution to the prediction of the class label—in this case, whether the traffic is associated with a HOIC attack or not. This method was chosen for its effectiveness in reducing the dimensionality of the dataset while retaining the most informative features. Using Weka's built-in Information Gain attribute evaluator, each of the 80 attributes was ranked according to its importance. This ranking enabled the selection of a subset of attributes that are most indicative of HOIC attack patterns. These selected attributes form the basis for further analysis and the development of detection models. Figure 3 illustrates the score of each attribute based on the calculated Information Gain, highlighting the importance of each feature in distinguishing between HOIC attacks and benign traffic.

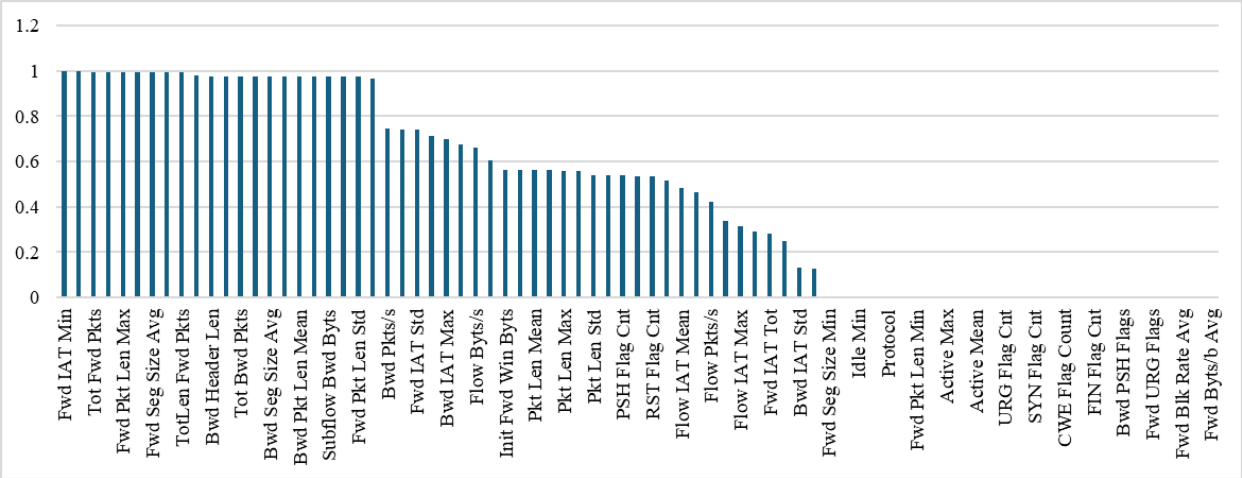

Figure 3: Information gain analysis.

Table 1
List of selected features

Feature	Description
Fwd IAT Min	Minimum time between two packets sent in the forward direction
Fwd Header Length	Total bytes used for headers in the forward direction
Tot Fwd Pkts	Total packets in the forward direction
Subflow Fwd Pkts	The average number of packets in a sub flow in the forward direction
Fwd Pkt Len Max	Maximum size of packet in forward direction
Fwd Pkt Len Mean	Mean size of packet in forward direction
Fwd Seg Size Avg	Average size observed in the forward direction
Subflow Fwd Byts	The average number of bytes in a sub flow in the forward direction
TotLen Fwd Pkts	Total size of packet in forward direction

By applying Information Gain attribute evaluator, we obtained a ranked list of attributes, with the highest-ranked features being those most strongly correlated with the occurrence of HOIC attacks. The top-ranked features based on Information Gain were selected for further analysis, as listed in Table 1. These features demonstrated a 99% correlation score with the class label, indicating their strong predictive power in identifying HOIC attack patterns. Among the 80 attributes, only 9 were chosen based on their correlation score, as these attributes provided the most significant insights while minimizing redundancy in the data. By focusing on this reduced set of highly relevant attributes, we enhance the efficiency and accuracy of the machine learning models, ensuring that they are trained on the most informative aspects of the network traffic data. This approach not only streamlines the computational process but also improves the overall performance of the detection mechanisms, making them more effective in real-world applications.

4.3. Skewed dataset

The selected dataset exhibits a significant class imbalance, with 360,833 instances labeled as Benign and only 686,012 instances labeled as Malicious. This imbalanced class distribution can lead to poor performance in machine learning algorithms, as they may struggle to accurately identify and differentiate between the minority and majority classes. When one class has a higher representation than another, the algorithm becomes biased in favor of the majority. This bias develops when the model is exposed to more instances from the majority class during training, causing it to prioritize accuracy on the dominant class over the minority class. As a result, the model may fail to accurately recognize and classify instances of the minority class, resulting in high rates of false negatives or false positives, depending on the context.

To address the challenge of class imbalance in machine learning, several techniques can be employed to improve model performance. Resampling methods, such as oversampling the minority class and undersampling the majority class, are commonly used to balance the dataset. Oversampling techniques, like SMOTE, generate synthetic instances of the minority class, while undersampling reduces the number of instances in the majority class to achieve balance. In our study, we have employed undersampling of the majority class to balance our dataset, enabling the machine learning algorithm to learn equally from both classes and improving its overall effectiveness. Figure 4 illustrates the number of instances in the original and balanced datasets.

To achieve the study's objective of reducing the number of features and instances required to train machine learning models for detecting DDoS attacks while maintaining detection accuracy, the overall number of instances in the balanced dataset was further resampled as illustrated in Figure 4. This process involved generating three subsets of the balanced dataset to evaluate the effectiveness of the selected features under different sample sizes. The first subset (Dataset A) consisted of 60% of the balanced dataset, providing a substantial amount of data to train the ML models while still reducing the overall computational load. The second subset (Dataset B) included 40% of the balanced dataset, allowing for a comparison of model performance as the dataset size decreased. The third subset (Dataset C), comprising 20% of the dataset, was created to test the models' ability to maintain detection accuracy with a significantly reduced amount of data.

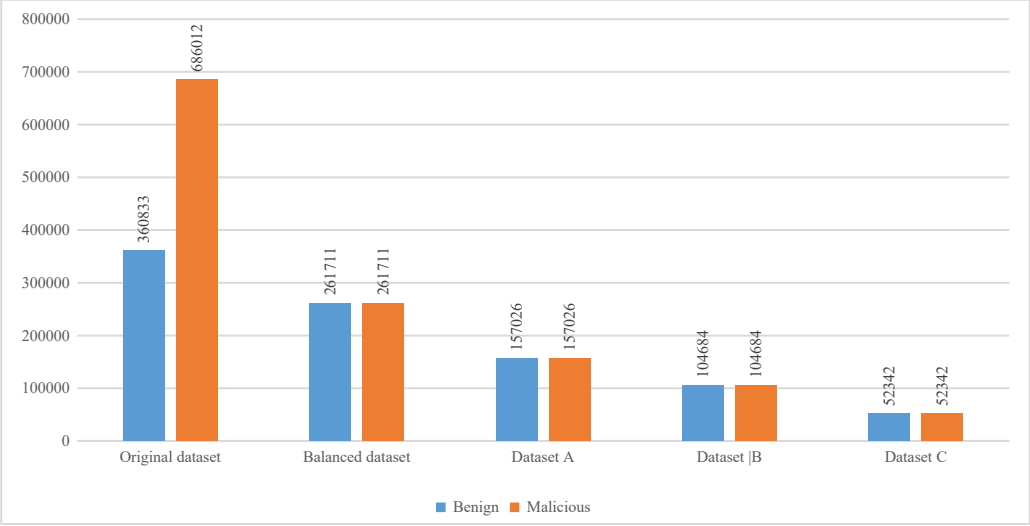

Figure 4: Distribution of benign and malicious instances across all datasets.

By resampling the dataset in this manner, the study aims to demonstrate that effective DDoS detection can be achieved with fewer instances and a limited number of key features. This approach not

only optimizes the computational resources required for model training but also highlights the potential for deploying efficient and scalable detection systems in real-world environments where data processing capacity may be constrained. The selected features and resampled datasets form the basis for developing and evaluating machine learning models capable of accurately identifying HOIC attacks across varying dataset sizes.

4.4. Model development and evaluation

Following the identification of significant attributes, these features will be utilized to train and test various machine learning models within Weka. The performance of these models will be evaluated based on their detection accuracy and the time required to build the machine learning model, which directly impacts the feasibility of deploying such models in real-time or resource-constrained environments. The models are built based on 66% of the provided data and have been tested based on the remaining portion. allowing for an unbiased evaluation of their performance. This approach provides a clear measure of how well the models generalize to new, unseen data, ensuring that the results accurately reflect the models' real-world applicability in detecting HOIC attacks.

J48 - The Weka workbench implementation of the C4.5 decision tree method that often used in categorization due to their simplicity and interpretability. The J48 algorithm iteratively divides the dataset by the attribute with the greatest Information Gain at each node to create a decision tree. The method continues until it reaches a stopping condition, such as a minimum leaf occurrence or tree depth.

JRip - It is Weka's variant of the RIPPER algorithm, which is a rule-based learner that use repeated incremental pruning to produce error reduction. JRip derives a collection of if-then rules from the training data, which are employed for the purpose of categorizing fresh occurrences. The algorithm incrementally incorporates rules to accurately represent positive instances while reducing errors on the negative instances. After the generation of the initial set of rules, JRip proceeds to selectively eliminate any superfluous complexity, hence improving the model's capacity to generalize. Rule-based classifiers such as JRip are highly appreciated for their interpretability and remarkable performance in dealing with datasets that contain noise.

Multilayer Perceptron - A type of artificial neural network that consists of an input layer, one or more hidden layers, and an output layer. Each layer contains neurons (nodes) that are interconnected, with each connection assigned a weight. MLPs are powerful classifiers capable of capturing complex patterns in data through the backpropagation learning algorithm, which adjusts the weights based on the error between the predicted and actual outputs. MLPs are particularly well-suited for problems where the relationship between the input features and the output is non-linear. Despite their computational complexity, MLPs are highly effective for a wide range of classification tasks, including those involving large and intricate datasets.

Random Forest - It constructs many decision trees during training and calculates the mode of the classes for classification or the mean prediction for regression. Each tree in the forest uses a randomly selected portion of the training data and divides nodes using a random selection of characteristics. This method reduces overfitting and improves generalization. The model is more resilient and precise due to the large range of tree variances. The Random Forest algorithm is known for its ability to analyze large datasets with many features and resist overfitting.

SMO (Sequential Minimal Optimization) – It is Weka's implementation of Support Vector Machine (SVM), used for classification. SVM maximizes the margin between class nearest points (support vectors) by selecting the hyperplane that best classifies data. SMO optimizes this process by dividing down SVM's massive quadratic programming problem into smaller, analytically solvable ones. SMO is efficient and scalable for huge datasets. SVM and SMO are ideal for difficult classification jobs due to their accuracy and efficiency in high-dimensional domains.

The following table presents the performance evaluation of the machine learning models used in this study. The table includes metrics for detection accuracy, Ture Positive (TP) rate, False Positive (FP) rate, Precision, Recall and F-Measure. The table provides insights into both the effectiveness and efficiency of the models in identifying HOIC attacks.

Table 2
Performance evaluation of selected ML

Dataset	Machine Learning Method	Accuracy	TP Rate	FP Rate	Precision	Recall	F-Measure
Dataset A	J48	99.9963	1.000	0.000	1.000	1.000	1.000
	JRip	99.9953	1.000	0.000	1.000	1.000	1.000
	Multilayer Perceptron	99.9663	1.000	0.000	1.000	1.000	1.000
	Random Forest	99.9981	1.000	0.000	1.000	1.000	1.000
	SMO	99.9532	1.000	0.000	1.000	1.000	1.000
Dataset B	J48	99.993	1.000	0.000	1.000	1.000	1.000
	JRip	99.993	1.000	0.000	1.000	1.000	1.000
	Multilayer Perceptron	99.9621	1.000	0.000	1.000	1.000	1.000
	Random Forest	99.9958	1.000	0.000	1.000	1.000	1.000
	SMO	99.9466	0.999	0.001	0.999	0.999	0.999
Dataset C	J48	99.9916	1.000	0.000	1.000	1.000	1.000
	JRip	99.9944	1.000	0.000	1.000	1.000	1.000
	Multilayer Perceptron	99.9747	1.000	0.000	1.000	1.000	1.000
	Random Forest	100	1.000	0.000	1.000	1.000	1.000
	SMO	99.9438	0.999	0.001	0.999	0.999	0.999

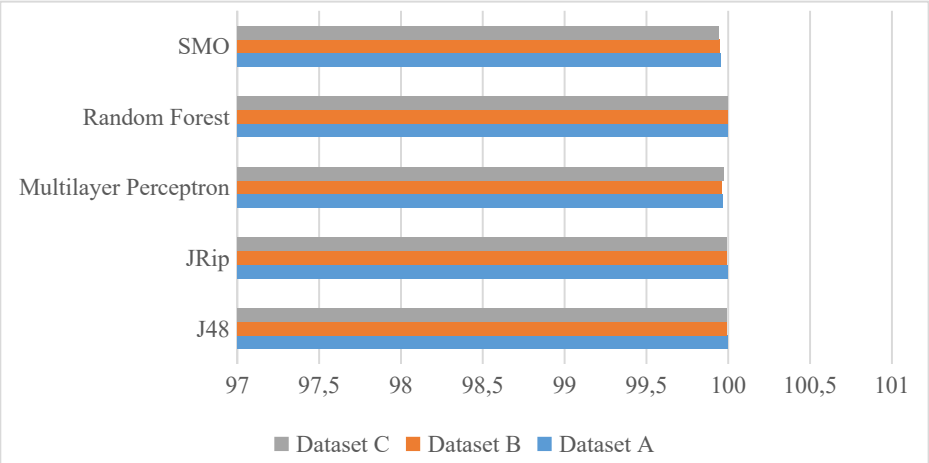

Figure 5: Comparative analysis of ML accuracy performance.

Figure 5 illustrates the comparative analysis of accuracy performance across the different machine learning models used in this study. The figure presents a bar chart showing the detection accuracy of each model, providing a visual representation of how effectively each algorithm identifies HOIC attacks.

The experimental results, as depicted in Table 2 and illustrated in Figure 5, indicate that the machine learning models were able to achieve similar accuracy even after reducing the number of instances. This outcome demonstrates the effectiveness of the selected features and the robustness of the models in maintaining high detection accuracy, despite the decrease in the amount of training data. The ability to achieve comparable accuracy with fewer instances not only validates the feature selection process but also highlights the potential for more efficient model training and deployment in resource-constrained environments. On the other hand, Table 3 depicts the time required to train the model based on 66% of the provided dataset.

Table 3
Time taken to build the model

	J48	JRip	Multilayer Perceptron	Random Forest	SMO
Dataset A	5.26	11.84	199.29	128.49	2.49
Dataset B	3.34	6.02	135.72	67.7	1.55
Dataset C	0.94	2.85	88.23	25.51	0.85

Figure 6 visually highlights the varying computational demands of different algorithms by showing the amount of time measured in seconds, each model takes to complete the training process. This comparison is essential in understanding the trade-offs between model complexity and the amount of the training data samples, especially when deploying these models in real-world scenarios where resources such as processing power and time are limited.

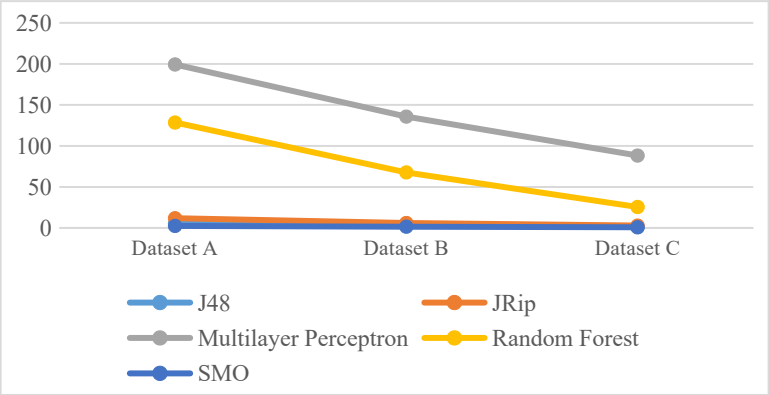

Figure 6: Comparative Analysis of ML Model Building Times.

As illustrated in Figure 6, reducing the amount of the dataset has significantly decreased the time required to build the machine learning models, while maintaining high detection performance. This reduction in model training time is crucial, particularly for future applications where intrusion detection systems need to be implemented in environments where time and computational power are critical constraints. The ability to efficiently train models with fewer instances without compromising accuracy opens up opportunities for deploying these systems in real-time scenarios, such as in edge computing or embedded systems, where resources are limited but rapid detection is essential. This efficiency ensures

that robust security measures can be maintained even in environments with stringent computational requirements.

5. Conclusion

This study set out to identify the optimal features and machine learning models for detecting High Orbit Ion Cannon (HOIC) attacks within the IDS 2018 dataset, with a focus on balancing detection accuracy and computational efficiency. Through the use of various machine learning algorithms—including J48, JRip, Multilayer Perceptron, Random Forest, and SMO—models were trained and evaluated to determine their effectiveness in recognizing HOIC attacks while minimizing the time required for model building.

The experimental findings showed that it is feasible to sustain excellent detection accuracy even after much fewer cases in the dataset are used. This result is important since it implies that less data may be used to build efficient intrusion detection systems, therefore saving computational resources required for training without compromising performance. Furthermore, the study underlined how significantly less time was needed to create machine learning models by shrinking the dataset size. Practical applications like real-time network monitoring, edge computing, and other resource-limited contexts where processing power and time are critical restrictions depend notably on this decrease in training time. The ability to train models quickly while retaining accuracy ensures that intrusion detection systems can be deployed efficiently in these scenarios.

In conclusion, the study successfully identified a set of features and models that not only achieve high accuracy in detecting HOIC attacks but also do so in a computationally efficient manner. These findings pave the way for the development of robust and scalable intrusion detection systems that can operate effectively in environments where rapid detection and resource optimization are essential.

Declaration on Generative AI

During the preparation of this work, the authors used GAI tools in order to: Proofread the text. After using these tools, the authors reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] M. Najafimehr, S. Zarifzadeh, and S. Mostafavi, "DDoS attacks and machine-learning-based detection methods: A survey and taxonomy," Engineering Reports. 2023. doi: 10.1002/eng2.12697.

[2] S. Songma, T. Sathuphan, and T. Pamutha, "Optimizing Intrusion Detection Systems in Three Phases on the CSE-CIC-IDS-2018 Dataset," Computers, vol. 12, no. 12, p. 245, Nov. 2023, doi: 10.3390/computers12120245.

[3] Q. Zhou, R. Li, L. Xu, A. Nallanathan, J. Yang, and A. Fu, "Towards Interpretable Machine-Learning-Based DDoS Detection," SN Comput. Sci., 2024, doi: 10.1007/s42979-023-02383-y.

[4] S. Black and Y. Kim, "An Overview on Detection and Prevention of Application Layer DDoS Attacks," in 2022 IEEE 12th Annual Computing and Communication Workshop and Conference, CCWC 2022, 2022. doi: 10.1109/CCWC54503.2022.9720741.

[5] H. Lin, S. Cao, J. Wu, Z. Cao, and F. Wang, "Identifying Application-Layer DDoS Attacks Based on Request Rhythm Matrices," IEEE Access, 2019, doi: 10.1109/ACCESS.2019.2950820.

[6] S. Ramezany, R. Setthawong, and T. Tanprasert, "A Machine Learning-based Malicious Payload Detection and Classification Framework for New Web Attacks," in 2022 19th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), IEEE, May 2022, pp. 1–4. doi: 10.1109/ECTI-CON54298.2022.9795455.

[7] A. M. Vartouni, S. S. Kashi, and M. Teshnehlab, “An anomaly detection method to detect web attacks using Stacked Auto-Encoder,” in 2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), IEEE, Feb. 2018, pp. 131–134. doi: 10.1109/CFIS.2018.8336654.

[8] A. Hashim, R. Medani, and T. A. Attia, “Defences Against web Application Attacks and Detecting Phishing Links Using Machine Learning,” in 2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE), IEEE, Feb. 2021, pp. 1–6. doi: 10.1109/ICCCEEE49695.2021.9429609.

[9] B. Gogoi, T. Ahmed, and A. Dutta, “Defending against SQL Injection Attacks in Web Applications using Machine Learning and Natural Language Processing,” in 2021 IEEE 18th India Council International Conference (INDICON), IEEE, Dec. 2021, pp. 1–6. doi: 10.1109/INDICON52576.2021.9691740.

[10] A. Joshi and V. Geetha, “SQL Injection detection using machine learning,” in 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), IEEE, Jul. 2014, pp. 1111–1115. doi: 10.1109/ICCICCT.2014.6993127.

[11] L. Zhou, T. Lu, and X. Hu, “Detecting Web Application Injection Attacks Using One-Class SVM,” in 2022 IEEE 5th International Conference on Computer and Communication Engineering Technology (CCET), IEEE, Aug. 2022, pp. 275–279. doi: 10.1109/CCET55412.2022.9906382.

[12] R. Banerjee, A. Baksi, N. Singh, and S. K. Bishnu, “Detection of XSS in web applications using Machine Learning Classifiers,” in 2020 4th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech), IEEE, Oct. 2020, pp. 1–5. doi: 10.1109/IEMENTech51367.2020.9270052.

[13] G. Kaur, Y. Malik, H. Samuel, and F. Jaafar, “Detecting Blind Cross-Site Scripting Attacks Using Machine Learning,” in Proceedings of the 2018 International Conference on Signal Processing and Machine Learning - SPML ’18, New York, New York, USA: ACM Press, 2018, pp. 22–25. doi: 10.1145/3297067.3297096.

[14] S. Sharma, P. Zavarsky, and S. Butakov, “Machine Learning based Intrusion Detection System for Web-Based Attacks,” in 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), IEEE, May 2020, pp. 227–230. doi: 10.1109/BigDataSecurity-HPSC-IDS49724.2020.00048.

[15] R. Tommy, G. Sundeep, and H. Jose, “Automatic Detection and Correction of Vulnerabilities using Machine Learning,” in 2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC), IEEE, Sep. 2017, pp. 1062–1065. doi: 10.1109/CTCEEC.2017.8454995.

[16] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,” in Proceedings of the 4th International Conference on Information Systems Security and Privacy, SCITEPRESS - Science and Technology Publications, 2018, pp. 108–116. doi: 10.5220/0006639801080116.