Development of an Intrusion Detection System using an Ensemble Voting Machine Learning Technique

Ahmed Najm Abdullah
Arts, Sciences and Technology University, Lebanon
ahmednajem308@gmail.com (corresponding author)

Received: 1 March 2025 | Revised: 19 March 2025 and 6 April 2025 | Accepted: 9 April 2025

 DOI: https://doi.org/10.48084/etasr.10764

ABSTRACT

Intrusion Detection Systems (IDSs) are essential for identifying unauthorized access and malicious activities in network environments. The current study presents the development of an IDS utilizing a voting-based ensemble Machine Learning (ML) approach. Utilizing the advantages of individual ML models, the voting classifier is a well-known ML model that may enhance overall prediction performance. This study provides a unique classification method that combines the benefits of the Naive Bayes (NB), K-Nearest Neighbors (KNN), and Adaptive Boosting (AdaBoost) algorithms into a voting ensemble approach. This ensemble voting classifier greatly improves network IDS accuracy. The experiments were conducted using the KDD99 dataset. The findings reveal that the voting ensemble technique outperforms individual classifiers, achieving a higher accuracy of 99.79%.

Keywords-intrusion detection systems; ML; voting classifier

I. INTRODUCTION

The rise in internet and computer system usage due to the digital transformation of data has created issues with the security, privacy, and confidentiality of information. Efforts to protect the privacy of computer systems have merely shifted the problem elsewhere, and no existing systems are entirely secure. The range of network-based attacks has also expanded, and more sophisticated abnormal behavior patterns are being added to the attack databases for automatic engagement. Therefore, various systems have been designed and implemented for different networks to address these emerging problems. One such system is the IDS, which is vital in the field of cybersecurity because it tracks the misuse of hardware and software resources of the network [1]. However, the primary challenge is that a considerable portion of IDSs produce excessive false positives (labeling non-threatening events as possible risks), which not only creates a significant workload for security analysts, but also allows real risks to be overlooked [2, 3].

The primary function of an IDS is to assist in protecting a digital infrastructure by detecting security breaches and attempting to counter them. These systems are broadly classified based on their detection approaches, implementation locations, and specific objectives. Each category has specific benefits and drawbacks for using the system in various circumstances [4]. The most common classification is based on the detection method, involving signature-based, anomaly-based, and hybrid approaches. Signature-based intrusion detection entails comparing incoming data against a database of known attack signatures. This approach is very effective at recognizing threats and has low false positive rates; however, it is unable to identify new or evolving threats, such as zero-day attacks, since signatures must be updated regularly [4, 5]. On the other hand, anomaly-based intrusion detection emphasizes identifying deviations from normal behavior patterns. By establishing a foundation for routine activities, this method can detect unexpected or unusual behaviors that may indicate a potential threat. Anomaly-based systems are ideal for dynamic environments where attack patterns are unpredictable and have the clear advantage of identifying unknown emerging threats. But there are challenges in accurately defining "normal" behavior, which lacks a clear standard that can lead to higher rates of false positives when surveillance systems are frequently changed [6].

Hybrid intrusion detection represents a complex combination of signature and anomaly-based detection methods. This establishes a comprehensive security framework, and the integrated approach utilizes predefined attack signatures. At the same time, it employs advanced heuristics to identify deviations from the established baseline by implementing a system of components. Improved detection capabilities are realized through their complementary nature. Signature-based detection provides accurate identification of known threats, while anomaly-based detection provides adaptive capabilities to recognize new attack vectors. This collaborative combination helps organizations to maintain strong security postures against established and emerging threats [7]. IDSs can also be classified based on their areas of use, including network-based, hosted, and cloud-based.

Network Intrusion Detection Systems (NIDS) monitor network traffic and analyze data packets to identify malicious activity. They are particularly effective at identifying threats that target network infrastructure, offering a comprehensive view of the network activity. However, they encounter the challenge of encrypted traffic and may overlook host-specific attacks [8].

In contrast, Host-based Intrusion Detection Systems (HIDS) [9] monitor a single device, such as a server or workstation, Examining changes to mainframe files and resource activity. HIDS detect patterns of unauthorized activity or insider threats that were missed by NIDS, since they possess a remarkable ability to detect. However, these are limited to the devices that they observe. Cloud-based Intrusion Detection Systems (CIDS) are a vital network protection solution for cloud environments in modern time cybersecurity. CIDS are used to monitor the flow of traffic between the cloud and the hosted resources or devices that are hosted on cloud services more. While they can scale and adapt to dynamic cloud environments, they depend on the capabilities and security measures offered by third-party providers. Concerns have been raised regarding data privacy and reliance on external tools [10]. Finally, IDSs may be classified according to their mode of operation. Passive IDSs [11] primarily monitor and alert the administrator to any intrusion without taking any action. An active IDS is sometimes referred to as an Intrusion Prevention System (IPS). It takes action to prevent further intrusion. For example, it blocks suspicious traffic or isolates compromised systems. Organizations can pursue tailored and effective implementations of security strategies after understanding the different types and applications. It also ensures robust security against ever-evolving safety threats [12].

The application of ML-based classification algorithms in developing IDSs has significantly increased. Researchers have proposed various ML-based models and evaluated their effectiveness using different available public datasets [13]. These models are trained and assessed using multiple performance metrics [9, 14]. The evaluation process focuses on critical metrics, such as accuracy, recall, F1-score, and precision. This study introduces the proposed methodologies, thoroughly examined using the KDD99 datasets across these key metrics. Traditionally, classification models have been predominantly utilized a single classifier. An enhanced Support Vector Machine (SVM)-based intrusion detection technique, using feature-based training, is introduced in [15], leveraging Kullback-Leibler (KL) divergence and cross-correlation to improve accuracy, achieving a 99.47% detection rate. In [16], a random forest method combined with feature selection techniques from data mining is applied to the NSLKDD dataset, enhancing classification accuracy and processing speed with 98.88% accuracy in web attack detection. Authors in [17] present an IDS that utilizes random forest classification and Principal Component Analysis (PCA), yielding an accuracy of 96.78% on the KDCup99 dataset and outperforming methods, such as SVM and NB. In [18], a CNN model is introduced, achieving over 99% accuracy for DoS attack detection on the KDD dataset, outperforming RNNs, and obtaining 91.5% accuracy on the CSE-CIC-IDS2018 dataset. Modified parameters for DoS detection enhance the CNN model's performance. A multi-layer perception Deep Neural Network (DNN) is proposed in [19], demonstrating 98.98% and 98.99% accuracy for multi-class and binary classifications, respectively, on the KDDCup 1999 dataset, although it requires substantial training data. The model proposed in [20] employs a CNN with a regularized multi-layer perception and semi-dynamic hyperparameter tuning, achieving 95.6% multi-class classification accuracy. However, its performance on "Zero-Day" exploits is limited, and it does not account for dimension reduction or data balancing. The IE-DBN model described in [21], built on information entropy, outperforms others on the KDD CUP 99 dataset by increasing detection accuracy, reducing false alarms, minimizing convergence time, and mitigating overfitting, while also incorporating the SMOTE technique to address data imbalance.

However, in recent years, there has been a shift toward using multiple classifiers, indicating progress in the ML field. This change is a highlight of the growing understanding of the benefits of combining the strengths of different classifiers to improve predictive performance. By combining different algorithms—NB, KNN, and AdaBoost—in a voting classifier, the advantages of each model may enhance predicted accuracy, particularly when the models have distinct biases. The primary contributions of the present research are:

- It provides an ensemble classifier that enhances classification performance through the voting classifier approach. This group combines AdaBoost, KNN, and NB models.
- The performance of the proposed ensemble classifier is assessed using KDD99 datasets.
- Key evaluation metrics, such as precision, recall, F1-score, and accuracy are used to evaluate the effectiveness of the proposed approach.
- The experimental results indicate that the proposed methodology enhances performance.

II. PROPOSED APPROACH

The targeted strategy aims to improve an existing IDS by employing an ensemble learning technique. The essence of this approach is to fuse the functionality of different classifiers to obtain accurate classifications, thereby increasing the detection of intrusions within the network traffic. The proposed model is illustrated in Figure 1. Each model step is described below:

A. Dataset Description

The KDD-99 dataset [22] is widely recognized as one of the most frequently used datasets for training ML algorithms. The KDD-99 dataset is diverse and includes a total of 4.8 million instances. The dataset contains primarily categorical and integer-based attributes, totaling forty-two in number. The KDDCup99 dataset contains one class labeled as "normal" and 22 different attack types, which are categorized into four major groups, as displayed in Figure 1. These categories include:

1. Denial of Service (DoS): This involves attacks, such as mail bombs, where the goal is to disrupt the target system's ability to handle legitimate requests.

2. Remote-to-Local (R2L): This category refers to unauthorized access from a remote machine, such as through exploits, like sending mail vulnerability.
3. User-to-Root (U2R): In this category, attackers escalate their privileges from a normal user to superuser or root, often through attacks, such as buffer overflow.
4. Probing: These attacks involve activities, such as surveillance or scanning for vulnerabilities, including port scanning.

Each record in the KDDCup99 training dataset consists of 41 fixed feature attributes and a class identifier. Of these 41 attributes, nine are discrete, while the rest are continuous. To make the data suitable for intrusion detection, preprocessing operations are required. The dataset was converted for only two classes: "normal" and "attack," to be suitable for binary classification. It was then divided into two parts: training data (70%) and testing data (30%).

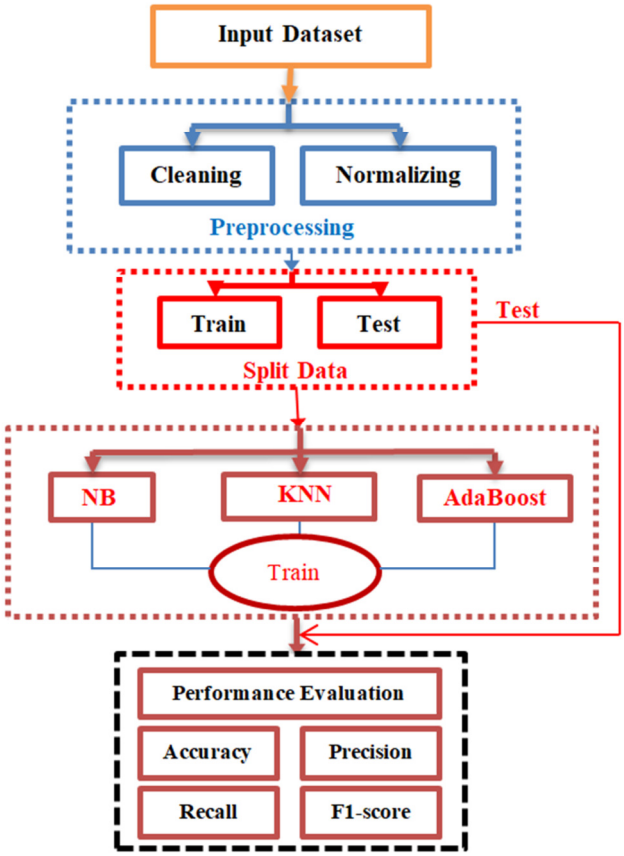

Fig. 1. Proposed ML-based IDS.

B. Preprocessing Step

Preprocessing is a crucial step in the data science pipeline, especially for ML tasks, as it directly affects the quality and performance of the data model. The first step in this technique is to collect and clean the data, which encompasses a few steps. Initially, the dataset is free of any duplicate entries to protect the quality of the data. Moving forward, the features are standardized to a common scale, enhancing the effectiveness of ML models that rely on feature magnitude. Moreover, categorical features, such as protocol type, service, and flag are converted into numeric values using label encoding to facilitate the processing of these variables by the models. The target feature, which depicts a network's traffic as normal or abnormal, is also adjusted for binary classification. Zero is assigned for all occurrences of normal traffic, and one is given to all other types of traffic, thus modifying the dataset to fit the requirements for binary classification. One-hot encoding has been applied to categorical features, while MinMax scaling has been utilized for numerical ones' attributes.

C. Machine Learning Algorithms

After preprocessing the data, the next stage involves training the ML models, as portrayed in Figure 1. The method employs three classifiers: NB, AdaBoost, and KNN. Each of these classifiers has its own unique characteristics: NB is regarded as simple and probabilistic, AdaBoost is recognized for boosting weak models, and KNN is popular for being instance-based. However, true creativity lies in the ensemble architecture's integration of classifiers through the voting classifier schema. In this type of ensemble, every classifier 'votes' for the class they believe the sample belongs to, and the final class is derived through a simple majority. The ensemble model consists of combinations of two classifiers, such as NB with AdaBoost, NB with KNN, and Naïve/NB AdaBoost with KNN, as well as a voting classifier that incorporates all three models along with the other ones.

The KNN classifier evaluates text classification by comparing the similarity and variance between a given text and its trained documents. Each of the "n" nearest neighbor data points to the new case receives a label based on their similarity level to the new point [23]. As an iterative ensemble boosting classifier known as AdaBoost, this technique serves as a fundamental method for addressing binary classification issues. The ensemble method, known as boosting, creates a strong classifier by combining the outputs of multiple weaker classifiers [24]. The NB algorithm is an ML model that uses Bayes' Theorem to perform probabilistic predictions. The NB algorithm is widely utilized in classification problems due to its simple design and effective performance in various domains, such as spam detection and sentiment analysis [25].

Voting classifier is an ensemble learning method that combines multiple ML models to improve classification performance and reduce individual model weaknesses. It operates by aggregating the predictions from multiple classifiers and selecting the final output based on a majority voting mechanism (hard voting) or the average of predicted probabilities (soft voting). Mathematically, for a given input sample x, let $C_1, C_2, \ldots, C_n$ be the classifiers used in the ensemble. The final class prediction $C_{final}(x)$ is determined as [26]:

- Hard Voting (Majority Rule):

$$C_{final}(x) = \arg max \sum_{i-1}^{n} 1[C_i(x) = c]$$

where c is a class label, $C_i(x)$ is the prediction of the ith classifier, and $1[\cdot]$ is an indicator function that equals 1 if the condition is true and 0 otherwise. The class that receives the highest number of votes is selected as the final prediction.

- Soft Voting (Probability Averaging):

$$C_{final}(x) = \arg max \sum_{i-1}^{n} w_i P[C_i = c|x]$$

where $P[C_i = c|x]$ represents the predicted probability of class c from classifier C_i, and w_i is the weight assigned to each classifier.

In this study, a hard voting classifier was utilized to integrate three different algorithms: KNN, AdaBoost, and NB. Each of these models contributes to the overall classification decision, ensuring that the final prediction benefits from their individual strengths.

D. Performance Evaluation Metrics

The evaluation of the models involves training the classifiers on part of the dataset and testing them on another portion to assess their ability to generalize. Performance is measured using several metrics, including accuracy, precision, recall, and F1-score, as presented in Figure 1. These metrics provide a comprehensive view of the classifiers' effectiveness, balancing between correctly identifying intrusions and minimizing false alarms [27, 28].

$$ACC = \frac{TP+TN}{TP+FP+TN+FN}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

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

$$F1\_score = 2 * \frac{Precision*recall}{Precision+recall}$$

III. EXPERIMENTAL RESULTS

The experiments in this study were conducted on a personal computer equipped with an Intel Core i7 processor running at 2.8 GHz, 16 GB of RAM, and operating on Windows 10 (64-bit). The implementation of the proposed models was performed using the Python programming language, version 3.9. The results clearly highlight that while individual models perform well, the true strength lies in combining them through a Voting Classifier, which leverages the advantages of each algorithm to achieve the most reliable and robust outcome predictions. The Grid Search technique was employed to optimize the classifiers' key hyperparameters. For KNN, the number of neighbors (n_neighbors) was optimized; for AdaBoost, both the number of estimators (n_estimators) and learning rate (learning_rate) were tuned; and for NB, the smoothing parameter (var_smoothing) was optimized. The final selected values of these parameters are presented in Table 1. Starting with NB, it demonstrates strong performance with 97.60% accuracy, showing its effectiveness in handling probabilistic relationships within the data. However, its reliance on the assumption of feature independence may limit its ability to capture complex patterns, which is evident in its slightly lower recall and F1-score compared to other models. AdaBoost improves upon this by achieving 98.71% accuracy, thanks to its ability to iteratively correct misclassifications and concentrate on difficult cases. Despite its strengths, AdaBoost can be sensitive to noisy data and outliers, which may impact its performance generalizability. KNN stands out with an almost perfect accuracy of 99.11%, which may appear to be the ideal choice. However, KNN is heavily dependent on the choice of k and can be computationally expensive when dealing with large datasets. It is also susceptible to overfitting, particularly when the dataset is highly structured. This means that while it performs exceptionally well in this context, its real-world performance may not always be as effective and reliable.

TABLE I. CLASSIFIER PARAMETER VALUES

Classifier	Parameter	Value
NB	priors	uniform prior
	var_smoothing	1e-9
AdaBoost	n_estimators	50
	learning_rate	0.1
KNN	n_neighbors	5
	weights	uniform
Voting Classifier	voting	hard

When combining classifiers, a noticeable improvement in accuracy and overall performance is observed. The ensemble of NB and AdaBoost achieves 97.67%, which is slightly better than NB alone, but still not as strong as other method combinations. Adding KNN to NB results in 98.29% accuracy, a further improvement, demonstrating the benefit of hybrid models. The combination of AdaBoost and KNN achieves an even higher accuracy of 98.89%, benefiting from both the adaptability of AdaBoost and the instance-based learning of KNN. However, the real breakthrough occurs when all three classifiers are combined into a Voting Classifier, achieving 99.79% accuracy. This approach leverages the strengths of each model: NB provides a solid probabilistic foundation, AdaBoost enhances decision-making through boosting, and KNN contributes instance-based learning. Together, they create a classifier that is not only highly accurate, but also more robust, balanced, and reliable than any single one model. The results presented in Table II showcase the performance of the proposed model, demonstrating its effectiveness in intrusion detection. The values for recall, precision, and F1-score have been rounded to the nearest whole number to improve clarity.

TABLE II. PERFORMANCE COMPARISON IN ALL CASES

Model	Accuracy	Precision	Recall	F1-score
NB	97.60%	98%	97%	97%
KNN	99.11%	99%	99%	99%
AdaBoost	98.71%	99%	98%	99%
NB+KNN	98.29%	99%	98%	98%
NB+ AdaBoost	97.67%	98%	97%	98%
KNN+ AdaBoost	98.89%	99%	99%	99%
KNN+NB+ AdaBoost	99.79%	100%	100%	100%

Unlike KNN, which may suffer from computational inefficiency, and AdaBoost, which is prone to overfitting on certain datasets, the Voting Classifier effectively balances precision, recall, and F1-score across all classes. This makes it the optimal choice for real-world applications, where reliability

and consistency are crucial. By leveraging the strengths of multiple classifiers, the Voting Classifier ensures that errors made by one model are corrected by the others, resulting in a highly generalized and stable predictive system. A performance comparison of the proposed approach and previous related works is illustrated in Table III. Table III offers a comprehensive evaluation of various intrusion detection techniques, comparing their effectiveness based on accuracy. Each approach, developed by different researchers, utilizes distinct methodologies and models to detect network intrusions. However, the proposed model stands out as the most effective one, achieving superior accuracy and demonstrating its robustness and reliability in intrusion detection.

TABLE III. PERFORMANCE COMPARISON WITH RELATED WORKS

Reference	Method	Accuracy
[15]	SVM with KL divergence	99.47%
[16]	RF with Gini translation	98.88%
[17]	RF with PCA translation	96.78%
[18]	CNN	99%
[19]	Multi-layer perceptron	98.99%
[20]	CNN	95.4%.
[31]	CNN and Deep Watershed Auto-encoder	98.05%
[32]	Grey Wolf Optimization algorithm and Quantum Binary Bat algorithm with RF	98.50%
Proposed	Voting Classifier	99.79%

IV. DISCUSSION

The proposed ensemble-based IDS demonstrates superior performance, combining high accuracy with low false positive rates, highlighting its suitability for real-world cybersecurity environments. By integrating NB, KNN, and AdaBoost within a voting classifier framework, the system effectively achieves a balanced trade-off between sensitivity (detection rate) and specificity (false positive control), which reduces the operational burden on security analysts.

Despite the enhanced detection capabilities, it is important to acknowledge certain limitations. Specifically, the ensemble approach introduces additional computational overhead compared to using individual classifiers. However, this trade-off is considered acceptable in high-security environments, where prediction precision and reliability are critical.

Furthermore, hard voting was selected over soft voting due to its superior stability and resilience to overfitting, particularly in high-dimensional datasets that are often encountered in network intrusion scenarios. The comparative analysis further confirms that the proposed ensemble model outperforms traditional IDS techniques, such as SVM and Random Forest, by providing a more generalized and robust prediction capability across diverse attack types and varying data distributions.

V. CONCLUSION

This study shows the tremendous power of ensemble learning in classification tasks in terms of accuracy and robustness. Although Naïve Bayes (NB), Adaptive Boosting (AdaBoost), and K-Nearest Neighbors (KNN) perform well, they all have certain limitations. Some of these include probabilistic assumptions, sensitivity to noise, and computational inefficiency. Their performance in real-life applications is always a question. Predictive performance has improved; it has been shown that using more than one classifier yields better results. Combining classifiers outperformed any hybrid models, particularly NB with AdaBoost or KNN, which tend to provide a distinct advantage over single models. However, the most outstanding improvement is provided by the Voting Classifier that combines all three, achieving 99.79% accuracy, which is higher than any single hybrid model. This demonstrates the strength of ensemble learning in utilizing varied algorithms to build a more robust and generalized predictive model.

Overall, the findings of this study emphasize that ensemble methods, specifically Voting Classifiers, offer the most reliable and efficient approach to complex classification tasks. Future research will extend evaluation to modern datasets, like NSL-KDD [29] and CIC-IDS2017 [30], integrate additional Machine Larning (ML) models, explore feature selection techniques, and investigate adaptive voting strategies to further enhance performance. These refinements will ensure the Intrusion Detection System (IDS) remains effective in addressing sophisticated cyber threats.

REFERENCES

[1] R. H. Altaie and H. K. Hoomod, "An Intrusion Detection System using a Hybrid Lightweight Deep Learning Algorithm," *Engineering, Technology & Applied Science Research*, vol. 14, no. 5, pp. 16740–16743, Oct. 2024, https://doi.org/10.48084/etasr.7657.

[2] D. E. Denning, "An Intrusion-Detection Model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222–232, Oct. 1987, https://doi.org/10.1109/TSE.1987.232894.

[3] J. Xu and C. R. Shelton, "Intrusion Detection using Continuous Time Bayesian Networks," *Journal of Artificial Intelligence Research*, vol. 39, pp. 745–774, Dec. 2010, https://doi.org/10.1613/jair.3050.

[4] O. H. Abdulganiyu, T. Ait Tchakoucht, and Y. K. Saheed, "A systematic literature review for network intrusion detection system (IDS)," *International Journal of Information Security*, vol. 22, no. 5, pp. 1125–1162, Oct. 2023, https://doi.org/10.1007/s10207-023-00682-2.

[5] Amarudin, R. Ferdiana, and Widyawan, "A Systematic Literature Review of Intrusion Detection System for Network Security: Research Trends, Datasets and Methods," in *2020 4th International Conference on Informatics and Computational Sciences (ICICoS)*, Semarang, Indonesia, Nov. 2020, pp. 1–6, https://doi.org/10.1109/ICICoS51170.2020.9299068.

[6] V. Jyothsna and V. V. R. Prasad, "A Review of Anomaly based Intrusion Detection Systems," *International Journal of Computer Applications*, vol. 28, no. 7, pp. 26–35, Aug. 2011.

[7] E. M. Maseno, Z. Wang, and H. Xing, "A Systematic Review on Hybrid Intrusion Detection System," *Security and Communication Networks*, vol. 2022, no. 1, 2022, Art. no. 9663052, https://doi.org/10.1155/2022/9663052.

[8] S. Kumar, S. Gupta, and S. Arora, "Research Trends in Network-Based Intrusion Detection Systems: A Review," *IEEE Access*, vol. 9, pp. 157761–157779, 2021, https://doi.org/10.1109/ACCESS.2021.3129775.

[9] H. Satilmiş, S. Akleylek, and Z. Y. Tok, "A Systematic Literature Review on Host-Based Intrusion Detection Systems," *IEEE Access*, vol. 12, pp. 27237–27266, 2024, https://doi.org/10.1109/ACCESS.2024.3367004.

[10] H. Alavizadeh and H. Alavizadeh, "Cloud-Based Intrusion Detection System Using a Deep Neural Network and Human-in-the-Loop Decision

Making," in *Deep Learning for Multimedia Processing Applications*, pp. 270-284, CRC Press, 2024.

[11] W. T. Yue and M. Çakanyıldırım, "A cost-based analysis of intrusion detection system configuration under active or passive response," *Decision Support Systems*, vol. 50, no. 1, pp. 21–31, Sep. 2010, https://doi.org/10.1016/j.dss.2010.06.001.

[12] Q.-V. Dang, "Active Learning for Intrusion Detection Systems," in *2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*, Ho Chi Minh City, Vietnam, Oct. 2020, pp. 1–3, https://doi.org/10.1109/RIVF48685.2020.9140751.

[13] S. Mukherjee and N. Sharma, "Intrusion Detection using Naive Bayes Classifier with Feature Reduction," *Procedia Technology*, vol. 4, pp. 119–128, Jan. 2012, https://doi.org/10.1016/j.protcy.2012.05.017.

[14] S. S. Sivatha Sindhu, S. Geetha, and A. Kannan, "Decision tree based light weight intrusion detection using a wrapper approach," *Expert Systems with Applications*, vol. 39, no. 1, pp. 129–141, Jan. 2012, https://doi.org/10.1016/j.eswa.2011.06.013.

[15] Y. Zhang, Q. Yang, S. Lambotharan, K. Kyriakopoulos, I. Ghafir, and B. AsSadhan, "Anomaly-Based Network Intrusion Detection Using SVM," in *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*, Xi'an, China, Oct. 2019, pp. 1–6, https://doi.org/10.1109/WCSP.2019.8927907.

[16] P. Negandhi, Y. Trivedi, and R. Mangrulkar, "Intrusion Detection System Using Random Forest on the NSL-KDD Dataset," in *Emerging Research in Computing, Information, Communication and Applications*, Singapore, 2019, pp. 519–531, https://doi.org/10.1007/978-981-13-6001-5_43.

[17] S. Waskle, L. Parashar, and U. Singh, "Intrusion Detection System Using PCA with Random Forest Approach," in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India, Jul. 2020, pp. 803–808, https://doi.org/10.1109/ICESC48915.2020.9155656.

[18] J. Kim, J. Kim, H. Kim, M. Shim, and E. Choi, "CNN-Based Network Intrusion Detection against Denial-of-Service Attacks," *Electronics*, vol. 9, no. 6, Jun. 2020, Art. no. 916, https://doi.org/10.3390/electronics9060916.

[19] M. Maithem and G. A. Al-sultany, "Network intrusion detection system using deep neural networks," *Journal of Physics: Conference Series*, vol. 1804, no. 1, Oct. 2021, Art. no. 012138, https://doi.org/10.1088/1742-6596/1804/1/012138.

[20] L. Ashiku and C. Dagli, "Network Intrusion Detection System using Deep Learning," *Procedia Computer Science*, vol. 185, pp. 239–247, Jan. 2021, https://doi.org/10.1016/j.procs.2021.05.025.

[21] H. Jia, J. Liu, M. Zhang, X. He, and W. Sun, "Network intrusion detection based on IE-DBN model," *Computer Communications*, vol. 178, pp. 131–140, Oct. 2021, https://doi.org/10.1016/j.comcom.2021.07.016.

[22] T. Jamal, "KDD99 dataset." kaggle, [Online]. Available: https://www.kaggle.com/datasets/toobajamal/kdd99-dataset.

[23] K. Taunk, S. De, S. Verma, and A. Swetapadma, "A Brief Review of Nearest Neighbor Algorithm for Learning and Classification," in *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, Madurai, India, May 2019, pp. 1255–1260, https://doi.org/10.1109/ICCS45141.2019.9065747.

[24] Y. Cao, Q.-G. Miao, J.-C. Liu, and L. Gao, "Advance and Prospects of AdaBoost Algorithm," *Acta Automatica Sinica*, vol. 39, no. 6, pp. 745–758, Jun. 2013, https://doi.org/10.1016/S1874-1029(13)60052-X.

[25] S. Chen, G. I. Webb, L. Liu, and X. Ma, "A novel selective naïve Bayes algorithm," *Knowledge-Based Systems*, vol. 192, Mar. 2020, Art. no. 105361, https://doi.org/10.1016/j.knosys.2019.105361.

[26] H. G. Jabbar, "Advanced Threat Detection Using Soft and Hard Voting Techniques in Ensemble Learning," *Journal of Robotics and Control (JRC)*, vol. 5, no. 4, pp. 1104–1116, Jun. 2024, https://doi.org/10.18196/jrc.v5i4.22005.

[27] C. J. Needham and R. D. Boyle,"Performance Evaluation Metrics and Statistics for Positional Tracker Evaluation," in *Computer Vision Systems*, Berlin, Heidelberg, 2003, pp. 278–289, https://doi.org/10.1007/3-540-36592-3_27.

[28] O. Aydemir, "A New Performance Evaluation Metric for Classifiers: Polygon Area Metric," *Journal of Classification*, vol. 38, no. 1, pp. 16–26, Apr. 2021, https://doi.org/10.1007/s00357-020-09362-5.

[29] L. Dhanabal and S. P. Shantharaja, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms." *International journal of advanced research in computer and communication engineering*, vol. 4, no. 6, pp. 446-452, Jun 2015, https://doi.org/10.17148/IJARCCE.2015.4696.

[30] A. K. Samha, N. Malik, D. Sharma, K. S, and P. Dutta, "Intrusion Detection System Using Hybrid Convolutional Neural Network," *Mobile Networks and Applications*, Aug. 2023, https://doi.org/10.1007/s11036-023-02223-6.

[31] "Intrusion detection evaluation dataset (CIC-IDS2017)." Canadian Institute for Cybersecurity, 2017, 2025. [Online]. Available: https://www.unb.ca/cic/datasets/ids-2017.html.

[32] M. Alotaibi *et al.*, "Hybrid GWQBBA model for optimized classification of attacks in Intrusion Detection System," *Alexandria Engineering Journal*, vol. 116, pp. 9–19, Mar. 2025, https://doi.org/10.1016/j.aej.2024.12.057.