Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH Open Access

Explainable AI-based innovative hybrid ensemble model for intrusion detection

Usman Ahmed[1], Zheng Jiangbin[1], Ahmad Almogren[2], Sheharyar Khan[1], Muhammad Tariq Sadiq[3,4], Ayman Altameem[5] and Ateeq Ur Rehman[6*]

Abstract

Cybersecurity threats have become more worldly, demanding advanced detection mechanisms with the exponential growth in digital data and network services. Intrusion Detection Systems (IDSs) are crucial in identifying illegitimate access or anomalous behaviour within computer network systems, consequently opposing sensitive information. Traditional IDS approaches often struggle with high false positive rates and the ability to adapt embryonic attack patterns. This work asserts a novel Hybrid Adaptive Ensemble for Intrusion Detection (HAEnID), an innovative and powerful method to enhance intrusion detection, different from the conventional techniques. HAEnID is composed of a string of multi-layered ensemble, which consists of a Stacking Ensemble (SEM), a Bayesian Model Averaging (BMA), and a Conditional Ensemble method (CEM). HAEnID combines the best of these three ensemble techniques for ultimate success in detection with a considerable cut in false alarms. A key feature of HAEnID is an adaptive mechanism that allows ensemble components to change over time as network traffic patterns vary and new threats appear. This way, HAEnID would provide adequate protection as attack vectors change. Furthermore, the model would become more interpretable and explainable using Shapley Additive Explanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME). The proposed Ensemble model for intrusion detection on CIC-IDS 2017 achieves excellent accuracy (97-98%), demonstrating effectiveness and consistency across various configurations. Feature selection further enhances performance, with BMA-M (20) reaching 98.79% accuracy. These results highlight the potential of the ensemble model for accurate and reliable intrusion detection and, hence, is a state-of-the-art choice for accuracy and explainability.

Keywords Stacking ensemble, Bayesian model averaging, Conditional ensemble method, Machine learning, Explainable AI, Network security, Intrusion detection

*Correspondence:
Ateeq Ur Rehman
202411144@gachon.ac.kr
[1] School of Software, Northwestern Polytechnical University, Xian 710072, China
[2] Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11633, Saudi Arabia
[3] School of Computer Science and Electronic Engineering, University of Essex, Colchester Campus, Colchester, UK
[4] Applied Science Research Center, Applied Science Private University, Amman, Jordan
[5] Department of Natural and Engineering Sciences, College of Applied Studies and Community Services, King Saud University, Riyadh 11543, Saudi Arabia
[6] School of Computing, Gachon University, Seongnam-si 13120, Republic of Korea

Introduction

Intrusion Detection Systems (IDSs) have long been one of the primary defenses against hacking and other unauthorized access, enabling users to identify abnormal behaviour and prevent malicious activities or penetrations of their network environments [1]. Yet traditional or network-based IDSs are widely recognized as suffering from various fundamental weaknesses that limit their effectiveness. Among the most persistent problems are high rates of false positives -situations where benign activities are misidentified as malicious - which tend to generate excessive numbers of unrelated alerts, consuming scarce analyst time and attention and thus diminishing the overall efficiency of security teams. In a worst-case scenario, this noise might delay response to real threats [2].

We found that the old IDS had difficulties recognizing advanced and ever-changing cyber threats. The old methods of IDSs are challenging and complex to adapt to the rapidly changing cyber-attack scenarios. The old methods depend on signature-based detection techniques to handle unrecognized zero-day attacks. This limitation of the detection technique can result in many false positive alarms, which causes false alerts and makes it more difficult for system administrators to pinpoint the real threats. Moreover, in particular scenarios, most IDSs face performance issues involving extensive network system traffic [3].

Modern IDSs are critical for preventing network breaches by identifying unauthorized access and uncommon behaviour. However, today's IDSs have several limitations. Firstly, the false positive rate is extremely high, where normal network activities trigger many alerts. Given these systems' tremendous number of alerts, security personnel cannot focus on real threats. A second reason for the low performance of existing IDSs is that many are signature-based and can only detect patterns that they have previously been trained on. In other words, for all the logic contained within the IDS, it might not pick up on novel or mutated attack patterns that were not incorporated into the sample data it was trained against [4].

Another major shortcoming of existing IDS models is this lack of dynamic adaptation in the presence of novel attack patterns. Since threats in the cyber world are dynamic - attackers amend and develop new attack scripts to breach the firewalls - many IDS models in the present context still rely on simple rule-based or signature-based attack detection [5]. These models remain vulnerable to attacks that have never been seen before. Attackers can exploit such vulnerabilities to launch novel attacks. Once attackers discover a method to evade an IDS, the IDS becomes ineffective until an administrator reformulates a new attack rule or updates signature detection. This adaptiveness nightmare exposes networks to novel attacks that could be developed in the future, which presents yet another concerning gap to be addressed by improved IDS models [6].

Moreover, many current IDS approaches are opaque in a way that is incompatible with the need for accountability and explainability. The 'black-box' nature of traditional IDS models presents a problem in decision-making contexts where a precise understanding of the reasoning behind alerts and classifications is essential. Unless a system can explain how it reached a particular conclusion, we cannot be confident that it is flagging what we, human experts, would consider suspicious. This unpredictability creates problems for the operational use of IDS and the iterative process of refining detection models over time [7].

Nowadays, maintaining a security system is difficult due to the fast development of networking technologies and the increasing number of cyber-attacks. IDSs are essential now for examining network system traffic and potential security threats. Machine Learning models should be integrated into the development of IDSs. Machine learning models have proven efficient and effective across various fields, such as natural language processing, computer vision, and pattern recognition. By integrating machine learning models, IDSs can benefit from improved precision and efficiency [8].

Meanwhile, increased network traffic and exponentially larger data scale make many IDS solutions fall behind in performance and scalability, leading to false or delayed intrusion detection. A core shortcoming is the lack of explainability, as many IDSs operate as 'black boxes', making the suspicious patterns nearly impossible to inspect or making the detection results hard to know. The opacity compromises the confidence of the security experts in the alerts and adds complexity to the investigation and response process [9].

After the global surge in connected devices, remote collection of real-time data from physical objects is enabled. This data is essential for developing complex algorithms that facilitate the intelligent decision-making system, which will be helpful for the Internet of Things (IoT) environments, [10]. The extensive deployment of remote devices in the real world increases cybersecurity threats. Malicious devices attached remotely to tamper traffic signals can disrupt network system performance. The most significant real-time cybersecurity challenges are Distributed Denial-of-Service (DDoS) attacks, the Mirai botnet attack, the denial-of-service (DoS) attacks, Port Scans, and website sabotage by botnet. Guarding IoT devices against such attacks is paramount in the security domain. Consequently, there's a pressing need for proactive measures to ensure physical and cybersecurity to counteract these formidable threats, underscoring the importance of thorough network system protection analysis [11].

Security attacks were classified into active and passive. Active attacks disrupt operations and can cause harm to physical devices during their execution. They are more challenging to carry out and detect than passive attacks. Examples of active attacks include DoS [12], which is particularly prevalent as packet replay, spoofing, and message modification. On the other hand, passive attacks involve monitoring and collecting data from a target without altering the information. Attackers remain concealed, maintaining access to the communication channel to gather data. The most frequent passive attacks include eavesdropping, network system mapping, and traffic analysis. Implementing a real-time network anomaly-based IDS is crucial to alleviate the effects of attacks on IoT devices and consumers that can identify and block these threats [13].

Furthermore, many IDSs are still built for binary classification. Still, it is difficult for them to tackle multi-class classification problems, especially in complex network environments where multiple types of attacks need to be correctly classified. All these gaps point to the necessity of developing a new model of IDS that overcomes these limitations in the future and can handle varied new attack patterns more accurately and reliably. That's where our proposed model comes in: the Hybrid Adaptive Ensemble for Intrusion Detection (HAEnID) model. We tackle the gaps by influencing the state-of-the-art ensemble methods, including Stacking Ensemble (SEM), a Bayesian Model Averaging (BMA), and a Conditional Ensemble method (CEM), as well as Explainable AI techniques such as Shapley Additive Explanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME) to improve the detection accuracy, adaptability in evolving patterns, and transparency of explanation [14].

Overall, our proposed model HAEnID for intrusion detection is an elegant solution that can mitigate the high false positive rates, provide a robust solution in detecting evolutions of attack patterns, and ensure the system remains transparent and trustable.

This proposed model, HAEnID, represents a significant advancement in intrusion detection technology, promising enhanced security for network systems through its innovative ensemble learning techniques and Explainable AI (SHAP, LIME) for model interpretability. Flow chart of the basic working model for the HAEnID system as shown in Fig. 1.

- The structure-based SEM utilizes distinct base classifiers-Decision Trees (DT), Random Forests (RF), Multi-Layer Perceptron (MLP), Logistic Regression(LR), Light gradient boosting framework (LGBM), and Adaptive Boosting(AdB) -trained on CIC-IDS2017 data subsets.
- BMA assigns weights to classifiers based on performance and prediction uncertainty. By estimating posterior probabilities, BMA integrates outputs robustly.
- CEM dynamically selects relevant classifiers, enhancing adaptability to evolving data patterns and improving intrusion detection efficacy.
- Evaluation of benchmark datasets demonstrates superior performance over traditional methods, affirming the ensemble's potential to fortify network system defences against cyber threats.
- We also implement and visually enhance the output of the proposed HAEnID model with 3D visualization in MATLAB.

Related work

One of the most recent trends in IDS is to instrument Explainable AI to improve the transparency and interpretability of these systems. Over the years, the models used in IDS became increasingly complex until they reached models that were hard to interpret, based on machine learning and deep learning, which are considered 'black-box' models. Now, scientists working on IDS are turning to Explainable AI to allow these high-performing models to become interpretable. For example, we see the usage of post-hoc explanation models such as SHAP and LIME, which help security analysts interpret complex models' decisions. These techniques tell the analyst which features most motivated the IDS model to detect an intrusion, making these uninterpretable models more transparent and trustworthy to the security analyst [15].

Various works have been done using modeling through agent technology; Labiod [16] has proposed an approach for intrusion detection based on intelligent agents. They have used the concept of Distributed Artificial Intelligence for intrusion detection. They have categorized the network attacks into two types: 1) Internal and 2) External attacks. They have two intrusion detection methods: behavior-based intrusion detection and knowledge-based intrusion detection. They have proposed their architecture for intrusion detection structure and defined their own set of agents in it. For instance, an attempt has been made to create a system based on multi-agents and wireless sensors. In [17], a hierarchical distributed IDS has been proposed. It works with the firewall and the network system management tool to give threefold security. The reason behind such a mechanism is that sometimes the virus updates are not up-to-date, and the packet bypasses the firewall. The system is based on MAS so that it can act independently. They use advanced techniques for intrusion detection and can communicate with each other. These measures make it a very much improved solution for network security protection.

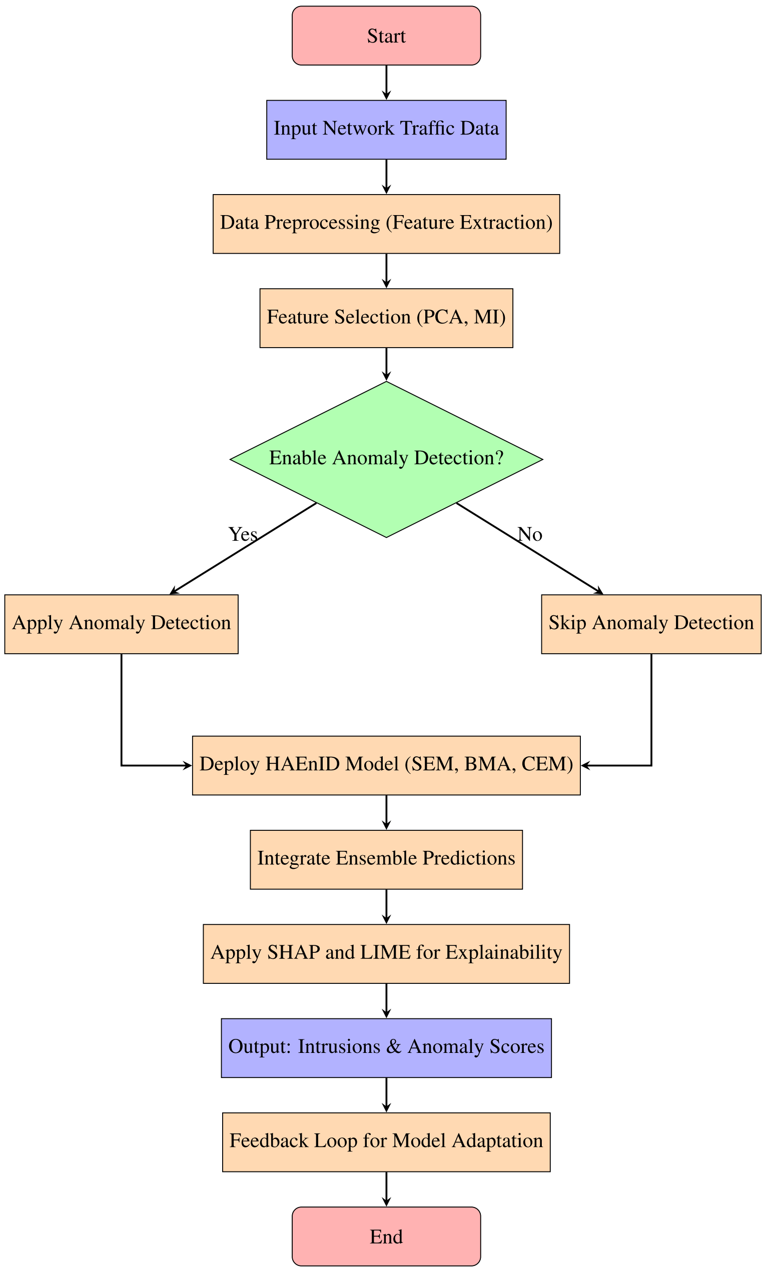

Start
Input Network Traffic Data
Data Preprocessing (Feature Extraction)
Feature Selection (PCA, MI)
Enable Anomaly Detection?
Yes
No
Apply Anomaly Detection
Skip Anomaly Detection
Deploy HAEnID Model (SEM, BMA, CEM)
Integrate Ensemble Predictions
Apply SHAP and LIME for Explainability
Output: Intrusions & Anomaly Scores
Feedback Loop for Model Adaptation
End

Fig. 1 Flow chart of the basic working model for the (HAEnID) system

One of the earliest works in this area is the integration of SHAP values with machine learning-based IDS models. SHAP explains the predictions of IDS models in a unified measure of feature importance that is consistent across different models. For example, Antwarg [18] deployed SHAP values to describe the decisions of ensemble models in IDS - specifically, why some network features help detect intrusions. Similarly, LIME has been utilized to provide local explanations for individual predictions to understand why that instance was explicitly identified as an attack. This explains-an-alert capability is as important as the detection, and its addition when using these Explainable AI techniques in IDS has helped to make machine learning model predictions more interpretable and to mitigate any subtle bias of training data, which can be leveraged to build better and more equitable models.

Researchers have explored various machine learning strategies in the initial phases of integrating Artificial Intelligence (AI) within IoT-based IDSs. While machine learning techniques offer improved accuracy and address many deficiencies in conventional approaches, they have limitations. Among machine learning methodologies, the Classification and Regression Tree (CART) is notably effective, providing high performance with minimal training time. However, its efficacy diminishes when handling complex datasets [19]. Traditional machine learning approaches often rely on shallow learning, emphasizing feature engineering and selection, but they possess limited learning capabilities, making them less effective for intricate datasets. The learning process in these methods only captures fragmentary information from each piece of data, necessitating a large dataset for practical training. This requirement becomes particularly critical when dealing with diverse datasets. In contrast, deep learning is crucial when managing vast amounts of data, offering automatic feature learning and the capacity to tackle sophisticated issues with substantial datasets [20].

Kussul et al. [21] have proposed an intrusion and anomaly detection system based on agent systems. The system identifies known attacks and anomalies in user activity. Neural networks have been employed to ward off attacks, while different agents provide intelligence features. The author gave an exciting insight into when to use MAS technology. Also, they have considered past and present works and how the existing technologies address the security issue in multi-agent systems (MAS). Hedin et al. [22] have chosen Gaia as a design methodology and incorporated some security requirements. Also, ensuring trust and reliability between nodes cooperating to reach the means end.

In [23], Talib et al. proposed designing a MAS architecture for Cloud Data Storage (CDS) using an agent-oriented methodology called Prometheus. They proposed various agents who act independently and display security and intelligent behaviour. They observed a lot of complexity with the methods and found that many dependencies were created, and the interfaces were not simple enough to design a complex problem. In [24], Sahai et al. proposed a heterogeneous distributed system, a dynamic architecture for network system management. The author's system uses the prevalent architectures and develops their architecture based on mobile agents, which is more scalable, decentralized, and has more managers; in their opinion, it creates fewer bottlenecks.

Ensemble-based [25] IDSs have received significant attention, as they can improve detection accuracy and reduce false alarm rates by incorporating multiple machine learning models. One of the prominent techniques is SEM learning, which combines the outputs of several base classifiers to produce the final predictions. This way, the final model is supposed to take advantage of the strength of each classifier while losing little from its weaknesses. For example, a recent study presented a stacking ensemble learning-based IDS that combines the predictions of three base classifiers: RF, DT, and k-NN. When the proposed system is trained using the UNSW-NB15 dataset, it achieves better accuracy and robustness in detecting network intrusions. One of the research papers revealed the power of the ensemble approach employed in computer systems for intrusion detection. They used RF, GB, AdB, and XGB algorithms and observed that they often achieve good precision, recall, and F1 scores, with perfect performance metrics up to 99% [26].

BMA has been considered to tackle the uncertainties related to the model selection problem by averaging a set of models weighted by their posterior probabilities. It minimizes the risks due to the choices made in poor model selection and, therefore, improves the reliability of the IDS. An ensemble-based IDS using LR, NB, and DT classifiers with a voting scheme was applied to IoT networks and reported significant improvements in accuracy and false alarm rates when benchmarked against the dataset of the CIC-IDS2017 contest [27].

Finally, CEM selects and weighs the contributions of models adaptively depending on the input conditions. For instance, a CEM learning approach for the security of the IoT device used feature selection methods and combined multiple supervised models, including RF, DT, LR, and k-NN, to improve the detection performance. They collectively contribute to a more solid and accurate IDS through differences in the strengths of classifiers and fading the possibility of false negatives and positives [28].

Nevertheless, there are still open problems associated with Explainable AI used in the context of IDS. For instance, there is a trade-off between explainability and model complexity. Some models, such as DT and RF, are inherently more interpretable than others (e.g., DNN). On the other hand, they tend to be less accurate and less prone to generalization compared to deep learning models. This problem has been explored in recent years with hybrid approaches [29] combining the two interpretable and complex methods. For instance, hybrid models that combine DT with deep learning architectures can be found where the DT module explains the model's decision at a high level, and the deep learning module ensures a high detection accuracy, thus allowing for a trade-off between interpretability and performance and enabling IDS models to be deployed that are effective and explainable [30].

Along with these hybrid models, people have been developing intrinsically interpretable models specifically for IDS [31], which allow for transparency in and of themselves rather than as post-hoc explanations. Attention mechanisms in deep learning models, for example, enable analysts to perceive where a model is looking in a literal sense - where the model's attention is concentrated when acting on input data features. So, for instance, a deep-learning IDS might have its attention drawn to anomalous network traffic features in its input, which an analyst can then perceive visually. Other recent works have explored the use of causal inference in IDS to determine which events in a network 'cause' intrusion to happen. This knowledge can help us build more interpretable intrusion models, both as 'detectors', and in showing exactly how and why an intrusion will occur. These advances demonstrate the ongoing evolution in making IDS more interpretable, aiming to keep this class of systems as sound as possible in the face of persistent advances in the cyber threat landscape [32].

Related work and their contribution are as under Table 1. The proposed HAEnID model has brought up numerous remarkable novelties and essential differences in contrast to the related works in intrusion detection.

One of the critical advances of the HAEnID model is detection accuracy. The accuracy of the parts (SEM, BMA, and CEM) in the HAEnID model were individually 97.44%, 97.94%, and 97.25% respectively. Although in the studies mentioned above, such as the 2023 study [33] on using RF and SVM, the present accuracy of each single model was up to 99.9%, HAEnID's advantage is that the hybrid adaptive ensemble approach enables considerable accuracy to be maintained, resulting in a more balanced and thorough detection system by combining several advanced methods.

Another critical difference in terms of adaptability is that the hybrid ensemble approach taken in HAEnID

Table 1 Related work and their contribution

Ref	Method	Description	Result	Contribution
2023 [33]	RF and SVM	Anomaly detection using RF and SVM in IoT devices.	RF - 99.9% SVM - 97.9%	The study uses ML models for anomaly detection and achieved good results.
2021 [34]	LR and ANN	Using LR and ANN to prevent unauthorized access to network devices.	LR - 99.9% ANN - 95.13%	Proposed model achieved F1-score of 95.13%, and LR achieved 99.9% accuracy with feature selection.
2023 [35]	Linear R, Logistic R, SVM, KNN	Machine learning models were used to detect botnet attacks.	Linear R - 97.8% Logistic R - 97.76% SVM - 97.85% KNN - 98.3%	Proposed model achieved good accuracy in detecting botnet attacks in IoT network system.
2021 [36]	NB, SVM, LR, k-NN, Adb, RF	Using machine learning models to detect abnormal traffic in network systems.	NB - 90.22% SVM - 97.66% LR - 96.83% k-NN - 97.17% Adb - 98.30% RF - 98.97%	Proposed model successfully identified abnormalities in network systems.
2024 [37]	DNN	Used neural network to detect cyberthreats in metaverse.	DNN (9 classes) - 87.74% DNN (2 classes) - 99.9%	Proposed model achieved good performance score using UNSWNB15 dataset.
2021 [38]	CatBoost	CatBoost for handling categorical featuresh	CB-99.46%	Effective machine learning approach to enhance cybersecurity in network systems.
2021 [39]	J48, NB, RF, RTree, REP Tree	training (296,412 instances) and testing (98,804 instances)	RF-CCR-99.97%	Effectiveness of various IDS algorithms for different attack types.
2022 [40]	AdB, RF, LR	Voting ensemble approach	VE-99.86%	Improved detection of novel attacks with higher accuracy and lower false alarms.
Our paper	SEM, BMA, CEM	(HAEnID), frameworks to enhance detection accuracy and reduce false alarms.	SEM - 97.44% BMA - 98.01% CEM - 98.05%	(HAEnID) model aims to provide a robust and adaptive solution for real-time intrusion detection.

is quite different from those seen in the related works where the IDSs are entirely based, respectively, on structure-based modeling or an ensemble of machine learning models such as LR, SVM, and kNN [36]. The HAEnID model combines structure- and ensemble-based intrusion detection frameworks, making it more adaptive in changing environments and a more comprehensive set of cyber threats. The crucial ability of the system to adapt in time to a broader set of attack vectors can dramatically improve its effectiveness, especially if we consider the real-time intrusion detection problem.

The related works show different machine learning models such as CatBoost, ensemble models AdB, RF, LR [40], and DT models J48, NB, RF, RTree [39] to enhance the detection of network system threats and anomalies. CatBoost [38] handled the data using categorical features with an accuracy of 99.46%. The voting ensemble had an accuracy of 99.86%. Also, using DT models showed that it could handle a large dataset with an accuracy of 99.97%. In comparison to the existing studies, imputation models were used, such as SEM, BMA, and CEM, to obtain a high accuracy rate (97.44-98.05%), besides significantly enhancing the detection accuracy with the lowest alarm rate for our proposed HAEnID. This shows that these models can achieve high accuracy from our study. We also enhance the adaptability of the models and make them robust with time in detecting and alarming cyber threats.

Furthermore, from the perspective of false alarms, one of the significant issues of IDSs, the hybrid nature of HAEnID, which utilizes multiple ensemble techniques, is expected to be highly accurate in detecting intruders while generating the least possible number of false alarms. While the related works (i.e., the models from 2021 and 2023) possessed a high level of accuracy, they did not explicitly mention the problem of false alarms. These results imply that although these models were highly influential in the context where they were built and tested, they would not be as effective in reducing false alarms in diverse settings - such as that examined by us.

Finally, the generalizability of the HAEnID model offers a clear differentiation from other works, which mainly concentrate on a specific application. HAEnID's hybrid adaptive ensemble technique proved effective in several application fields concerning accuracy and robustness; therefore, it is domain-independent, which is not valid for models with a specific application. This property is a crucial advantage of HAEnID during deployment, which will be more flexible concerning a more general class of cybersecurity than that addressed by domain-specific, narrow-focused models. This superiority regarding high accuracy, generalizability, low false alarms, and broader applicability is evident and appealing compared to the methods used in the related work.

Proposed methodology

The proposed methodology as shown in Fig. 2 integrates advanced ensemble learning techniques to enhance the detection accuracy and adaptability of IDSs. Our approach was structured around three core components: SEM, BMA, and CEM, as shown in Algorithm 1. The proposed HAEnID algorithm, as shown in Algorithm 2 is re-designed and modified version to facilitate the strengths of different classifiers' nature to improve the overall robustness and performance of the system.

The SEM as shown in Fig. 3 is a potent and sophisticated technique in ensemble learning that combines predictions produced by multiple base classifiers to improve the overall performance of the predictive model. In the HAEnID model, SEM involves training multiple base classifiers - such as DT and RF, MLPs (Deep Learning), and LGBM - on the same dataset. Each of these classifiers separately predicts using the data supplied to it. Their predictions are then fed as inputs to the meta-classifiers. This meta-classifier then learns to give the final output by optimally combining the output of base classifiers. This method allows HAEnID to use the benefits of the individual models' characteristics and overcome the issues of variance and bias, leading to a more accurate and reliable IDS that can identify diverse attack scenarios.

BMA as shown in Fig. 4 is a statistical technique that accounts for model uncertainty by averaging across a set of models rather than selecting the best-performing model. In HAEnID, BMA plays a critical role by considering each model's posterior probabilities (e.g., a probabilistic model of a DDoS attack). These probabilities can be interpreted as weights of each of the considered models given the observed data, and they are used to weigh a combination of the individual predictions of each considered model into a single average prediction. With the help of BMA, HAEnID can feel uncertainty and variability in the data, making the IDS more robust - particularly in the presence of novel or emerging attack patterns. Using BMA makes the model's predictions less dependent on the (mostly unknown) assumptions behind each model's parameters, resulting in more accurate detection (fewer false positives) and more resilient IDSs.

The CEM, as shown in Fig. 5, is designed to take advantage of ensemble learning's dynamic nature by conditioning models' selection and weighting on specific input data features. This allows HAEnID to adapt its ensemble of models to the features of particular inputs (for instance,

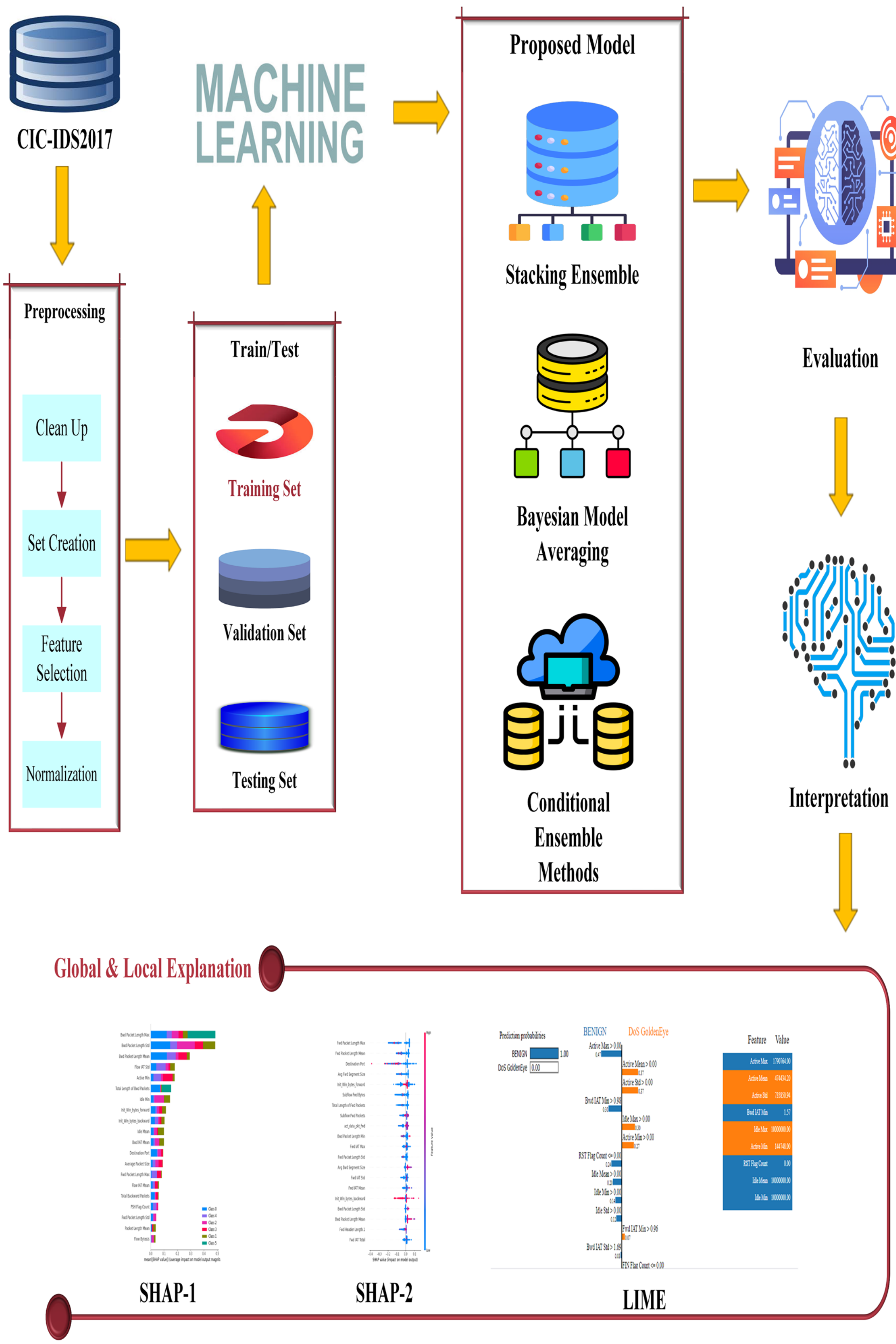


Fig. 2 Proposed HAEnID model, built with the collective intelligence of multiple machine-learning models through a unique ensemble approach

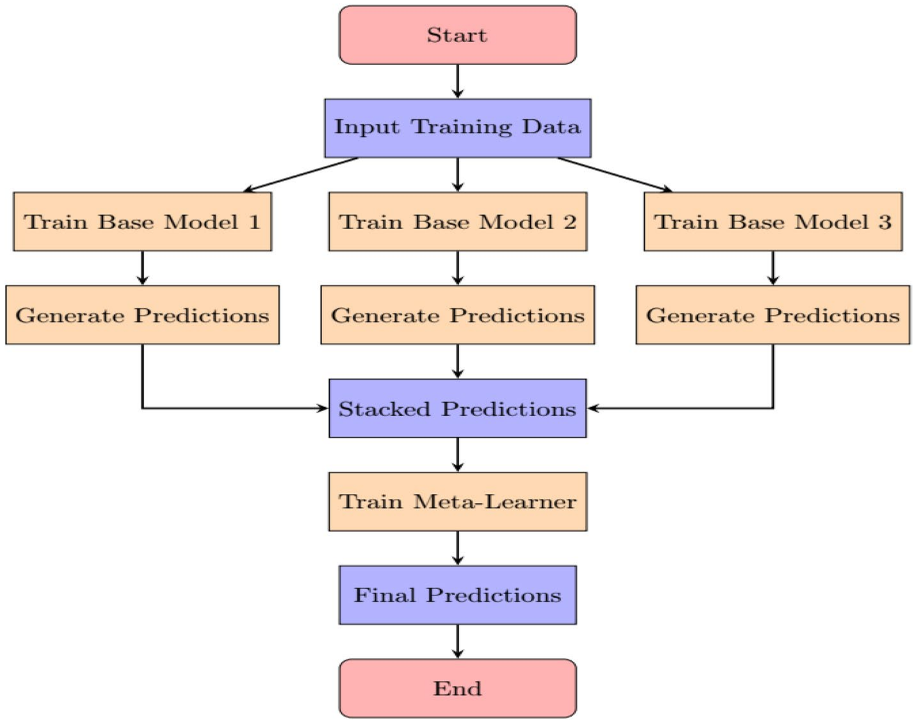


Fig. 3 Flowchart of the SEM model

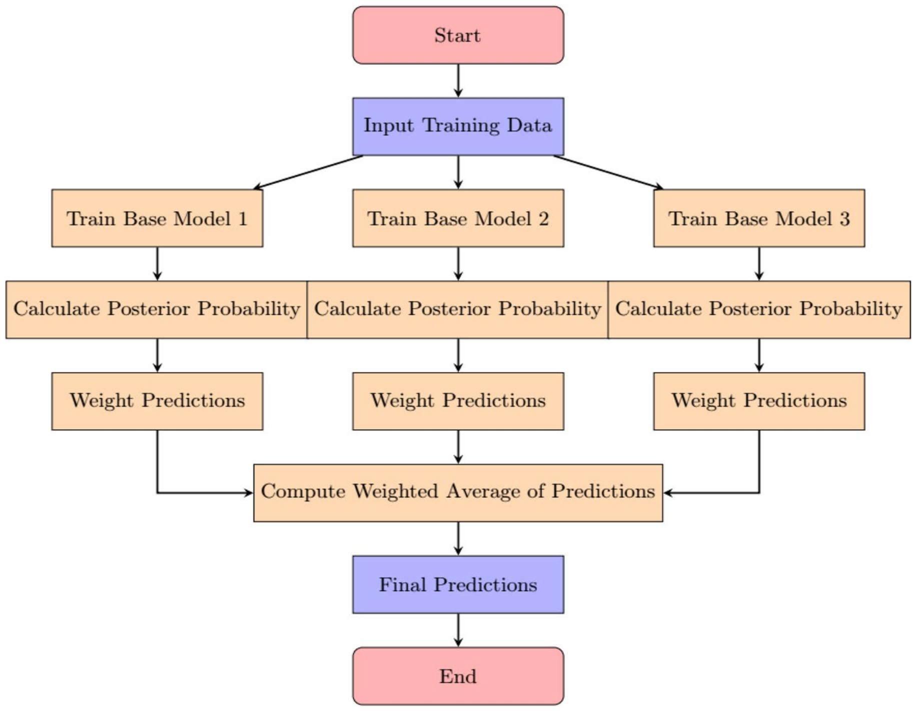


Fig. 4 Flowchart of the BMA model

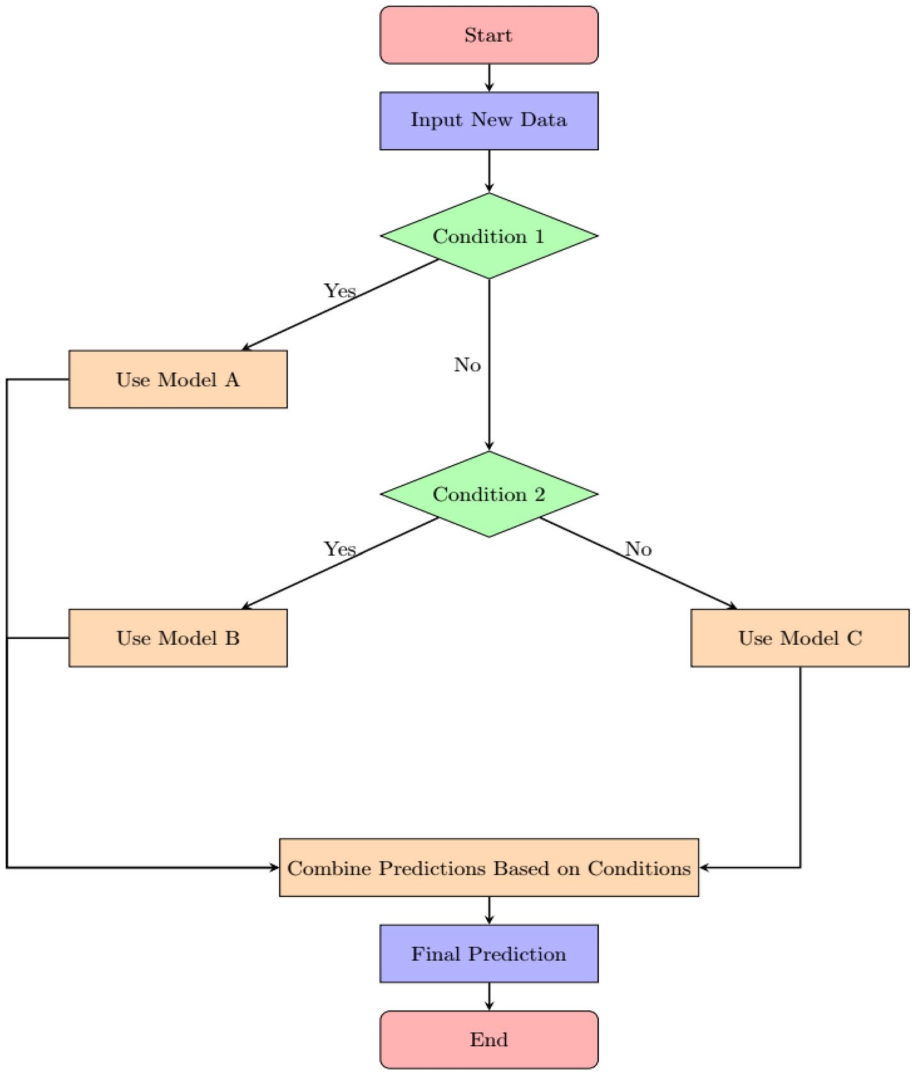


Fig. 5 Flowchart of the CEM model

adjusting dynamically to different data distributions). CEM evaluates the features of the incoming data and conditions, including weighting models, and how well it expects the model to perform under those conditions. For example, one of the models might perform much better than the others when a specific traffic anomaly arises. CEM can take advantage of this by boosting the weights of those models when the anomaly occurs. Because the ensemble can learn dynamically and tune for specific conditions, it is possible to reach a single ensemble that can deal with the differences across intrusion detection tasks, playing a role in reducing false alarms and maximizing detection rates.

Combining SEM, BMA, and CEM into the architecture of our proposed HAEnID model yields a highly flexible IDS. SEM improves the accuracy of our system by fusing the outputs of different base classifiers via the meta-classifier. BMA provides a measure of robustness by incorporating model uncertainty and makes our system more reliable in the face of novel or evolving threat models. Our CEM component also helps the model to learn to perform well in the face of different circumstances by learning to choose and weight models appropriate for the input data's characteristics so that it can perform well in different network situations while also dealing with varying types of attacks. Together, these components ensure that HAEnID is a cutting-edge IDS with high accuracy, low false positive rates, and the much-needed flexibility to track an attack landscape that is constantly changing and growing in complexity.

Algorithm 1 Comprehensive ensemble algorithm

```
Require: Training data $\mathscr{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$, Base classifiers $\mathscr{M}_1, \mathscr{M}_2, \ldots, \mathscr{M}_M$, Base classifiers for
    Conditional Ensemble $\mathscr{C}_1, \mathscr{C}_2, \ldots, \mathscr{C}_M$
Ensure: Final prediction model $f_{\text{ensemble}}(x)$
Step 1: Train Base Classifiers
for each base classifier $\mathscr{M}_m$ do
    Train $\mathscr{M}_m$ on $\mathscr{D}$
end for
Step 2: Generate Predictions for Training Data
for each training example $x^{(i)}$ do
    for each base classifier $\mathscr{M}_m$ do
        Compute prediction $\hat{y}_m^{(i)} = \mathscr{M}_m(x^{(i)})$
    end for
end for
Step 3: Form the Prediction Matrix $\hat{Y}$
Form the prediction matrix $\hat{Y}$ using all $\hat{y}_m^{(i)}$
Step 4: Train Meta-Learner for Stacking
Train the meta-learner $\mathscr{M}_{\text{meta}}$ on $\hat{Y}$ and $y$
Step 5: Compute SEM Prediction
Compute the SEM Prediction $f_{\text{SEM}}(x)$
Step 6: Compute BMA Prediction
Compute the BMA Prediction $f_{\text{BMA}}(x)$
Step 7: Compute CEM Prediction
for each base classifier $\mathscr{M}_m$ do
    Train the conditional classifier $\mathscr{C}_m$ on the misclassifications of $\mathscr{M}_m$
    Compute the probability $P(\mathscr{C}_m \mid x)$ for each class using $\mathscr{C}_m$
end for
Aggregate conditional probabilities $P(\mathscr{C}_1 \mid x), P(\mathscr{C}_2 \mid x), \ldots, P(\mathscr{C}_M \mid x)$
Compute the Conditional Ensemble prediction $f_{\text{CEM}}(x)$
Step 8: Final Prediction
Combine the SEM, BMA, and CEM predictions to form the final ensemble prediction:
$f_{\text{ensemble}}(x) = \alpha_1 f_{\text{SEM}}(x) + \alpha_2 f_{\text{BMA}}(x) + \alpha_3 f_{\text{CEM}}(x)$
where $\alpha_1, \alpha_2, \alpha_3$ are combination weights such that $\alpha_1 + \alpha_2 + \alpha_3 = 1$
return $f_{\text{ensemble}}(x)$
```

The proposed HAEnID model was built with the collective intelligence of multiple machine-learning models through a unique ensemble approach. We implement this model with several enhancements that are potentially unique. These enhancements introduce innovative elements to our model, which makes it more unique and capable of capturing complex patterns. The model comprises the following components:

- **Stacking Ensemble Component**
  - Base Classifiers: Include base classifiers with different characteristics to increase diversity. We add a DT, RF, MLP, and LGBM.
  - Feature Engineering: We apply feature engineering techniques to capture more information from the data. We also implement polynomial features to increase the complexity of the feature space.
  - Model Selection: We select the most relevant base classifiers based on their performance on test data and use techniques like cross-validation for performance testing.
  - Hyperparameter Tuning: We perform hyperparameter tuning for base classifiers and the meta-learner to optimize its performance.
  - Advanced Fusion Techniques: We implement model stacking with meta-learner optimization to combine prediction. Let $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_M$ be a set of M base classifiers. The prediction of model $\mathcal{M}_m$ for an input x is denoted by $f_m(x)$. The meta-learner $\mathcal{M}_{\text{meta}}$ is trained on the predictions of the base classifiers. Let $\hat{y}_m^{(i)}$ denote the prediction of model $\mathcal{M}_m$ for the i-th training sample $x^{(i)}$. In Eq. 1 the training data for the meta-learner consists of the predictions of the base classifiers.

$$\hat{y}_m^{(i)} = f_m(x^{(i)}), \quad \text{for } i = 1, 2, \ldots, N \text{ and } m = 1, 2, \ldots, M, \tag{1}$$

where N is the number of training samples. The final prediction for an input x using the stacking ensemble is denoted by $f_{\text{stack}}(x)$. The predictions of the base classifiers for x are $f_1(x), f_2(x), \ldots, f_M(x)$. These predictions are used as inputs to the meta-learner $\mathcal{M}_{\text{meta}}$ to produce the final prediction in Eq. 2:

$$f_{\text{stack}}(x) = \mathcal{M}_{\text{meta}}(f_1(x), f_2(x), \ldots, f_M(x)). \tag{2}$$

Algorithm 2 HAEnID algorithm

```
Require: Training dataset $D$, base learners $L_1, L_2, \ldots, L_k$, meta learner $M$, base classifiers $B_1, B_2, \ldots, B_k$,
    conditional model $C$
Ensure: Ensemble Model
Train base learners $L_1, L_2, \ldots, L_k$ on $D$
Generate predictions $\hat{Y}_i$ using $L_i$ for $i = 1, 2, \ldots, k$
Construct feature matrix $\mathbf{X}^*$ using $\hat{Y}_1, \hat{Y}_2, \ldots, \hat{Y}_k$
Train $M$ on $\mathbf{X}^*$ and true labels from $D$
if some condition then
    Compute predictions $\hat{Y}_i$ using $B_i$ for $i = 1, 2, \ldots, k$
    Compute weights $w_i$ based on performance
    Compute weighted average: $Y_{\text{bayesian}} = \sum_{i=1}^{k} w_i \cdot \hat{Y}_i$
else
    Compute predictions $\hat{Y}_i$ using $C$ for $i = 1, 2, \ldots, k$
    Train $C$ on predictions of base classifiers
end if
Dynamic Thresholding: Compute dynamic threshold $\tau = \frac{1}{n} \sum_{i=1}^{n} c_i$
Advanced Fusion Techniques: Compute $Y_{\text{fused}} = \frac{\sum_{i=1}^{N} w_i Y_i}{\sum_{i=1}^{N} w_i}$
Feature Engineering: Transform features $T_{\text{transformed}} = f(T(X_{\text{original}}), \alpha, \beta)$
Hyperparameter Tuning: Optimize hyperparameters $\Theta_{\text{opt}} = \text{argmin}\left(\frac{1}{m} \sum_{j=1}^{m} L(\Theta_j)\right)$
return HAEnID
```

- **Bayesian Model Averaging Component**
  - Feature Engineering: We apply feature engineering techniques to capture more information about the data. For this, we implement polynomial features to increase the complexity of the feature space.
  - Ensemble Methods: We have implemented an ensemble method called "Weighted Bayesian Model Averaging".
  - Advanced Formulation: Below is the advanced formulation of BMA. Let $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_M$ be a set of M models. Given data $\mathcal{D}$, the posterior probability of model $\mathcal{M}_m$ is denoted by $P(\mathcal{M}_m \mid \mathcal{D})$. The prediction for an input x from model $\mathcal{M}_m$ is denoted by $f_m(x)$. The Bayesian Model Averaging (BMA) prediction for an input x is given by Eq. 3:

$$f_{\text{BMA}}(x) = \sum_{m=1}^{M} P(\mathcal{M}_m \mid \mathcal{D}) f_m(x), \tag{3}$$

where $P(\mathcal{M}_m \mid \mathcal{D})$ is the posterior probability of model $\mathcal{M}_m$ given the data $\mathcal{D}$. Using Bayes' theorem, the posterior probability of model $\mathcal{M}_m$ is Eq. 4:

$$P(\mathcal{M}_m \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \mathcal{M}_m)P(\mathcal{M}_m)}{\sum_{k=1}^{M} P(\mathcal{D} \mid \mathcal{M}_k)P(\mathcal{M}_k)}, \quad (4)$$

where: - $P(\mathcal{D} \mid \mathcal{M}_m)$ is the marginal likelihood of the data given model $\mathcal{M}_m$, - $P(\mathcal{M}_m)$ is the prior probability of model $\mathcal{M}_m$. The marginal likelihood of the data given model $\mathcal{M}_m$ is Eq. 5:

$$P(\mathcal{D} \mid \mathcal{M}_m) = \int P(\mathcal{D} \mid \theta_m, \mathcal{M}_m)P(\theta_m \mid \mathcal{M}_m)\, d\theta_m, \quad (5)$$

where: - θ_m are the parameters of model $\mathcal{M}_m$, - $P(\mathcal{D} \mid \theta_m, \mathcal{M}_m)$ is the likelihood of the data given the parameters and the model, - $P(\theta_m \mid \mathcal{M}_m)$ is the prior distribution of the parameters given the model.

- **Conditional Ensemble Component**
  - Ensemble Diversity: We implement classifiers to increase diversity, such as LR, ADB, LGBM, and MLP.
  - Thresholding: We adjust the threshold based on the distribution of confidence scores. Let the ensemble consist of M models, denoted as $f_1, f_2, \ldots, f_M$. The predictions of these models for an input x are given by $f_1(x), f_2(x), \ldots, f_M(x)$. The conditional ensemble prediction is given by Eq. 6:

$$f_{\text{ensemble}}(x) = \sum_{m=1}^{M} w_m(x) f_m(x), \quad (6)$$

where $w_m(x)$ are the weights assigned to each model's prediction, dependent on the input x. To ensure that the weights $w_m(x)$ are non-negative and sum to one, we can use a softmax function in Eq. 7:

$$w_m(x) = \frac{\exp(g_m(x))}{\sum_{k=1}^{M} \exp(g_k(x))}, \quad (7)$$

In Eq. 8where $g_m(x)$ is a function that determines the raw (un-normalized) weight for model m. Assume $g_m(x)$ is a linear function of x:

$$g_m(x) = \beta_m^\top x + \gamma_m, \quad (8)$$

In Eq. 9where β_m and γ_m are parameters for model m. Combining these, the weight for model m becomes:

$$w_m(x) = \frac{\exp(\beta_m^\top x + \gamma_m)}{\sum_{k=1}^{M} \exp(\beta_k^\top x + \gamma_k)}. \quad (9)$$

Thus, the conditional ensemble prediction can be expressed in Eq. 10:

$$f_{\text{ensemble}}(x) = \sum_{m=1}^{M} \left(\frac{\exp(\beta_m^\top x + \gamma_m)}{\sum_{k=1}^{M} \exp(\beta_k^\top x + \gamma_k)} \right) f_m(x). \quad (10)$$

Matlab 3D bar chart
To create a 3D bar chart, in Eq. 11, we need to generate a grid of (x, y) values. Let N_x and N_y be the number of points in the x and y directions, respectively. We define the grid spacing as $\Delta x = \frac{X_{\max} - X_{\min}}{N_x - 1}$ and $\Delta y = \frac{Y_{\max} - Y_{\min}}{N_y - 1}$, where $X_{\max}$, $X_{\min}$, $Y_{\max}$, and $Y_{\min}$ are the maximum and minimum values of x and y respectively. Then, we generate the grid using the `meshgrid` function.

$$x, y = \text{meshgrid}(X_{\min} : \Delta x : X_{\max}, Y_{\min} : \Delta y : Y_{\max}) \quad (11)$$

In Eq. 12 using the function $f(x, y)$ and the generated grid, compute the corresponding z values.

$$z = f(x, y) \quad (12)$$

In Eq. 13 normalize the z values to ensure that they lie within the range [0, 1].

$$z_{\text{normalized}} = \frac{z - \min(z)}{\max(z) - \min(z)} \quad (13)$$

In Eq. 14 3D bar chart will use the `bar3` function with the computed x, y, and $z_{\text{normalized}}$ values.

$$\text{bar3} = f(x, y, z_{\text{normalized}}) \quad (14)$$

This formulation of the 3D bar creation chart uses the function $f(x, y, z)$ with additional normalization.

Data balancing stage
We implement all this in a two-way structure. First, we select all features to perform our proposed model on binary and multi-class and then choose 20 features selection with random forest classifier and apply Synthetic Minority Oversampling Technique (SMOTE) to implement our proposed model on multi-class. By integrating these ensemble methods, the proposed HAEnID framework aims to achieve superior detection performance, enhanced adaptability to new threats, and reduced false positive rates. This methodology addresses the challenges of detecting sophisticated cyber-attacks and offers a scalable and flexible solution adaptable to various network environments and attack scenarios.

Training process stage
However, several pre-processing steps were carried out to improve the quality and relevance of the dataset before training our models. The first step was data cleaning - to deal with missing values, i.e., areas in the table that had no information; replace NaNs (i.e., missing values) with

the mean or median of the feature the value was assigned to; identify and eliminate duplicate records; remove outliers using the z-score method. This step reduced the volume of data noise and helped make the dataset more presentable in a real-world scenario [41].

Feature selection was another vital step in the preprocessing pipeline to remove less useful features. We applied correlation analysis to select features with correlation coefficients above a certain threshold, not to reproduce each other and reduce redundancy. We also used information gain to screen the top features with the most significant information content for the target variable and PCA to reduce dimensions [42].

Normalization was also applied to standardize the dataset. Features were rescaled using min-max scaling, which rescales each feature to the same range [0, 1]. In this case, this is preferable because scaling prevents a feature, such as temperature, that spans an extensive range from disproportionately influencing the model. We also used Z-score normalization, transforming a feature into zero mean and unit variance. This normalization also prevents a single feature from overpowering the model. Lastly, we used one-hot and label encoding to transform categorical variables into machine-learnable formats [43].

Finally, random sampling stratified the data set in training and test sets. This was done to preserve the class distribution of the data in both sets. The main reason for having a balanced proportion of different attack types was that the models could be trained and tested on a balanced sample of data. In conclusion, the data set was thoroughly pre-processed, which helped develop a more efficient and trustworthy model [44].

Data preprocessing includes normalization, handling missing values, and feature selection to prepare the network traffic data for efficient processing. Each ensemble component is trained on a dataset containing various attack vectors and standard traffic patterns to ensure the model covers potential threats. Later, the model's performance is evaluated based on accuracy, precision, recall, and F1-score, providing a balanced assessment of its effectiveness in intrusion detection.

Instead of simply averaging the prediction probabilities from each classifier, we use a condition to decide how to combine them. There are many strategies, but we used weighted predictions by classifier confidence, which utilizes a meta-classifier, to predict the best-performing classifier on a validation set. **Probabilities Handling:** When dealing with multi-class classification, *predict_proba* outputs probabilities for each class. Evaluation metrics such as accuracy, confusion matrix, and classification report naturally extend to multi-class without needing binary-specific handling [45].

The primary data preprocessing and hyperparameters of the developed classifiers as shown in Tables 2 and 3.

Evaluation

For building a robust system for ML/DL intrusion detection, we use a system with these specifications in Table 4:

Table 2 Dataset preprocessing

Data Cleaning	**CIC-IDS2017**
Preprocessing Step	
Handling Missing Values	Replace NaN values with mean or median.
Removing Duplicates	Eliminate duplicate records.
Outlier Detection	Identify outliers using z-score: $z_i = \frac{x_i - \mu}{\sigma}$; remove if $\lvert z_i \rvert > k$.
Feature Selection	
Correlation Analysis	Select features with $\lvert C_{ij} \rvert >$ threshold.
Information Gain	Choose top k features with highest information gain.
Principal Component Analysis (PCA)	Perform PCA to reduce dimensionality.
Normalization	
Min-Max Scaling	Scale features to range $[0, 1] : x'_{ij} = \frac{x_{ij} - \min(x_i)}{\max(x_i) - \min(x_i)}$.
Z-Score Normalization	Transform features to have zero mean and unit variance: $x'_{ij} = \frac{x_{ij} - \mu_i}{\sigma_i}$.
Handling Categorical Variables	
One-Hot Encoding	Convert categorical variables to binary vectors.
Label Encoding	Map categorical labels to integers.
Splitting into Training and Test Sets	
Stratified Sampling	Ensure class distribution is preserved while splitting data.

Table 3 Hyperparameters for various classifiers

Classifier	Hyperparameters
Logistic Regression (LR)	penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, multi_class='auto', verbose=0
AdaBoost (AdB)	base_estimator=None, n_estimators=50, learning_rate=1.0, algorithm='SAMME.R'
LightGBM (LGBM)	objective='binary', metric='auc', boosting='gbdt'
Random Forest (RF)	n_estimators=100, oob_score=False, max_depth=None, min_samples_split=2, min_samples_leaf=1, max_features='auto'
Multi-layer Perceptron (MLP)	hidden_layer_sizes=(100,), activation='relu', solver='adam'
Decision Tree (DT)	criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, max_features=None

This setup balances computational power, speed, and storage, allowing for efficient training and deployment of machine learning models.

Dataset

The dataset we use is the Canadian Institute of Cybersecurity (CIC), introduced as CIC-IDS2017 [46], an innovative dataset incorporating modern network systems features and reflecting evolving cyber threats to assess their efficacy. This dataset addresses limitations identified in previous datasets and has gathered significant research attention. However, an experimental analysis of CIC-IDS2017 revealed the integral faults. This work presents an integrated dataset addressing these issues to enhance future IDS detection and classification capabilities. Table 5 shows the concise summary of the CIC-IDS2017 Wednesday subset.

Table 5 Concise summary of wednesday attack

Details of df8	Distribution of target variable in df8
Number of Rows: 692,703	BENIGN: 440,031
Number of Columns: 79	DoS Hulk: 231,073
Data Types: Float64 (24 columns), Int64 (54 columns), Object (1 column)	DoS GoldenEye: 10,293
Memory Usage: 417.5+ MB	DoS slowloris: 5,796
	DoS Slowhttptest: 5,499
	Heartbleed: 11

The dataset excerpts from the CIC-IDS2017 dataset selected from Wednesday's records. It represents the records of an attack dataset called 'df8'. It has 692,703 samples, 79 features, and mixed data entries of type Float64, Int64, and Object. The column types are 24 Float64, 54 Int64, and 1 Object, occupying 417.5+ MB of memory. The characteristics of features in the dataset are diversified, ensuring that the data used is representative of all the characteristics of both normal and malicious network traffic. It can be leveraged to train better intrusion detection models. The dataset includes five types of attacks: DoS attack variants and a handful of Heartbleed samples. The dataset contains 231,073 samples of DoS Hulk attacks, 10,293 samples of DoS GoldenEye, 5,796 samples of DoS slowloris, 5,499 samples of DoS Slowhttptest, and 11 samples of Heartbleed attacks. The dataset also contains 440,031 benign samples comprising normal, non-attack traffic. This distribution is an accurate representation of real-world traffic, containing both more popular attacks and rarer ones.

Evaluation matrices

This study uses several assessment metrics to evaluate the proposed intrusion detection model: accuracy, precision, Recall, and F1-score. These mathematical metrics

Table 4 System specifications

System	Description
CPU	AMD Ryzen 9 for efficient multi-threaded processing.
GPU	NVIDIA RTX 3080 with CUDA acceleration.
RAM	32GB DDR4 for handling large datasets.
Storage	512GB NVMe SSD + 500GB HDD for data storage.
Cooling System	High-quality cooling system for optimal performance.
Power Supply	750W Gold certified for efficiency and stability.
Operating System	Windows 10 Professional Edition.
Programming Language	Python for ML/DL and MATLAB.
ML Libraries	TensorFlow, PyTorch, scikit-learn.
Data Processing	Pandas, NumPy for manipulation.

measure different aspects of model performance and thus are widely used for evaluating the model [47].

After carefully categorizing data points, conventional measures like accuracy in Eq. 15,

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{15}$$

Accuracy measures how often the model correctly predicts the class of an instance. In the case of class imbalance, where one class represents a small fraction of the total number of samples, a simple accuracy score can be highly misleading. Imagine a dataset where 95% of the samples are benign, and a model that always predicts attacks as benign will achieve a 95% accuracy, but of course, it will not pick up any of the attacks. So, while accuracy is a valuable metric, it should be considered alongside others.

F1-score in Eq. 16,

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{16}$$

It is beneficial when precision needs to be balanced against the Recall, and it can be especially effective in cases where the class labels are imbalanced, such as when a class (which here represent negative examples such as normal traffic behaviour) is more common than other classes. This behaviour tends to lower the benign value of both precision and Recall. The F1-score is also relevant when minimizing false positives (false alarms) and false negatives (missed intrusions) - for example, in tasks such as intrusion detection, where false alarms and missed intrusions are damaging. The F1 score helps us evaluate a system's overall performance, measuring the trade-off between the detection of attacks and a reduction in false positives.

Classification reports in Eq. 17,

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{17}$$

Precision represents what is correct. Precision is paramount in intrusion detection because a false positive (normal traffic incorrectly predicted to be an attack) can spark an alert, wasting resources and potentially disrupting service, with the cost of this disruption often higher than that of misses. In such a setting, high precision is critical.

and ROC in Eq. 18 and AUC in Eq. 19 curve are used to evaluate the overall performance.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{18}$$

$$\text{AUC} = \int_0^1 \text{ROC}(t)\, dt \tag{19}$$

Conversely, Recall measures how well the model identifies all positive examples (attacks in our example). High Recall is a critical metric in intrusion detection since missing an attack (false negative) could have disastrous consequences. Thus, in this setting, as long as we can detect most attacks (as our model does with high recall), it is often better to have more false positives than more missed attacks. In scenarios where it is imperative to catch all the attacks, recall becomes the most crucial metric for evaluating models.

These evaluation metrics give a complete picture of the model's performance. Accuracy is an overall measure of correctness, but it can be insufficient, especially when the dataset is skewed or imbalanced. Precision and recall offer insight into the trade-offs between detecting all of the attacks and making the fewest errors, with the F1 score offering a well-balanced metric between precision and recall. For an IDS, wherein it is valuable to detect as many real threats as possible while making the fewest possible false alerts, all of these metrics are necessary to ensure that the model remains effective and dependable [48].

A cross-validation in Eq. 20 method was applied to evaluate the model performance. Using k-fold cross-validation with 5-fold, we assessed the models' accuracy, consistency, and generalizability across different training data splits [49].

$$\text{CV Accuracy} = \frac{1}{k} \sum_{i=1}^{k} \text{Accuracy}_i \tag{20}$$

where $k = 5$ in our case. Each fold involves training the model on $k - 1$ parts and testing it on the remaining part.

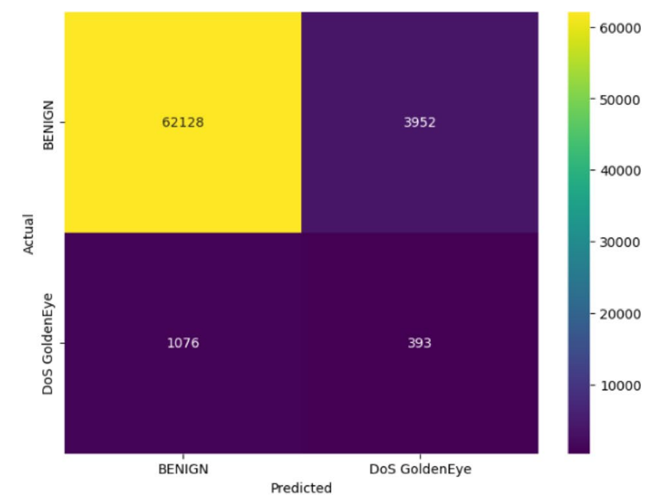
We lastly created learning curves to show how model performance varied with the quantity of training data. These curves made it easier to see any overfitting or underfitting problems and assess how big of a training set to use in Eqs. 21 and 22.

$$\text{Training Error} = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{y}_i^{\text{train}}) \tag{21}$$

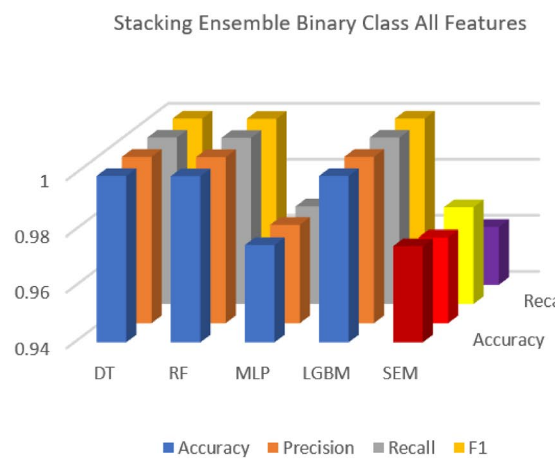
$$\text{Validation Error} = \frac{1}{m} \sum_{i=1}^{m} L(y_i, \hat{y}_i^{\text{val}}) \tag{22}$$

where L is the loss function, n is the number of training samples, and m is the number of validation samples.

This comprehensive approach thoroughly evaluated the ensemble's performance, guiding informed



(a) SEM Binary Confusion Matrix



(b) SEM Performance Barchart

Fig. 6 Performance of the SEM model using all features for binary class classification performance, with some mis-classifications noted for specific cases

decisions for model selection and improvement. SAHP and LIME techniques are then used for the HAEnID framework to support trust and comprehension in decision-making [50].

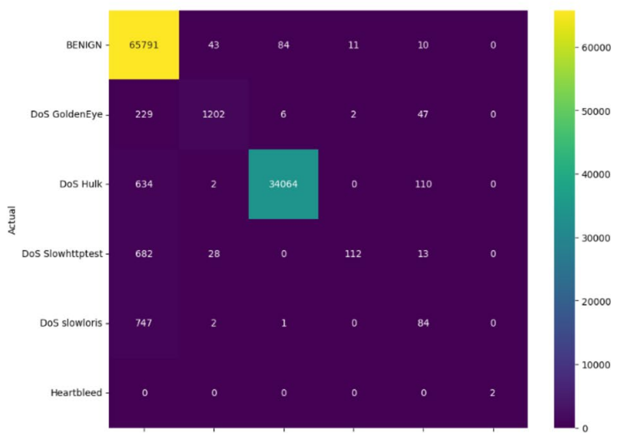
Receiver-operating characteristic (ROC), the curve plots the true positive rate against the false positive rate, and the Area Under the Curve (AUC) quantifies the classifier's performance. The ROC curve is obtained by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The AUC is calculated as the area under the ROC curve [51].

Results and discussion

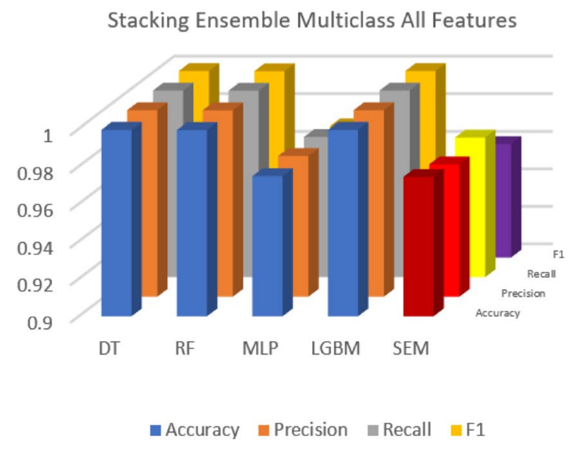
This section discusses each result separately and in light of the suggested model's conclusion to comprehend its performance using all features.

With near-perfect accuracy of over 97.44% across all measures, our state-of-the-art SEM model performed well in binary and multi-class classification on the CIC-IDS2017 dataset, as shown in Figs. 6 and 7.

DT performed best in binary classification tasks, achieving near-perfect accuracy, precision, recall, and an F1 score of 0.9994. However, the SEM performed best in multi-class classification tasks, where accuracy, precision, recall, and F1 scores were 0.9744, 0.9745, and



(a) SEM Multi-class Confusion Matrix



(b) SEM Performance Barchart

Fig. 7 Performance of the SEM model using all features for multi-class classification performance, with some mis-classifications noted for specific cases

Table 6 Performance metrics for SEM and individual models in both binary and multi-class classification tasks

Stacking Ensemble Binary					Stacking Ensemble Multi				
Model	**Accuracy**	**Precision**	**Recall**	**F1**	**Model**	**Accuracy**	**Precision**	**Recall**	**F1**
DT	0.9994	0.9994	0.9994	0.9994	DT	0.7595	0.8261	0.7596	0.7866
RF	0.9903	0.9903	0.9903	0.9903	RF	0.9993	0.9993	0.9993	0.9993
MLP	0.9748	0.9751	0.9748	0.9699	MLP	0.9748	0.9751	0.9748	0.9699
LGBM	0.9894	0.9894	0.9894	0.9894	LGBM	0.9994	0.9904	0.9914	0.9909
SEM	**0.9744**	**0.9706**	**0.9745**	**0.9606**	**SEM**	**0.9744**	**0.9706**	**0.9745**	**0.9606**

0.9606 respectively. The binary task also showed similar performance for both RF and LGBM, where the accuracy, precision, recall, and F1 score was 0.9903 across all metrics for RF and 0.9894 for all metrics of LGBM.

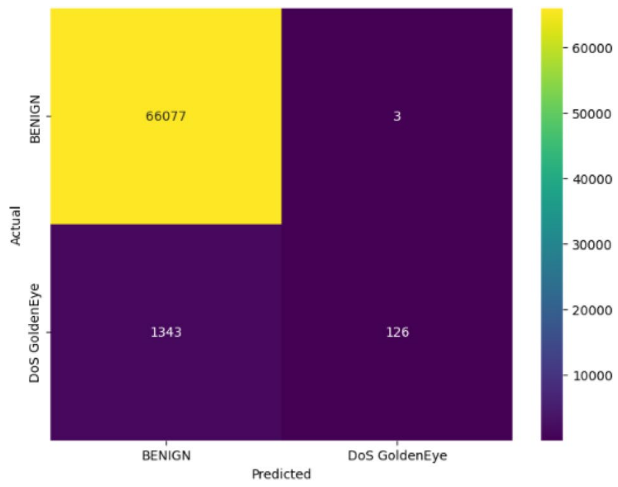
For the multi-class classification task, SEM had the highest F1 score of 0.9606, accuracy 0.9744, precision 0.9706, and recall 0.9745. Specifically, the SEM was good at balancing precision and recall - hence, it is suitable for cases where both false positives and false negatives matter. Note that LGBM had an accuracy of 0.9994, precision of 0.9904, recall of 0.9914, and F1 score of 0.9909, making it the second-best performing model for the multi-class classification task. Note that MLP came third, after LGBM, but before SEM. MLP's accuracy was 0.9748, while SEM's accuracy was precisely 0.9744.

However, baseline classifiers like LGBM and RF demonstrated impressive performance, with nearly 99% accuracy in both cases, as shown in Table 6. At the same time, other models, such as DT and MLP, scored well in multi-class classification.

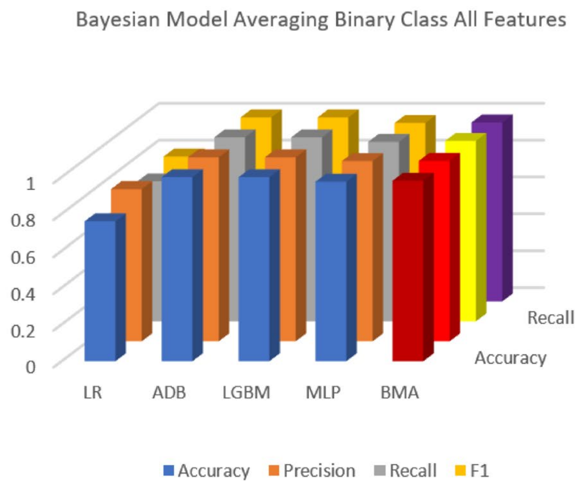
Nevertheless, it excelled in both classification types and had the best-balanced accuracy and the smallest standard error of probability calculation across different model-selection folds. These findings show that SEM is a reliable method to sustain a balanced and robust prediction for case studies where a balanced classification between different evaluation metrics is important.

The BMA for CIC-IDS2017 dominated the binary class with over 98.01% and the multi-class with over 97.94% classification, which was an outstanding most individual model (LR and MLP). AdB and LGBM performed well in binary and multi-class tasks, as shown in Figs. 8 and 9.

Regarding binary classification, the performance of the AdB and LGBM models was nearly perfect, boasting accuracy, precision, recall, and F1 of 0.9994 and 0.9993, respectively. The BMA method performed very well, with an accuracy of 0.9800 with precision and recall of 0.9800 and 0.9801, respectively, yielding an

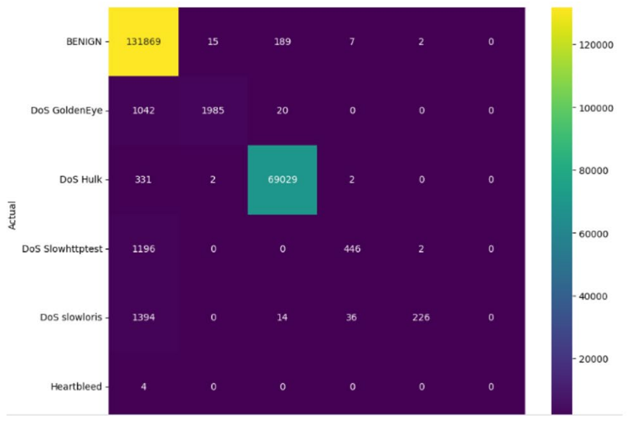


(c) BMA Binary Confusion Matrix

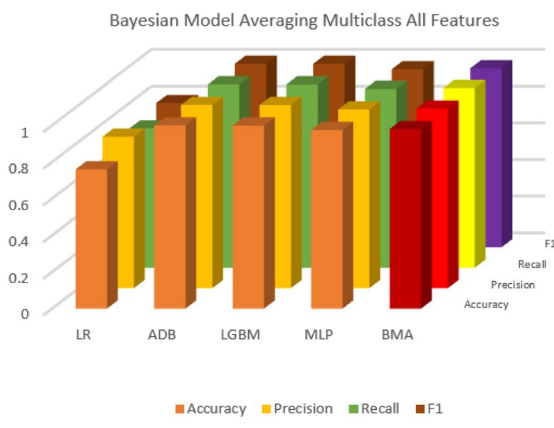


(d) BMA Performance Barchart

Fig. 8 Performance of the BMA model using all features for binary class classification performance, with some mis-classifications noted for specific cases



(c) BMA Multi-class Confusion Matrix



(d)BMA Performance Barchart

Fig. 9 Performance of the BMA model using all features for multi-class classification performance, with some mis-classifications noted for specific cases

F1 score of 0.9718. MLP performed the best in the F1 score, with a 0.9689, and yielded an accuracy of 0.9748, slightly lower than BMA. LR performed poorly concerning all these metrics, hovering around 0.7595.

LGBM achieved the highest performance for the multi-class classification, with the same metrics for all reaching 0.9993. AdB is coming in second, with all metrics reaching 0.9884. However, the performance of BMA is more robust, achieving an accuracy of 0.9794, with precision and recall of 0.9794 and 0.9795, respectively. However, the F1 score is 0.9749. Again, MLP and LR are at the bottom. But this time, LR is even weaker than MLP (accuracy and F1 score reaching 0.7595 and 0.7866, respectively). While BMA is a strong generalist, LGBM and ADB are better suited when high accuracy and precision are required, especially regarding higher complexity in the classification tasks as shown in Table 7.

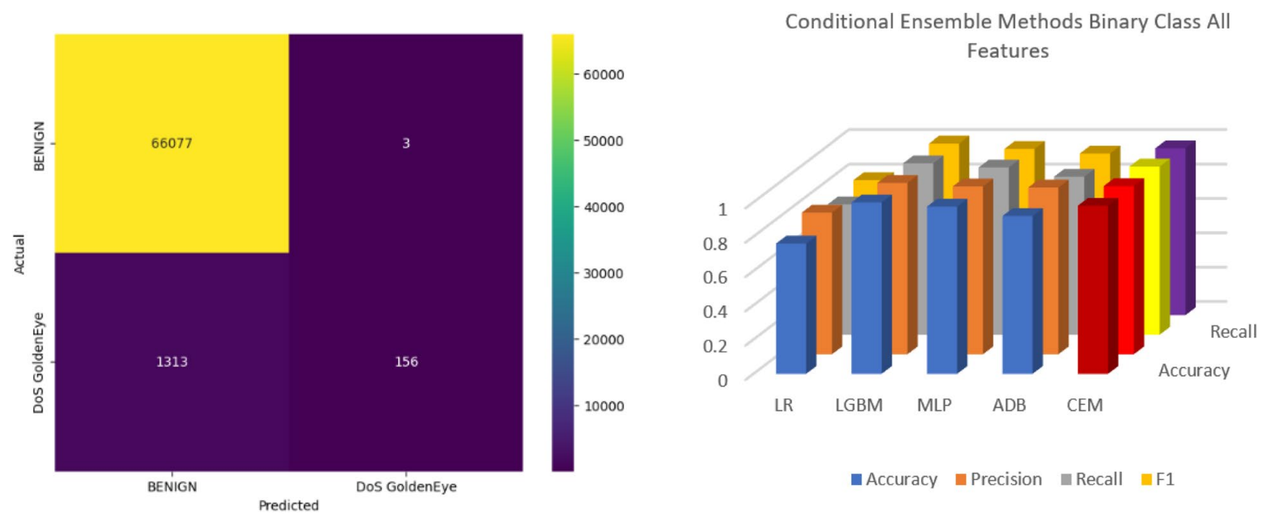
The CEM also performed well in CIC-IDS2017, achieving a near-perfect accuracy of 98.05% in binary classification and an outstanding performance of 97.25% accuracy with 96.55% F1-score in multi-class tasks. LGBM excelled in the ensemble's binary and multi-class performance dominance, as shown in Figs. 10 and 11.

In the binary classification task, the LGBM achieved the highest scores, with 0.9993 perfect in all metrics: accuracy, precision, recall, and F1. The CEM, followed by the LGBM algorithm, performed too well, with the highest accuracy, precision, recall, and F1 scores, with 0.9805 each. We can conclude that CEM is a viable option. There is almost no difference in performance in this task compared with the LGBM lighting learning. However, the MLP approach performs very well, with its final scores in accuracy and F1, which are 0.9748 and 0.9689, respectively. This point raises the MLP skill of this algorithm in handling binary classification. Finally, compared to all other algorithms, the LR algorithm performs the worst, with all metrics around 0.7595.

LGBM was again the best model with perfect accuracy, precision, recall, and F1 at 0.9993 for the multi-class classification task. CEM again did very well with an accuracy of 0.9725, precision of 0.9726, recall of 0.9725, and an F1 score of 0.9655. MLP slightly dropped scores, with an F1 of 0.9699. AdB gain did somewhat worse than CEM, with an F1 of 0.9424 and worse than in the binary task. LR was

Table 7 Performance metrics for BMA and individual models in both binary and multi-class classification tasks

Bayesian Model Averaging Binary					**Bayesian Model Averaging Multi**				
Model	**Accuracy**	**Precision**	**Recall**	**F1**	**Model**	**Accuracy**	**Precision**	**Recall**	**F1**
LR	0.7595	0.8261	0.7596	0.7866	LR	0.7595	0.7595	0.7596	0.7866
ADB	0.9994	0.9994	0.9994	0.9994	ADB	0.9884	0.9884	0.9884	0.9884
LGBM	0.9993	0.9993	0.9993	0.9993	LGBM	0.9993	0.9993	0.9993	0.9993
MLP	0.9748	0.9789	0.9748	0.9689	MLP	0.9748	0.9751	0.9748	0.9699
BMA	**0.9800**	**0.9800**	**0.9801**	**0.9718**	**BMA**	**0.9794**	**0.9794**	**0.9795**	**0.9749**



(e) CEM Binary Confusion Matrix (f) CEM Performance Barchart

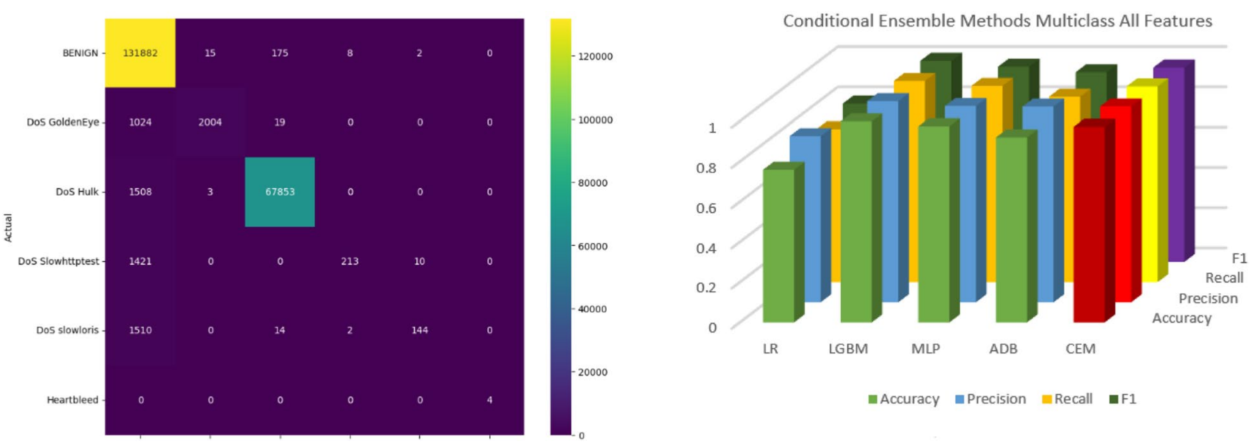
Fig. 10 Performance of the CEM model using all features for binary class classification performance, with some mis-classifications noted for specific cases

again the worst model, with the lowest scores possible. Undoubtedly, LGBM is still a better model overall than others. However, CEM appears robust and reliable when you need a balanced performance across different metrics as shown in Table 8.

This section discusses results in light of the suggested model's conclusion to comprehend its performance using 20% features for multi-class classification.

With near-perfect accuracy of over 98.17% across all measures, our state-of-the-art SEM performed well in multi-class classification on the CIC-IDS2017 dataset, as shown in Fig. 12. However, baseline classifiers like RF, DT and MLP demonstrated impressive performance, with nearly 99% accuracy. At the same time, other models, such as LGBM, scored well in multi-class classification with 20% feature selection.

LGBM achieved an accuracy of 0.8604, precision of 0.8456, recall of 0.8604, and an F1 score of 0.8673. In this category, the performance of the other models surpasses the LGBM model. RF and DT hold the highest performance on all score card (Accuracy: 0.9996, 0.9994; Precision: 0.9996, 0.9994; Recall: 0.9996, 0.9994; F1 Score: 0.9996, 0.9994, respectively), which means that RF and DT have similar performance in feature selection



(e) CEM Multi-class Confusion Matrix (f) CEM Performance Barchart

Fig. 11 Performance of the CEM model using all features for multi-class classification performance, with some mis-classifications noted for specific cases

Table 8 Performance metrics for CEM and individual models in both binary and multi-class classification tasks

Conditional Ensemble Methods Binary					Conditional Ensemble Methods Multi				
Model	**Accuracy**	**Precision**	**Recall**	**F1**	**Model**	**Accuracy**	**Precision**	**Recall**	**F1**
LR	0.7595	0.8261	0.7596	0.7866	LR	0.7595	0.8261	0.7596	0.7866
LGBM	0.9993	0.9993	0.9993	0.9993	LGBM	0.9993	0.9993	0.9993	0.9993
MLP	0.9748	0.9789	0.9748	0.9689	MLP	0.9748	0.9751	0.9748	0.9699
ADB	0.9199	0.9722	0.9211	0.9424	ADB	0.9199	0.9722	0.9211	0.9424
CEM	**0.9805**	**0.9805**	**0.9805**	**0.9728**	**CEM**	**0.9725**	**0.9726**	**0.9725**	**0.9655**

scenario, strong performance, and stable data processing. MLP is also a strong performer, achieving an accuracy of 0.9972, a precision of 0.9973, a recall of 0.9973, and an F1 score of 0.9972 as shown in Table 9.

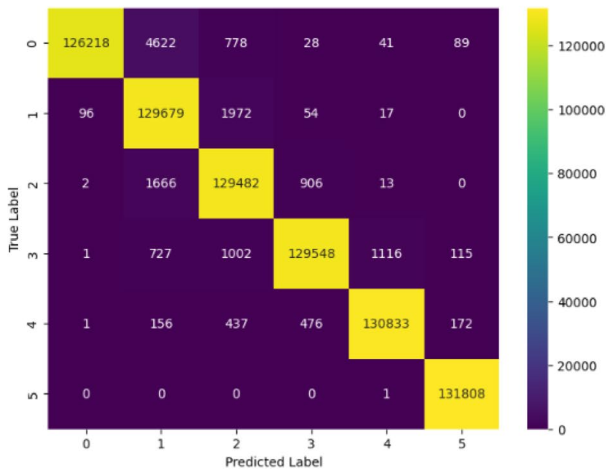
The BMA model for CIC-IDS2017 dominated the multi-class with over 98.79%, an outstanding most individual model (LR and LGBM). AdB and MLP also performed well in multi-class tasks with a high accuracy of 99%, as shown in Fig. 13.

Since LR performs poorly in this setting, its accuracy is 0.7604, precision is 0.8456, recall is 0.7604, and F1 score is 0.7973. Strangely, it works worse in this case. Therefore, ADB and MLP maintain their excellent performance. ADB gives perfect scores (Accuracy 0.9994, Precision: 0.9994, Recall: 0.9994, F1 Score: 0.9994) and MLP keeps its high metrics (Accuracy: 0.9972, Precision: 0.9973, Recall: 0.9973, F1 Score: 0.9972). BMA has the highest overall performance in this category, and all metrics gain perfect scores (Accuracy: 0.9879, Precision: 0.9879, Recall: 0.9879, F1 Score: 0.9879). It guarantees that it can greatly help feature elimination, as shown in Table 9.

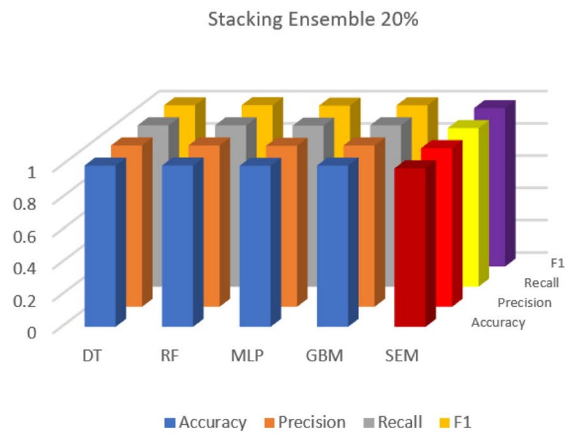
The CEM model dominated CIC-IDS2017, achieving an outstanding performance of 96.76% accuracy and a 96.15% F1 score in multi-class tasks. MLP and ADB excelled in the ensemble's dominance in multi-class performance, as shown in Fig. 14.

LR and ADB perform similarly to other feature selection methods. LGBM and MLP confirm their previous results. CEM highlights the validity of its results with an accuracy of 0.9676, precision of 0.9684, recall of 0.9676, and F1 score of 0.9615. Although not the highest in all metrics, it keeps a balanced and competitive performance as shown in Table 9.

Overall, RF and DT present the highest scores in the SEM. Conversely, BMA provides the highest and most stable metrics in the BMA method. CEM is also competitive due to its feature selection ability, particularly in the conditional ensemble setting.



(a) SEM Multi-class 20-feature Confusion Matrix



(b) SEM Performance Barchart

Fig. 12 Performance of the SEM model using 20 feature selections for multi-class classification performance, with some mis-classifications noted for specific cases

Table 9 Performance metrics for different feature selection methods across various models

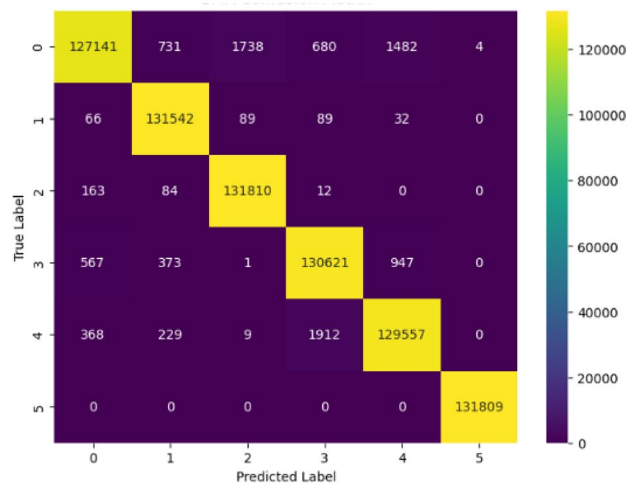
Model	Accuracy	Precision	Recall	F1
Stacking Ensemble Feature Selection				
LGBM	0.8604	0.8456	0.8604	0.8673
RF	0.9996	0.9996	0.9996	0.9996
MLP	0.9972	0.9973	0.9973	0.9972
DT	0.9994	0.9994	0.9994	0.9994
SEM	**0.9817**	**0.9821**	**0.9817**	**0.9818**
Bayesian Model Averaging Feature Selection				
LR	0.7604	0.8456	0.7604	0.7973
ADB	0.9994	0.9994	0.9994	0.9994
LGBM	0.8604	0.8456	0.8604	0.8673
MLP	0.9972	0.9973	0.9973	0.9972
BMA	**0.9879**	**0.9879**	**0.9879**	**0.9879**
Conditional Ensemble Model Feature Selection				
LR	0.7604	0.8456	0.7604	0.7973
ADB	0.9994	0.9994	0.9994	0.9994
LGBM	0.8604	0.8456	0.8604	0.8673
MLP	0.9972	0.9973	0.9973	0.9972
CEM	**0.9676**	**0.9684**	**0.9676**	**0.9615**

Figures 15, 16, and 17 show similar trends between training and cross-validation scores of binary and multi-class classifications. This trend confirms the state-of-the-art model's ability to learn from the data, indicating that the model effectively captures underlying patterns and generalizes new data well. We have also compared the results with the state-of-the-art model as shown in Table 10.

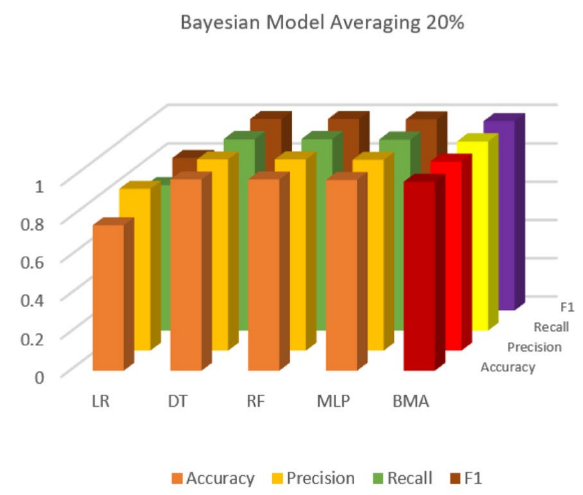
Performance analysis

The main contribution of this work is developing a HAE-nID model, which produces improved results and the process on various models across the CIC-IDS2017 data-set, including binary and multi-class classification with full features and 20 feature selections; we have performed some ablation studies. These studies are discussed in the following subsection.

The performance of our proposed model shows promising state-of-the-art results in both binary and multi-class classification with full feature selection as shown in Fig. 18. Among the models, SEM-B (SEM-Binary) showed excellent accuracy, precision, recall, and an F1 score of 97.44%. Similarly, BMA-B (BMA-Binary) exhibited strong performance with 98.01%, 98.02%, 98.01%, and 97.18% of accuracy, precision, recall, and F1 score, while the CEM-B (CEM Binary) model performed well with a perfect 98.05% across accuracy and F1 score with97.28%. Moving on to the multi-class classification results, the proposed SEM-M (SEM-Multiclass) model performed well with 97.44%, 97.06%, 97.45%, and 96.06% accuracy, precision, recall, and F1 score. The proposed BMA-M (BMA-Multiclass) model is classified equally and correctly with an accuracy and precision of 97.94% and 97.94%, with a 97.49% f1 score. The proposed CEM-M (CEM-Multiclass) model also showed

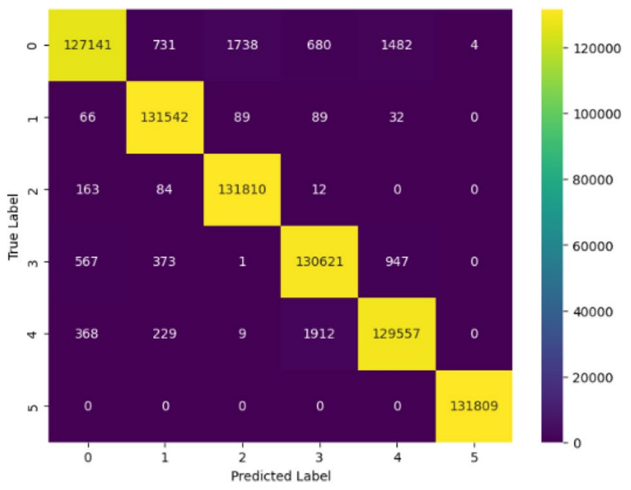


(c) BMA Multi-class 20-feature Confusion Matrix



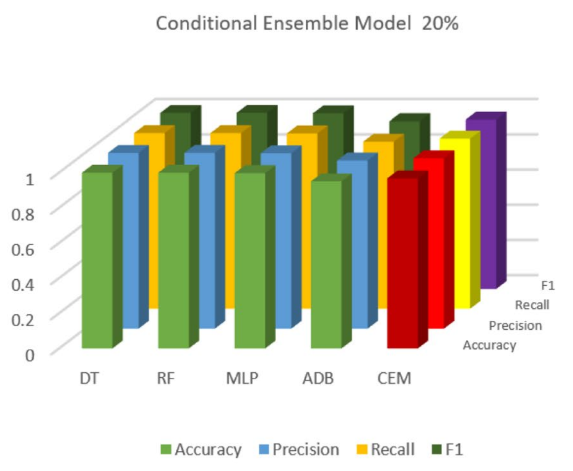
(d) BMA Performance Barchart

Fig. 13 Performance of the BMA model using 20 feature selections for multi-class classification performance, with some mis-classifications noted for specific cases



	0	1	2	3	4	5
0	127141	731	1738	680	1482	4
1	66	131542	89	89	32	0
2	163	84	131810	12	0	0
3	567	373	1	130621	947	0
4	368	229	9	1912	129557	0
5	0	0	0	0	0	131809

(e) CEM Multi-class 20-feature Confusion Matrix



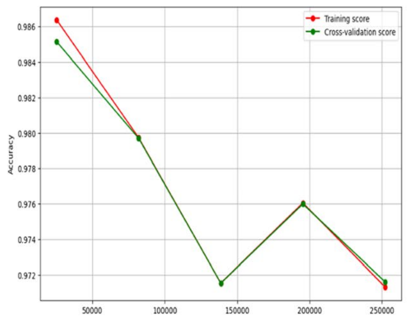
(f) CEM Performance Barchart

Fig. 14 Performance of the CEM model using 20 feature selections for multi-class classification performance, with some mis-classifications noted for specific cases

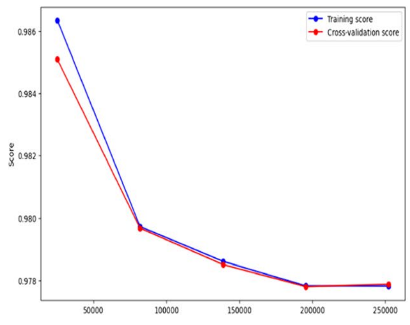
substantial accuracy and a precision score of 97.25%, 97.26% recall, and F1 score of 97.25% and 96.55%.

Likewise, the proposed model performs desirably for classified in multi-class classification with 20 feature selections. The model with SEM-M (20 features) provides accuracy and precision at 98.17% and 98.21%, respectively-similarly, the accuracy and precision with 98.17% and 98.18% for recall and F1 score. The proposed BMA-M (20 features) model also shows balanced metrics and good performance, with a 98.79% accuracy, a 98.79% precision, and a 98.79% F1 score in another. The CEM-M (20 features) model shows substantial accuracy, precision of 96.76%, 96.84%, recall, and F1 scores of 96.76% and 96.15%.

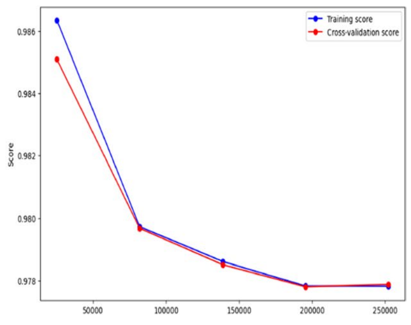
The main objective is to improve the Explainable AI-IDS framework with clarity and confidence in black-box AI models, which will be utilized for intrusion detection. With the justifications and clarity of the decisions produced by these models, the Explainable AI framework seeks to establish trust among the operators and analysts, enabling the more significant omission of self-directed AI subsystems. It is also accomplished by producing specific and overall explanations using various Explainable AI models. Also, the framework identifies the significance of features unique to the model and intrusion. Lastly, it makes it easier for human operators to understand and interpret AI models.



(a) SEM Binary Learning Curve

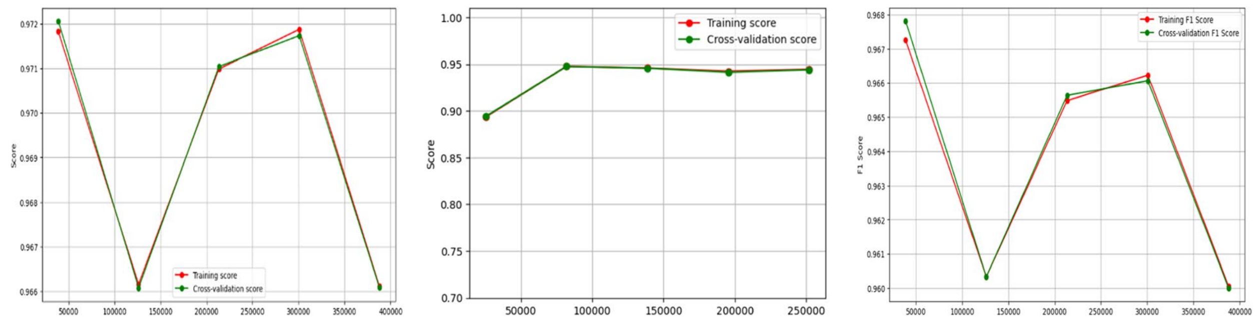


(b) BMA Binary Learning Curve



(c) CEM Binary Learning Curve

Fig. 15 Learning curve analysis on our cross-validation output for binary class



(a) SEM Multi Learning Curve (b) BMA Multi Learning Curve (c) CEM Multi Learning Curve

Fig. 16 Learning curve analysis on our cross-validation output for multi-class

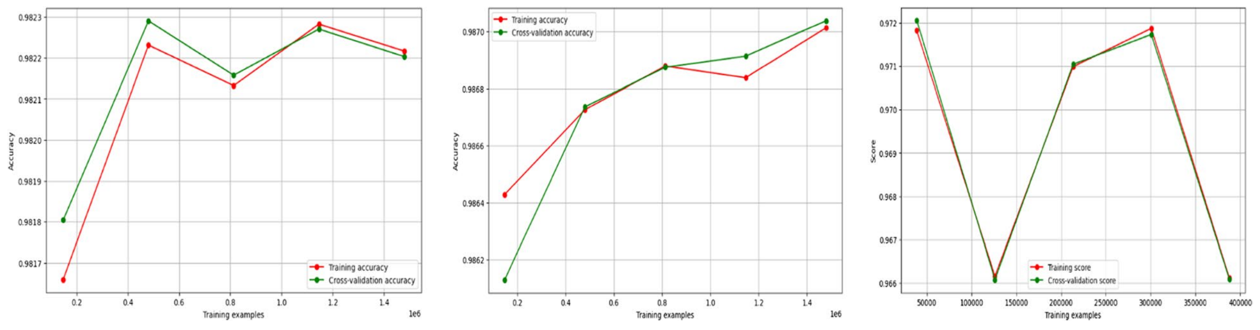
Explainable AI analysis

Some manifestation of interpretability is necessary for establishing trust in the IDS models, and if malicious intent is found, it must be defensible. A possible way forward is through techniques like SHAP and LIME, which could make IDS models more interpretable and, therefore, understand how and why they make their decisions [52].

SHAP values are derived from game theory and provide an aggregated view of feature contributions to model predictions. Applied to the context of an IDS, this translates into an explanation of why a model tags a network activity as an intrusion. For example, suppose the obtained model tags a certain activity in the network as suspicious. In that case, the SHAP values will convey which features were the most important in triggering this detection (such as abnormal access via uncommon ports, the anomalous volume of traffic or packet size, etc). These values will then allow security analysts to determine which part of the network traffic is malicious, help them tune their IDS, or even better understand why some behaviours are suspicious [53].

In contrast, LIME gives a local explanation by fitting an interpretable surrogate model locally to the IDS model around each prediction. This local explanation is often informative in understanding why a network activity pattern has been diagnosed maliciously. For example, in cases where the model has diagnosed a single network packet as malicious, LIME can localize that diagnosis such that, as features of the packet are perturbed (e.g., the source IP address or protocol type), the diagnosis becomes less likely. This local explanation can help analysts learn more quickly if the activity is genuinely suspicious or a false positive. It also provides more actionable information about why the model arrived at its conclusion [54].

SHAP and LIME, we can make such IDS models more transparent. Regarding SHAP, we can grasp the global picture of feature importance and the overall model behaviour, i.e., the performance in all instances. Regarding LIME, we can also examine the detailed information about a particular prediction. In the security domain, this will, first and foremost, increase our trust in the anomaly detection results shown. Secondly, it will help us track



(a) SEM Feature Learning Curve (b) BMA Feature Learning Curve (c) CEM Feature Learning Curve

Fig. 17 Learning curve analysis on our cross-validation output for 20 features multi-class

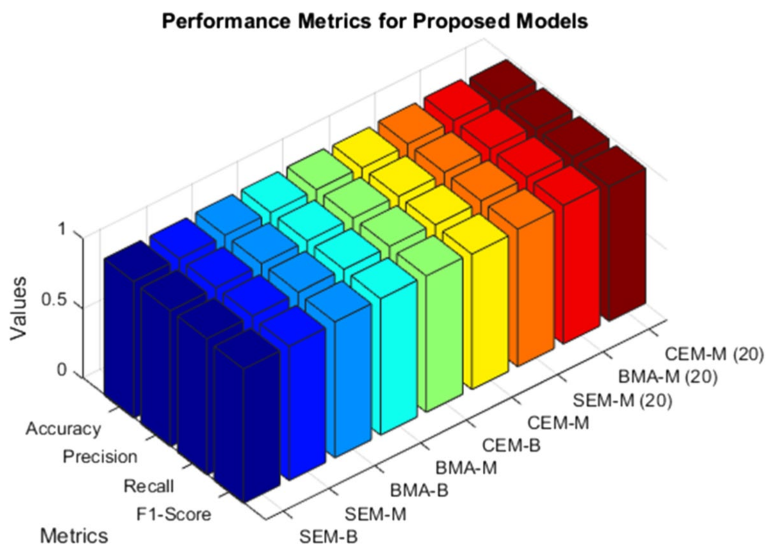


Fig. 18 Overall performance of the proposed model architecture for Binary and Multi-class Classification with full and 20 features selection

a particular type of threat. Thirdly, improving the IDS models will help us reduce false positives and false negatives [55].

We use the framework of Explainable AI-IDPS, SHAP, and LIME. This end-to-end explainable AI framework increases the understandability of AI models for network IDSs. It also involves benchmarking black-box AI models as shown in Fig. 19, including real-world network intrusion datasets. It generates local and global explanations, using SHAP and LIME models and extracting the model-specific, intrusion-specific, essential features. This framework also aims to help security analysts, making them more informed in decision-making. There are also different explanations for achieving the security level [56]. To calculate SHAP we use SHAP formula in Eq. 23.

$$\text{SHAP}(\text{feature } i) = \phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} [f(S \cup \{i\}) - f(S)] \tag{23}$$

In this formula:

- ϕ_i represents the SHAP value for feature i.
- N represents the set of all features.
- S represents a subset of features excluding feature i.
- $|S|$ and $|N|$ represent the cardinality of sets S and N, respectively.
- $f(S)$ represents the model's output when considering only the features in subset S.
- The outer sum iterates over all possible subsets S of features N excluding i.
- The term $\frac{|S|!(|N|-|S|-1)!}{|N|!}$ represents the binomial coefficient, which represents the number of ways to choose a subset S of size $|S|$ from N features.

Summary plot

The SHAP summary plot is a global interpretability tool that depicts the significance and effects of the features in a machine learning model. This plot is based on Shapley values from game theory, which assigns an average contribution value to each feature for the prediction. The y-axis displays the features of the model ranked in descending order of importance. The importance here is determined by the sum of SHAP value magnitudes across all samples. The x-axis represents the SHAP values, which measure the impact of each feature on the model's prediction. These values are calculated for each feature across all the data points [57].

Figure 20a summarizes the total influence of the features on our model - the SHAP summary plot. Each dot in this plot represents a feature value's influence on an instance's prediction. Each feature's value is represented on the x-axis by the SHAP value, and the features are ordered on the y-axis by their importance. The color of the dots represents the feature value, with blue representing low values and red representing high values.

In Fig. 20a, features with the most substantial impact on the model output - like 'Active Max' and 'Active Std'

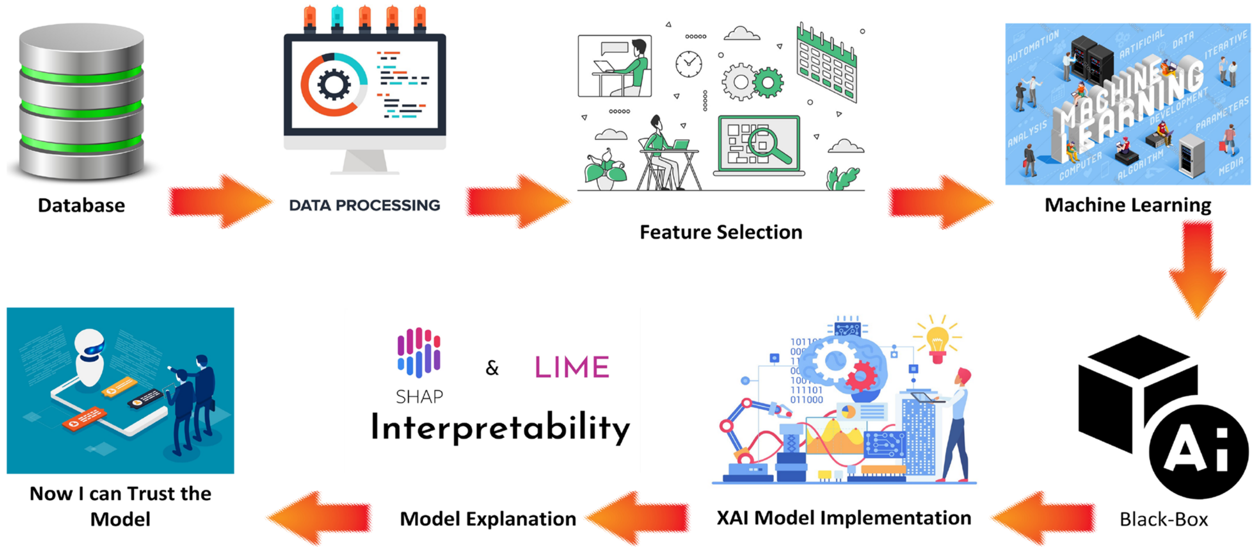
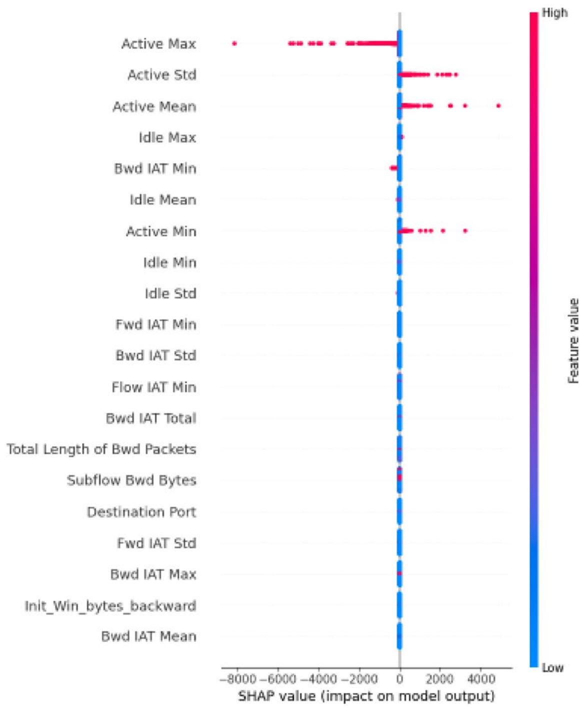


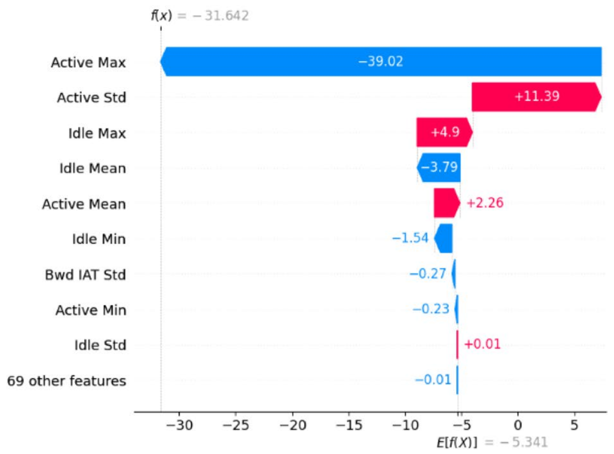
Fig. 19 The black-box AI model, including network intrusion datasets

- are at the top. Notice the dots are much more concentrated around zero, suggesting many feature values have little or no effect, and the ones further away have more influence.

The SHAP summary plot in Fig. 21a illustrates the most critical features across multi-class predictions. Median explanatory features strongly influence the outcome: 'Flow IAT Max', 'Flow Duration' and 'Idle Max' have the most significant effects, reflected in the high span of their SHAP values. 'Packet Length Variance' and 'Idle Std' have a more moderate impact - their SHAP values range closer to 0. The distribution of dots visualizes the persistent

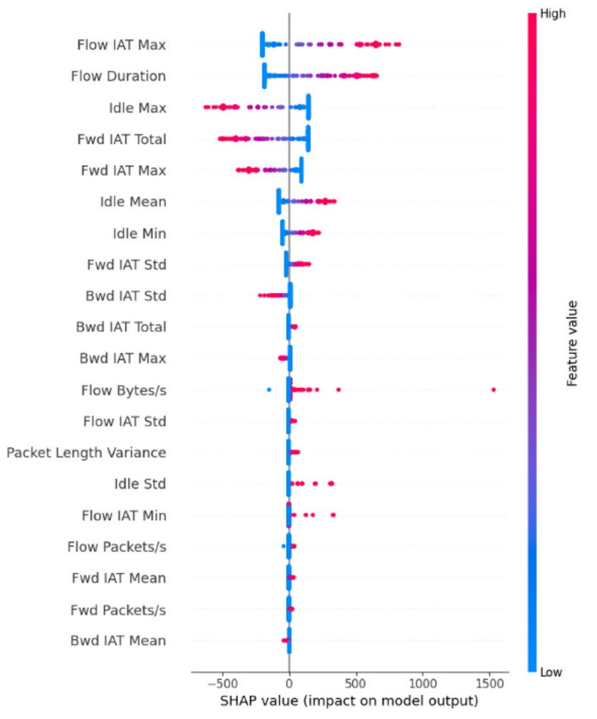


(a) SHAP Summary Plot

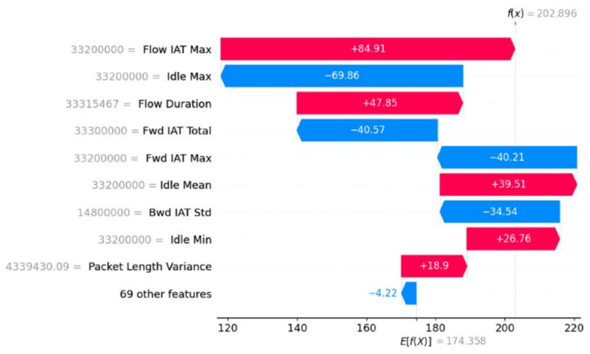


(b) SHAP Waterfall Plot

Fig. 20 SHAP Summary plot of base classifiers that we used in the model's prediction for binary classification

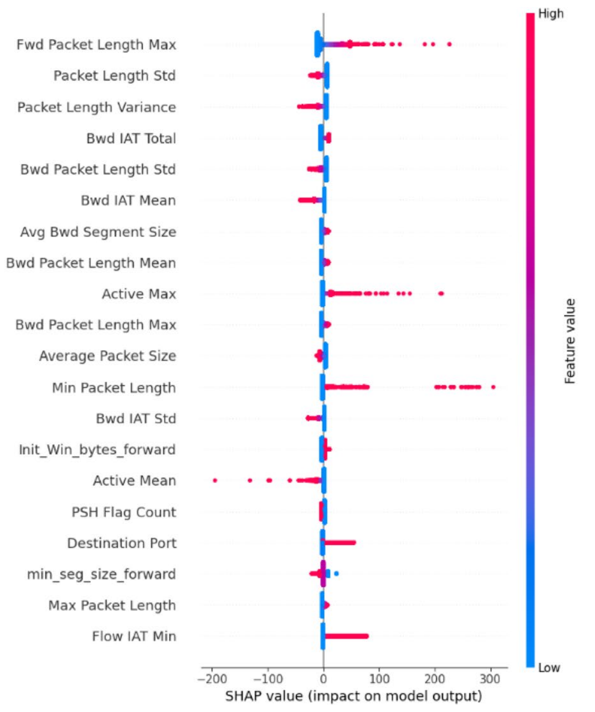


(a) Shap Summary Plot

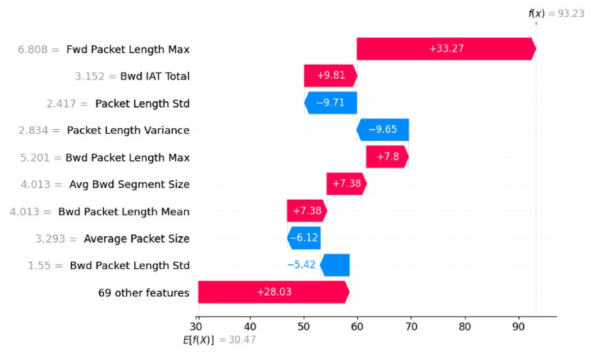


(b) Shap Waterfall Plot

Fig. 21 SHAP Summary plot of base classifiers that we used in the model's prediction for multi-class classification



(a) SHAP Summary Bar Plot



(b) SHAP Summary Plot

Fig. 22 SHAP Summary plot of base classifiers that we used in the model's prediction against 20 feature importance for multi-class classification

features that will impact the model's prediction and the direction of the influence toward either increasing or decreasing the predicted outcome.

Figure 22a below depicts a SHAP Summary Plot in which 'Fwd Packet Length Ma'x and the 'Packet Length Std' appear to be the most critical features that affect prediction results. The distribution of those data points makes it easier to understand that 'Fwd Packet Length Max' will influence model predictions in different ways according to an instance of data. It is the most essential feature of all the ones contributed.

Waterfall plot

The graph in the image is a type of explanatory visualization known as a waterfall plot. This plot illustrates how the individual features contribute to a model's prediction for a specific instance-starting from the baseline average prediction and adding the effects of each feature to make a final prediction [58].

The SHAP waterfall plot in Fig. 20b provides a detailed divergence breakdown for a single prediction by showing a feature-by-feature combination of SHAP values that leads to the final prediction. The base value features are added in sequential order based on their rank of importance. Adding subsequent features does not alter the order in which the previously added features are ranked. In that sense, the base value is considered a neutral reference point. Each feature either increases or decreases the prediction depending on the magnitude of the SHAP value.

In Fig. 20b, the base value is around -2.341 with "Active Max" and "Idle Max" dramatically shifting the prediction to -31.64. Features such as "Active Mean", "Idle Mean" and more are then shown to contribute more to the prediction, leading to the final output of -31.642.

Figure 21b shows that 'Flow IAT Max' (332000000) increases the prediction by +64.81, 'Idle Max' (332000000) and 'Flow Duration' (3331546.7) have a similar impact. 'Fwd IAT Max' (332000000) and 'Idle Mean' (332000000) have a more negligible but positive effect. 'Packet Length Variance' (4339430.0), 'Idle Min' (332000000) have a slight negative impact. The combined effect of these feature contributions brings the total predicted value to approximately 202.90, starting from the baseline, around 4.20.

The most prominent features in Fig. 22b plot contribute much more positively to the predicted class. The feature' Fwd Packet Length Max' has a very high positive SHAP value contribution to the model, while splits such as 'Bwd IAT Total' and 'Packet Length Std' have high negative SHAP values as they weaken the expected class. This plot clearly shows how each feature contributes to the prediction decision, with those features having higher contributions rising and those having less influence going down, granting the track of the model's reasoning and making it genuinely interpretable.

Force plot

The image displays a feature importance ranking derived from a machine learning model. Feature importance rankings are used to identify which features (or input variables) are considered most influential in predicting the target variable by the model [59].

The SHAP force plot in Fig. 23 provides more information about which specific combination of feature values contributed to this prediction with a value of -31.64. From the plot, features that moved the prediction higher (to the right) are distinguished from features that pushed the prediction lower (to the left).

In Fig. 23, a feature value of 321 mph for 'Active Max' impacts the prediction negatively, moving it to the left. In comparison, a feature value of 150 mph for 'Idle Max 'and a feature value of 96 mph for 'Active Std', moved the prediction positively, though by a lesser amount. The force plot visualizes the contribution of increments in each feature to the final prediction.

In Fig. 24, the positive force values on the prediction are high values for 'Flow IAT Max' and 'Idle Max', 'Flow Duration', and others. While opposing these positive forces are features like 'Packet Length Variance', 'Idle Min', and others. The figure visualizes how feature values interact individually with the final prediction, 202.90.

Figure 25, SHAP Force Plot depicts the individual feature effects - 'Packet Length Variance' is seen to have a negative impact on prediction, whereas 'Packet Length Std' shows the same feature having a positive effect.

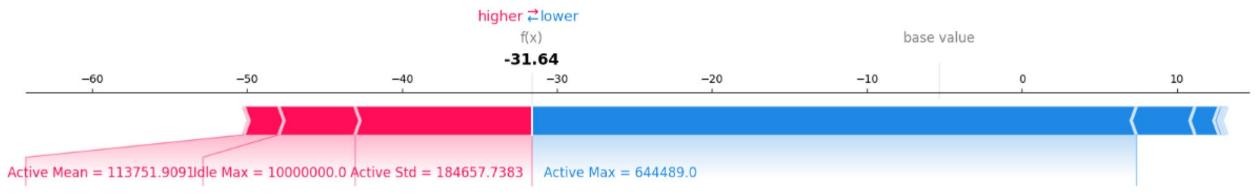


Fig. 23 SHAP force plot

Similarly, 'Fwd Packet Length Max' is seen to have an opposite effect compared to 'min seg size forward'.

LIME

This LIME visualization allows for an in-depth analysis of how the model processes individual data points, which is crucial for model validation and debugging and in cases where model interpret-ability is necessary, such as in domains with regulatory requirements explained to stakeholders in Eq. 24.

$$\hat{g}(x) = \arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g) \tag{24}$$

In this formula:

- $\hat{g}(x)$ represents the locally interpretable model that approximates the complex model f around instance x.
- G represents the set of candidate models.
- $L(f, g, \pi_x)$ represents the loss function measuring the discrepancy between the predictions of the complex model f and the locally interpretable model g, weighted by the proximity π_x of the instance x.
- $\Omega(g)$ represents the complexity penalty imposed on the locally interpretable model g, encouraging simpler explanations.

The goal of LIME is to find a locally interpretable model $\hat{g}(x)$ that both accurately approximates the complex model f around instance x and is simple enough to be interpretable [60].

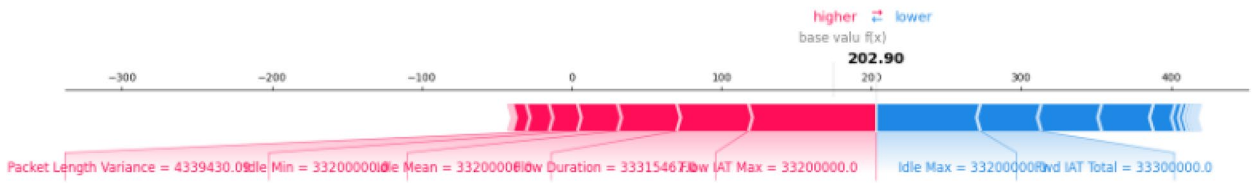


Fig. 24 SHAP force plot

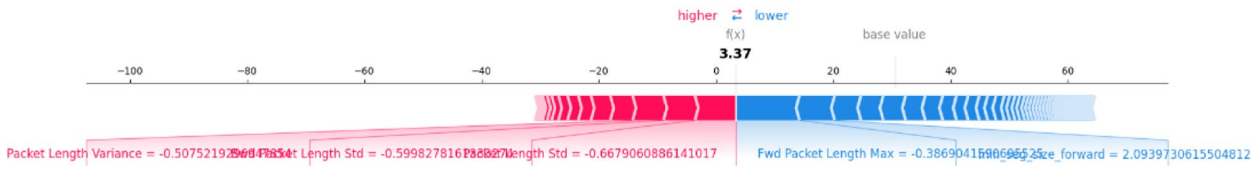
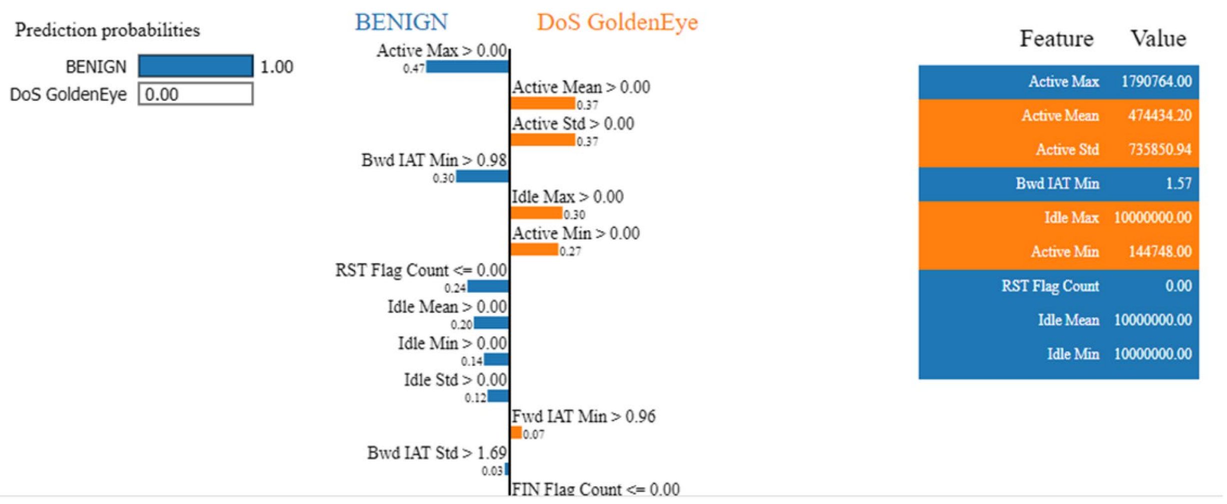


Fig. 25 SHAP force plot



Feature	Value
Active Max	1790764.00
Active Mean	474434.20
Active Std	735850.94
Bwd IAT Min	1.57
Idle Max	10000000.00
Active Min	144748.00
RST Flag Count	0.00
Idle Mean	10000000.00
Idle Min	10000000.00

Fig. 26 Local explanation model's predictive probabilities for a binary classification task distinguishing between 'normal' and 'attack' network attack events

Figure 26 illustrates a local explanation of the probability of a predictive model. In this case, it is a predictive task for a binary classification, that is, distinguishing between "normal" (BENIGN) versus "attack" (DoS Golden Eye) network events. The figure shows a breakdown of the prediction's confidence score and features that the model used to make a decision. We see that the predicted probability of observing the event belongs to the 'BENIGN' class, which is extremely high (1.00), and the expected probability of the alternative class, 'DoS Golden Eye', is very low (0.00). This indicates a specific model, which is observed as 'BENIGN'.

Figure 26 gives the judgment as 'BENIGN', because the 'Active Max'> 0.00. This means that multiple values of 'Active Mean', 'Active Std', 'Bwd IAT Min, RST Flag Count', 'Idle Mean', are also more significant than the threshold, and non-zero values of other features enforce this decision. The breakdown of the prediction to each step reveals how the specific value of the features pushes the model to predict 'BENIGN' instead of 'DoS Golden Eye', and illustrates the values of the features that the particular instance has. Those feature values exactly map to the breakdown's features. For example, the value of 'Active Max' (>0.00) plays a significant role in predicting 'BENIGN'.

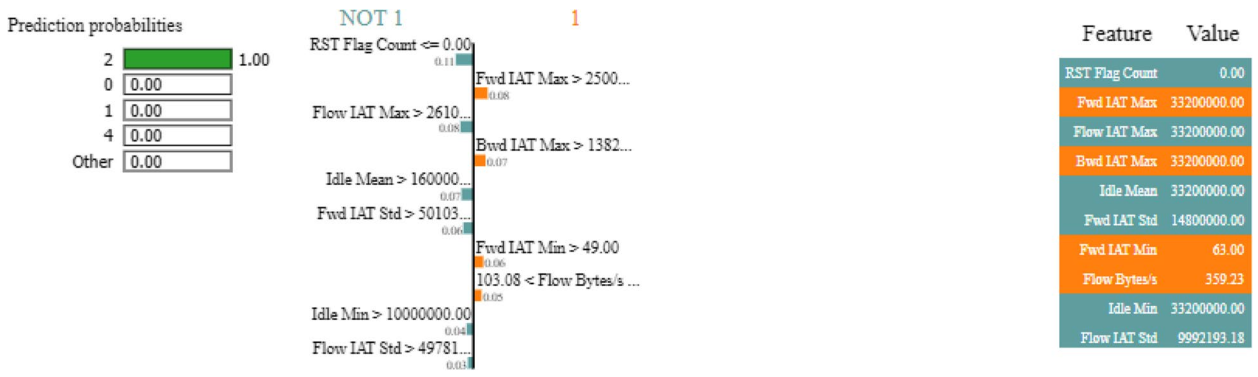
Figure 27 depicts a model's predictive probability for a multi-class classification problem and distinguishes between network attack events. The central part of the figure shows the contribution of various features to 'normal' prediction: illustrating how an expected decision was made. The most noticeable features are 'RST Flag Count' =< 0.00 led to standard classification; and 'Flow IAT Max' > 210.0 and 'Fwd IAT Max' > 195.2 were predicted to represent attack; 'Idle Max' 16000.0 and 'Fwd IAT Std' > 5015.0 were predicted as average prediction; 'Idle Min' > 10000000.0 and 'Flow IAT Std' > 4971.0 towards standard classification, support usual classification; and 'Fwd IAT Min' =< 49.00 and 'Tot Bwd Pkts' > 0.0, from average to attack prediction.

The right-hand panel quantifies those features' values, putting the classification in clearer perspective: 'Rst Flt' 0.00 'Fwd IAT Max' (32000000.00), 'Flow IAT Max (32000000.00), 'IDLE MEAN' (32000000.00), 'IDLE MIN' (14880000.00), 'FWD IAT STD' (5015.00), 'FLOI IAT STD' (4971.00). These are the specific data points the model uses to make predictions.

Based on the critical values indicated in Fig. 28, 'Bwd Packet Length Std' > -0.60, 'Fwd PSH Flags' <= -0.21, 'Fwd Packet Length Max' > 0.17, 'min seg size forward' > 0.23 and 'SYN Flag Count' <= -0.21 are the key features that predict a "DoS Golden Eye", whereas for the prediction of other classes, features like, 'Init Win bytes backward' <= 1.76 and 'Idle Max' > 0.02 are essential. Analyzing such critical features for each class and the diversity of features used for different predictions can help us understand the boundary in terms of the key features that drive the machine learning model to a particular class. Other features drive predictions to different classes.

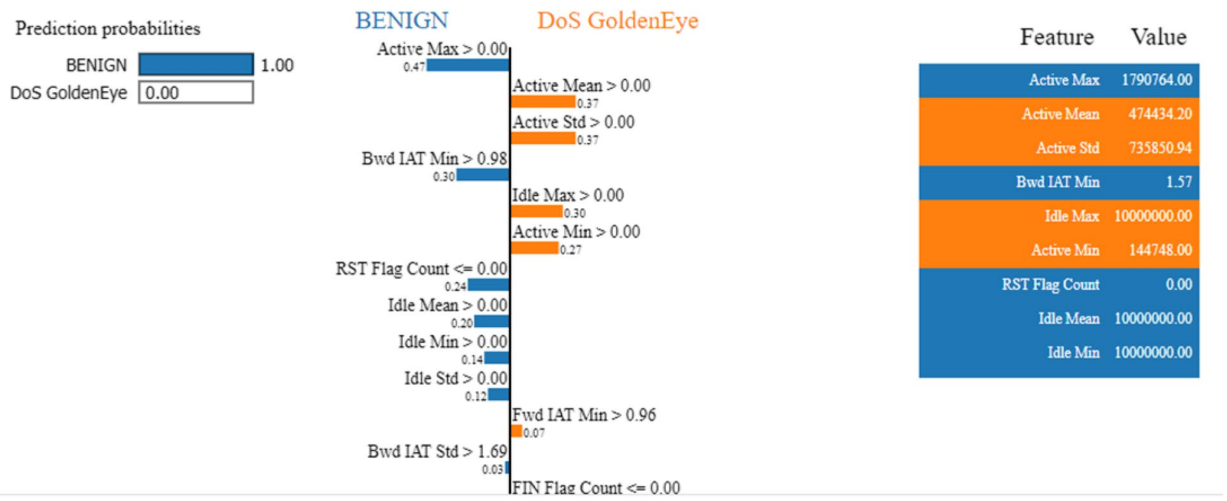
The right panel shows the feature values with their quantified interpretation: namely 'Bwd Packet Length Std': -0.60, 'Fwd PSH Flags': -0.21, 'Fwd Packet Length Max': 0.17, 'min seg size forward': 0.23, 'SYN Flag Count': -0.21, 'Idle Min': -0.50, 'Bwd IAT Mean': 3.90, 'Active Std': 18.91, 'Bwd IAT Total': 1.88, and 'Init Win bytes backward': 1.76. The quantification of important features adds a new layer of insight to the model's decision-making.

These plots vividly demonstrate what the model predicted, the elements that influenced the prediction, and the values of specific features that contributed to the classification. Such highly informative visualizations are critical to understanding and explaining model behavior, especially when those models tackle complex tasks such as network intrusion detection, where both accuracy and transparency are desired.



Feature	Value
RST Flag Count	0.00
Fwd IAT Max	33200000.00
Flow IAT Max	33200000.00
Bwd IAT Max	33200000.00
Idle Mean	33200000.00
Fwd IAT Std	14800000.00
Fwd IAT Min	63.00
Flow Bytes/s	359.23
Idle Min	33200000.00
Flow IAT Std	9992193.18

Fig. 27 Local explanation model's predictive probabilities for a multi-class classification task distinguishing between different network attack events



Feature	Value
Active Max	1790764.00
Active Mean	474434.20
Active Std	735850.94
Bwd IAT Min	1.57
Idle Max	10000000.00
Active Min	144748.00
RST Flag Count	0.00
Idle Mean	10000000.00
Idle Min	10000000.00

Fig. 28 Local explanation model's predictive probabilities against 20 feature importance for a multi-class classification task distinguishing between different network attack events

Comparative study

This section compares the methods and feature algorithms to verify the performance of the proposed HAEnID module in terms of using all features and 20 feature selection processing. Also, it was compared with the state-of-the-art intrusion detection methods to demonstrate the advantages of the proposed HAEnID IDS.

Comparison with classical methods

In this comparative study, we evaluated the performance of machine learning models on accuracy, precision, recall, and F1 score as shown in Fig. 29. Among the traditional models, LIBSVM showed excellent performance with an accuracy of 90.14% and recall of 82.44%, which shows its effectiveness in identifying instances for different classes. Although SVM shows slightly lower accuracy than

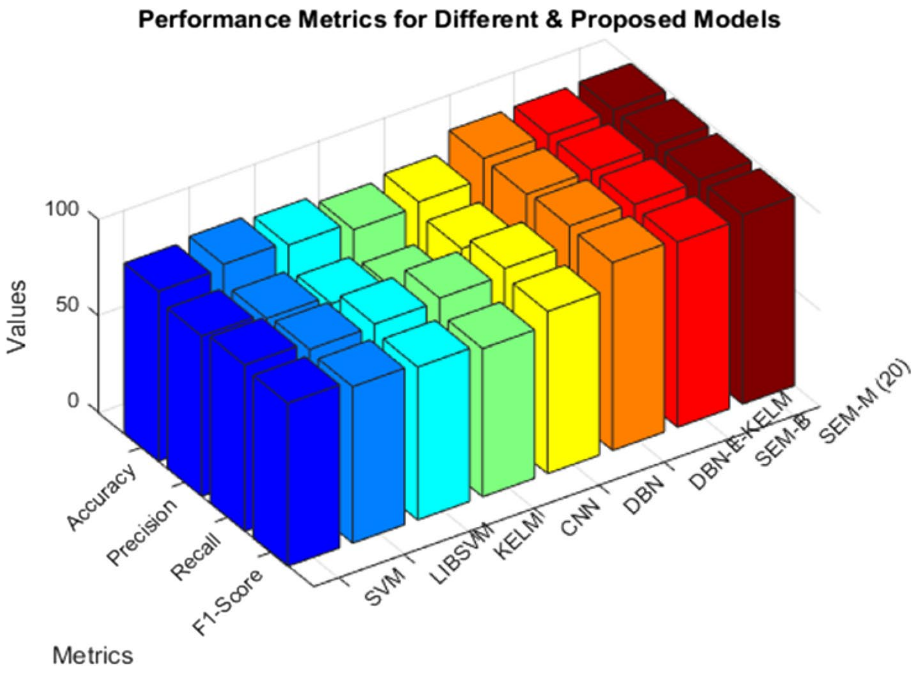


Fig. 29 The comparative performance metrics for different and proposed models' architecture for Binary and Multi-class with full and 20 feature selection

LIBSVM, it demonstrated an excellent precision score of 83.14%, indicating its ability to minimize false positives. SVM and LIBSVM performed well across all metrics and better than DBN-E-KELM and SEM-M (20). Models like CNN and KELM suggest potential improvement areas that are lower in performance than other models.

At the same time, the proposed model SEM-M (20) shows performance with 98.17% accuracy and 98.21% precision, and the proposed model SEM-B also shows results that compare with a small percentage to SEM-M (20). The study results show a clear comparison for selecting appropriate models for the problem and efficiency of feature extraction by deep learning techniques and ensemble methods.

Comparison with state-of-the-art methods

To evaluate the performance of the proposed **HAEnID** model, we need to examine the standard measures of accuracy, precision, recall, and the F1 score against the existing state-of-the-art methods, as shown in Table 10.

The SEM-B model achieves 97.44% accuracy and balance precision-recall, reflected by an F1-score of 96.06%. While the SEM-M model shows incremental improvements in accuracy (98.01%), its F1-Measure was slight to 97.18%. The BMA-B model's performance is noteworthy; however, its F1-Measure of 97.19% slightly lags behind its BMA-M counterpart, which boasts an F1-Measure of 97.49%. A substantial improvement is observed with the CEM-B model, yielding an F1-measure of 97.28%. Despite its lower F1-Measure of 96.55%, the CEM-M model still presents itself as a robust model within the context of these datasets.

The BMA-M 20 features selection model outperforms all others with an exceptional F1 measure of 98.79%, underscoring its balanced precision and recall. This performance contrasts with the drop-down F1 measures observed in standard DNNs, MLPs, and RFs [61–63], which demonstrate less efficacy, particularly in recall. The limited performance of the AE-MLP, AE-RL, and CAFE-CNN models [64, 68, 70] further reinforces the superior accuracy and balance of the 20-feature selection proposed models.

Quantitatively, the ensemble variants BMA-B, SEM-B, and CEM-M perform better by scoring higher accuracies, precisions, recalls, and average F1 scores on the CIC-IDS2017 dataset. For instance, BMA-B achieved an accuracy of 98.01%, a precision of 98.04%, a recall of 98.01%, and an F1-score of 98.01. This is not the case

Table 10 The table compares the proposed ensemble model with the state-of-the-art models using the same datasets regarding Accuracy, Precision, Recall, and F-Score

Model	Dataset	Accuracy	Precision	Recall	F1
MLP [61]	CIC-IDS2017	95	95.46	94.51	94.98
DNN [62]	NSL-KDD	95.38	97.73	98.12	97.83
DNN [62]	NSL-KDD	97.26	97.73	98.12	97.83
DNN [62]	NSL-KDD	96	94.12	96	95.75
RF [63]	NSL-KDD	97	90.72	61.75	96.61
MLP [63]	NSL-KDD	74	85.57	72.93	77.43
AE-RL [64]	NSL-KDD	80.16	79.74	80.16	79.40
GMM-WGAN-IDS [64]	NSL-KDD	86.59	88.55	86.59	86.88
AE [65]	NSL-KDD	/	87.85	82.04	81.21
AESMOTE [66]	NSL-KDD	82.09	/	/	82.43
LCVAE [67]	NSL-KDD	85.51	/	68.90	80.78
CAFE-CNN [68]	NSL-KDD, CIC-IDS2017	83.34	85.35	83.44	82.60
I-SiamIDS [69]	NSL-KDD CIDDS-001	80.00	/	/	68.34
AE-MLP [70]	CIC-2019	98.34	97.91	98.48	98.18
SEM-B	CIC-IDS2017	**97.44**	**97.06**	**97.45**	**96.06**
BMA-B	CIC-IDS2017	**98.01**	**98.01**	**98.01**	**97.18**
CEM-B	CIC-IDS2017	**98.05**	**98.05**	**98.05**	**97.28**
SEM-M	CIC-IDS2017	**97.44**	**97.06**	**97.45**	**96.06**
BMA-M	CIC-IDS2017	**97.94**	**97.94**	**97.95**	**97.49**
CEM-M	CIC-IDS2017	**97.25**	**97.26**	**97.25**	**96.55**
SEM-M (20)	CIC-IDS2017	**98.17**	**98.21**	**98.17**	**98.18**
BMA-M (20)	CIC-IDS2017	**98.79**	**98.79**	**98.79**	**98.79**
CEM-M (20)	CIC-IDS2017	**96.76**	**96.84**	**96.76**	**96.15**

with other novel models, such as MLP, DNN, and RF, which achieved lower accuracies, precisions, recalls, and F1 scores on the CIC-IDS2017 NSL-KDD dataset compared to the proposed ensemble model.

Qualitatively, this indicates the ability of the proposed model to deal with the complexity of the dataset well, especially when imbalanced data is present. The ensemble methods applied - SEM-B, with an F1-score of 97.44%, and BMA-M, with an accuracy of 98.07% - show a balanced ability to maintain high recall while maintaining low false positives - an essential aspect of an IDS. Other models, such as AE-MLP and AE-RL, have produced excellent and promising results on NSL-KDD with F1-scores of 98.02%, 88.67%, and 79.48%, respectively. However, they perform less effectively on the more challenging CIC-IDS2017 dataset.

In general, our proposed model performs not only higher scores than other methods in quantitative evaluation by all evaluation metrics but also better in qualitative evaluation. From our simulated experiments, our method shows excellent capability at improving efficiency while lowering the false alarm rate of IDS. It also distinguishes more variations of attacks and is thus more applicable and reliable. Thus, these findings advocate for proposed models in scenarios demanding precise and balanced intrusion detection, as their performance exceeds conventional machine learning approaches.

Conclusion

The study develops a new IDS (combined multi-classifier model) for integrated detection and classification of cyber threats called the HAEnID model. The experimental findings illustrate that the model performs very well in accuracy, with a high detection and classification rate for different cyber threats and a moderate false positive rate in "all types of network states". The combined model also maintains a good trade-off between the multiple dimensions. The interpretability of the new machine learning technique is further enhanced by modeling the SHAP and LIME algorithms. These contributions provide a novel solution to the limitations of IDS in general, as well as take a leading forward approach to prevent, detect, and analyze new unknown cyber threats, such as those that could not be solved by previous methods, such as their inability to achieve a good trade-off between multiple dimensions, such as accuracy, and their adaptability and explainability. HAEnID achieves this by leveraging the strength of the three different classifiers with their various dimensions and modeling approaches and with a dynamic adaptive learning capability of a core combined model to change in network states constantly. The multi-classifier combined method developed in this study to detect and classify intrusions, i.e., HAEnID, is a state-of-the-art model, as it performs even better than its predecessor IDS approaches. Some promising and pragmatic routes to expanding the scope of the HAEnID model and its applicability to other cybersecurity domains include future-proofing the model with more efficient adaptive dynamics (such as those used in the CEM). Specifically, imbuing the model with better detection of new and emerging threats and a more refined ability to respond to such attacks promptly could reduce the number of false positives and improve detection accuracy. A second potential direction is to use the HAEnID model in other cybersecurity domains, such as intrusion detection in cloud architecture, IoT networks, or industrial control systems, where the adaptive and explainable nature of the model could provide significant benefits. A third potential direction is to integrate new advanced deep-learning techniques and hybrid models that leverage supervised and unsupervised learning to improve the model's ability to detect novel attack vectors without pre-existing knowledge. Overall, HAEnID provides an excellent starting point for future research to show what is possible for intrusion detection and beyond.

Authors' contributions

Usman Ahmed: Conceptualization; Data curation; Formal analysis; Methodology; Writing - original draft; Software. Zheng Jiangbin: Supervisions; Investigation; Methodology; Writing - original draft; Writing - review & editing. Ahmad Almogren: Project administration; Investigation; Methodology; Writing - review & editing. Sheharyar Khan: Writing - review & editing; Software; Resources; Methodology. Muhammad Tariq Sadiq: Validation; Investigation; Writing - review & editing. Ayman Altameem: Funding Acquisition; Writing - review & editing; Software; Resources; Methodology. Ateeq Ur Rehman: Writing - review & editing; Methodology; Conceptualization.

Funding

This work was supported by King Saud University, Riyadh, Saudi Arabia, through Researchers Supporting Project number RSP2024R498.

Data availability

The dataset used in this study is publicly available at https://www.unb.ca/cic/datasets/ids-2017.html https://www.kaggle.com/datasets/dhoogla/cicids2017.

Declarations

Consent for publication

For this type of study, informed consent is not required.

Competing interests

The authors declare no competing interests.

Received: 3 July 2024 Accepted: 4 October 2024
Published online: 21 October 2024

References

1. Kizza JM (2024) System intrusion detection and prevention. In: Guide to computer network security. Springer, Verlag London, p 295–323

2. Ahmad Z, Shahid Khan A, Wai Shiang C, Abdullah J, Ahmad F (2021) Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Trans Emerg Telecommun Technol 32(1):e4150
3. Hnamte V, Hussain J (2023) Dependable intrusion detection system using deep convolutional neural network: A novel framework and performance evaluation approach. Telematics Inform Rep 11:100077
4. Masdari M, Khezri H (2020) A survey and taxonomy of the fuzzy signature-based intrusion detection systems. Appl Soft Comput 92:106301
5. Panagiotou P, Mengidis N, Tsikrika T, Vrochidis S, Kompatsiaris I (2021) Host-based intrusion detection using signature-based and ai-driven anomaly detection methods. Inf Secur Int J 50(1):37–48
6. CHAHIRA J (2019) Model for improving performance of network intrusion detection based on machine learning techniques. PhD thesis, Kabarak University
7. Liu Q, Hagenmeyer V, Keller HB (2021) A review of rule learning-based intrusion detection systems and their prospects in smart grids. IEEE Access 9:57542–57564
8. Moustafa N, Slay J (2015) Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In: 2015 military communications and information systems conference (MilCIS). IEEE, Canberra, p 1–6
9. Moustafa N, Koroniotis N, Keshk M, Zomaya AY, Tari Z (2023) Explainable intrusion detection for cyber defences in the internet of things: Opportunities and solutions. IEEE Commun Surv Tutorials 25(3):1775–1807
10. Dao TN, Van Le D, Tran XN (2023) Optimal network intrusion detection assignment in multi-level iot systems. Comput Netw 232:109846
11. Bhavsar M, Roy K, Kelly J, Olusola O (2023) Anomaly-based intrusion detection system for iot application. Discover Internet Things 3(1):5
12. Baldini G, Amerini I (2022) Online distributed denial of service (ddos) intrusion detection based on adaptive sliding window and morphological fractal dimension. Comput Netw 210:108923
13. Dong B, Wang X (2016) Comparison deep learning method to traditional methods using for network intrusion detection. In: 2016 8th IEEE international conference on communication software and networks (ICCSN). IEEE, Beijing, p 581–585
14. Arshad J, Azad MA, Abdeltaif MM, Salah K (2020) An intrusion detection framework for energy constrained iot devices. Mech Syst Signal Process 136:106436
15. Liu H, Zhong C, Alnusair A, Islam SR (2021) Faixid: A framework for enhancing ai explainability of intrusion detection results using data cleaning techniques. J Netw Syst Manag 29(4):40
16. Boudaoud K, Labiod H, Boutaba R, Guessoum Z (2000) Network security management with intelligent agents. In: NOMS 2000. 2000 IEEE/IFIP Network Operations and Management Symposium 'The Networked Planet: Management Beyond 2000' (Cat. No. 00CB37074). IEEE, Honolulu, p 579–592
17. Huang W, An Y, Du W (2010) A multi-agent-based distributed intrusion detection system. In: 2010 3rd international conference on advanced computer theory and engineering (ICACTE), vol 3. IEEE, Chengdu, p V3–141
18. Antwarg L, Miller RM, Shapira B, Rokach L (2021) Explaining anomalies detected by autoencoders using shapley additive explanations. Expert Syst Appl 186:115736
19. Jose J, Jose DV (2023) Deep learning algorithms for intrusion detection systems in internet of things using cic-ids 2017 dataset. Int J Electr Comput Eng (IJECE) 13(1):1134–1141
20. Yin C, Zhu Y, Fei J, He X (2017) A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access 5:21954–21961
21. Kussul N, Shelestov A, Sidorenko A, Skakun S, Veremeenko Y (2003) Intelligent multi-agent information security system. In: Second IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2003. Proceedings. IEEE, Lviv, p 120–122
22. Hedin Y, Moradian E (2015) Security in multi-agent systems. Procedia Comput Sci 60:1604–1612
23. Talib AM, Atan R, Abdullah R, Murad MAA (2011) Multi agent system architecture oriented prometheus methodology design to facilitate security of cloud data storage. J Softw Eng 5(3):78–90
24. Sahai A, Morin C (1998) Towards distributed and dynamic networks management. In: NOMS 98 1998 IEEE Network Operations and Management Symposium, vol 2. IEEE, New Orleans, p 455–464
25. Bhati NS, Khari M (2022) A new ensemble based approach for intrusion detection system using voting. J Intell Fuzzy Syst 42(2):969–979
26. Hossain MA, Islam MS (2023) Ensuring network security with a robust intrusion detection system using ensemble-based machine learning. Array 19:100306
27. Abbas A, Khan MA, Latif S, Ajaz M, Shah AA, Ahmad J (2022) A new ensemble-based intrusion detection system for internet of things. Arab J Sci Eng 1–15
28. Alotaibi Y, Ilyas M (2023) Ensemble-learning framework for intrusion detection to enhance internet of things' devices security. Sensors 23(12):5568
29. Sajid M, Malik KR, Almogren A, Malik TS, Khan AH, Tanveer J, Rehman AU (2024) Enhancing intrusion detection: a hybrid machine and deep learning approach. J Cloud Comput 13(1):123
30. Colledanchise M, Ögren P (2016) How behavior trees modularize hybrid control systems and generalize sequential behavior compositions, the subsumption architecture, and decision trees. IEEE Trans Robot 33(2):372–389
31. Bhati BS, Dikshita, Bhati NS, Chugh G (2022) A comprehensive study of intrusion detection and prevention systems. Wireless Commun Secur 11:115–142
32. Asif M, Abbas S, Khan MA, Fatima A, Khan MA, Lee SW (2022) MapReduce based intelligent model for intrusion detection using machine learning technique. J King Saud Univ Comp Inform Sci 34(10):9723–9731. https://doi.org/10.1016/j.jksuci.2021.12.008
33. Al Obaidli A, Mansour D, Shafi'i MA, Halima NB, Al-Ghushami A (2023) Machine learning approach to anomaly detection attacks classification in iot devices. In: 2023 1st International Conference on Advanced Innovations in Smart Cities (ICAISC). IEEE, Jeddah, p 1–6
34. Abbasi F, Naderan M, Alavi SE (2021) Anomaly detection in internet of things using feature selection and classification based on logistic regression and artificial neural network on n-baiot dataset. In: 2021 5th International Conference on Internet of Things and Applications (IoT). IEEE, Isfahan, p 1–7
35. Khaleefah AD, Al-Mashhadi HM (2023) Detection of iot botnet cyber attacks using machine learning. Informatica 47(6):54–64
36. Istratova E, Grif M, Dostovalov D (2021) Application of traditional machine learning models to detect abnormal traffic in the internet of things networks. In: International Conference on Computational Collective Intelligence. Springer, Cham, p 735–744
37. Nkoro EC, Nwakanma CI, Lee JM, Kim DS (2024) Detecting cyberthreats in metaverse learning platforms using an explainable DNN. Internet Things 25:101046
38. Bhati NS, Khari M (2021) A new intrusion detection scheme using catboost classifier. In: Forthcoming Networks and Sustainability in the IoT Era: First EAI International Conference, FoNeS–IoT 2020, Virtual Event, October 1-2, 2020, Proceedings 1. Springer, Cham, p 169–176
39. Bhati NS, Khari M (2021) Comparative analysis of classification based intrusion detection techniques. In: 2021 5th International Conference on Information Systems and Computer Networks (ISCON). IEEE, pp 1–6
40. Bhati NS, Khari M (2022) An ensemble model for network intrusion detection using adaboost, random forest and logistic regression. In: Applications of Artificial Intelligence and Machine Learning: Select Proceedings of ICAAAIML 2021. Springer, pp 777–789
41. Guo C, Ping Y, Liu N, Luo SS (2016) A two-level hybrid approach for intrusion detection. Neurocomputing 214:391–400
42. Malhi A, Gao RX (2004) Pca-based feature selection scheme for machine defect classification. IEEE Trans Instrum Meas 53(6):1517–1525
43. Alshaher H (2021) Studying the effects of feature scaling in machine learning. PhD thesis, North Carolina Agricultural and Technical State University
44. Liu H, Cocea M (2017) Semi-random partitioning of data into training and test sets in granular computing context. Granul Comput 2:357–386
45. Dwivedi A, Mishra D, Kalra P (2006) Handling uncertainties-using probability theory to possibility theory. Mag IIT Kanpur 7(3):1–12
46. Stiawan D, Idris MYB, Bamhdi AM, Budiarto R et al (2020) Cicids-2017 dataset feature analysis with information gain for anomaly detection. IEEE Access 8:132911–132921

47. Agarwal A, Sharma P, Alshehri M, Mohamed AA, Alfarraj O (2021) Classification model for accuracy and intrusion detection using machine learning approach. PeerJ Comput Sci 7:e437
48. Kim G, Lee S, Kim S (2014) A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. Expert Syst Appl 41(4):1690–1700
49. Liao Y, Vemuri VR (2002) Use of k-nearest neighbor classifier for intrusion detection. Comput Secur 21(5):439–448
50. Slack D, Hilgard S, Jia E, Singh S, Lakkaraju H (2020) Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society. ACM, New York, p 180–186
51. Chicco D, Jurman G (2020) The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. BMC Genomics 21:1–13
52. Neupane S, Ables J, Anderson W, Mittal S, Rahimi S, Banicescu I, Seale M (2022) Explainable intrusion detection systems (x-ids): A survey of current methods, challenges, and opportunities. IEEE Access 10:112392–112415
53. Abou El Houda Z, Brik B, Khoukhi L (2022) "Why should i trust your ids?": An explainable deep learning framework for intrusion detection systems in internet of things networks. IEEE Open J Commun Soc 3:1164–1176
54. Hariharan S, Rejimol Robinson R, Prasad RR, Thomas C, Balakrishnan N (2023) Xai for intrusion detection system: comparing explanations based on global and local scope. J Comput Virol Hacking Tech 19(2):217–239
55. Krishna S, Han T, Gu A, Pombra J, Jabbari S, Wu S, Lakkaraju H (2022) The disagreement problem in explainable machine learning: A practitioner's perspective. arXiv preprint arXiv:220201602. Available at: https://arxiv.org/abs/2202.01602
56. Wang M, Zheng K, Yang Y, Wang X (2020) An explainable machine learning framework for intrusion detection systems. IEEE Access 8:73127–73141
57. Kakogeorgiou I, Karantzalos K (2021) Evaluating explainable artificial intelligence methods for multi-label deep learning classification tasks in remote sensing. Int J Appl Earth Obs Geoinformation 103:102520
58. Zhou X, Wen H, Li Z, Zhang H, Zhang W (2022) An interpretable model for the susceptibility of rainfall-induced shallow landslides based on shap and xgboost. Geocarto Int 37(26):13419–13450
59. Lubo-Robles D, Devegowda D, Jayaram V, Bedle H, Marfurt KJ, Pranter MJ (2020) Machine learning model interpretability using shap values: Application to a seismic facies classification task. In: SEG international exposition and annual meeting. SEG, Tulsa, p D021S008R006
60. Gramegna A, Giudici P (2021) Shap and lime: an evaluation of discriminative power in credit risk. Front Artif Intell 4:752558
61. Perez-Diaz JA, Valdovinos IA, Choo KKR, Zhu D (2020) A flexible sdn-based architecture for identifying and mitigating low-rate ddos attacks using machine learning. IEEE Access 8:155859–155872
62. Hussain J, Hnamte V (2021) Deep learning based intrusion detection system: Software defined network. In: 2021 Asian Conference on Innovation in Technology (ASIANCON). IEEE, Pune, p 1–6
63. Najar AA, Manohar Naik S (2022) Ddos attack detection using mlp and random forest algorithms. Int J Inf Technol 14(5):2317–2327
64. Cui J, Zong L, Xie J, Tang M (2023) A novel multi-module integrated intrusion detection system for high-dimensional imbalanced data. Appl Intell 53(1):272–288
65. Ieracitano C, Adeel A, Morabito FC, Hussain A (2020) A novel statistical analysis and autoencoder driven intelligent intrusion detection approach. Neurocomputing 387:51–62
66. Ma X, Shi W (2020) Aesmote: Adversarial reinforcement learning with smote for anomaly detection. IEEE Trans Netw Sci Eng 8(2):943–956
67. Xu X, Li J, Yang Y, Shen F (2020) Toward effective intrusion detection using log-cosh conditional variational autoencoder. IEEE Internet Things J 8(8):6187–6196
68. Shams EA, Rizaner A, Ulusoy AH (2021) A novel context-aware feature extraction method for convolutional neural network-based intrusion detection systems. Neural Comput Appl 33(20):13647–13665
69. Bedi P, Gupta N, Jindal V (2021) I-siamids: an improved siam-ids for handling class imbalance in network-based intrusion detection systems. Appl Intell 51(2):1133–1151
70. Singh A, Jang-Jaccard J (2022) Autoencoder-based unsupervised intrusion detection using multi-scale convolutional recurrent networks. arXiv preprint arXiv:220403779. Available at: https://arxiv.org/abs/2204.03779

Publisher's Note