IJMIC Vol. 3, Issue 1, 1 - 27

Review Article

Network Traffic Classification using Machine Learning: A Review of Decision Tree, Random Forest, and SVM Approaches on CIC-IDS2017

Ahmad Razin Mohd Zaidi[1], Gohar Rahman[1], Mehmood Ahmed[2]

[1]Faculty of Computing and Informatics, Jalan UMS, 88400 Kota Kinabalu, Sabah, Universiti Malaysia Sabah

[2]Department of Information Technology, The University of Haripur, 22620 Khyber-Pakhtunkhwa, Pakistan

*Corresponding author: Gohar Rahman, gohar_315@ums.edu.my

Received: [Jan 10, 2026]
Accepted: [April 10, 2026]
Published: [July 30, 2026]

Abstract – Cyber-attacks are proliferating rapidly, and networks today are increasingly complex. For this reason, the ability to sort and locate intrusions in network traffic is critical. Researchers have employed methods based on machine learning (ML) as the conventional automated and signature focused detection methods often fail to catch newly emerged attack patterns. The purpose of this work is categorized as the survey is to review state-of-the-art machine learning (ML) network traffic classification techniques, focusing on one specific dataset, i.e., the CIC-IDS 2017 dataset and provides an analysis of today's research. The study drew on the IEEE Xplore, Springer and Elsevier databases for conducting an extensive review of literature published between 2020 and 2024 having identified studies involving supervised machine learning methods like Decision Trees, Random Forest, Support Vector Machine (SVM) that detected desirable and undesirable traffic. Researchers have also reviewed examples of performance measures (e.g., accuracy, precision, recall and F1-score) in different studies to determine areas of strength/weakness or patterns. Studies show that Decision Trees are interpretable, Random Forest has good memory and generalization abilities and SVM is accurate, but it requires computational optimization in large datasets. Despite these benefits, challenges such as dataset imbalance, feature redundancy, and real-time implementation continue to remain unsolved. This paper has shown how machine learning can improve the network intrusion detection and challenging issues that need to be addressed in further. The obtained insights lead to a basis for an enhanced scalable and robust traffic classification model, contributing to the goals of current research in this domain.

Keywords: Network Traffic Classification, Machine Learning, CIC-IDS 2017, Intrusion Detection

1. INTRODUCTION

Due to the exponential growth in the importance of the internet and related communication technologies in the past ten years, network security has become an essential area of study. It uses intrusion detection systems (IDS), firewalls, and antivirus software to keep the network and everything on it safe in cyberspace [1]. Machine learning algorithms have the ability to learn from prior data and recognize new attack behaviours. Research have indicated that ensemble methods, such as Random Forest, produce

strong results and have potential for large-scale usage. Contrasted with ensemble approaches, Support Vector Machines (SVM) are effective on higher dimensional spaces. Decision Trees are a favourite of analysts due to their ease of analysability, although they are just as likely to overfit [2]. Overall, the aforementioned type of vulnerability to cyber threats has increased dramatically [3]. Furthermore, Intrusion Detection Systems (IDSs) have become a key aspect of cybersecurity, primarily serving the purpose of monitoring and analysing network traffic, while detecting and reporting malicious behaviour [3][4].

Historically, intrusion detection systems (IDSs) have primarily applied signature-based methods (also referred to as misuse or knowledge-based detection) to detect known attacks by searching for patterns and signatures recorded in a database [3], [4]. Although effective for known threats, with low false positive rates, these systems are inherently unable to detect novel or "zero-day" attacks that are not represented in their databases [3]. Anomaly-based intrusion detection systems, alternatively known as "behaviour-based IDS," build a model of how a system should typically behave and identify any significant deviations away from that model as potentially threatening. Their strength is the ability to identify new or unknown attacks. However, they often produce unacceptably high false positive rates and limit their use [3].

Researchers continue to seek improvements in IDS methods and, to date, the methods still face significant challenges, including high rates of false alarms, low rates of detection, and constraints regarding computational power and/or resources [4]. These systems struggle with today's complex, modern attacks and encrypted data [2], [4]. Rationalizing these challenges and the ever-growing complexities of cyberattacks, there has likely been a significant move towards the use of artificial intelligence (AI), specifically machine learning (ML) and deep learning (DL) technologies into network intrusion detection (NID). While the move toward AI does not completely address the numerous challenges associated with NID systems, the changes present some notable advantages for NID systems, such as the ability to process data at an increased rate, one of the height aspects of new IDS methodology considering the exponential amount of data produced daily at extremely high speeds, as older systems typically fail to do. ML has improved pattern identification, allowing NID systems to identify possible minor indications of malicious activity by applying new network traffic classification and pattern identification techniques. The ability of machine learning algorithms to adapt provide intrusion detection systems with flexibility and the ability to learn the differences between normal and malicious activity. The ability to ascertain distinguishing features in problems that are difficult by machine learning amongst features that are unknown and the ability to explore encrypted data streams in the network. Machine learning and deep learning models have demonstrated the potential and promise to classify encrypted unmanaged network traffic [5].

2. BACKGROUND

Many network attacks are often dynamic and involve unpredictable behaviours which cannot be detected by traditional intrusion detection systems (IDS). Traditional approaches, like signature matching, have a hard time at detecting new or zero-day attacks, while anomaly-based systems typically generate high false-positive rates. In order to mitigate these restrictions, machine learning (ML) and deep learning (DL) approaches, which offer the advantage of learning traffic features automatically and achieving good performance in distinguishing between benign and malicious traffic behaviours, have been investigated by researchers [2], [3]. Such techniques contribute towards making detection more adaptive, require less manual configuration of rules and can be scaled out to detect real-time attacks.

Machine learning based IDS models can be broadly divided into supervised, unsupervised, and hybrid learning frameworks. Supervised models, including Decision Trees (DT), Random Forests (RF) and Support Vector Machines (SVM), are dependent on labelled datasets, for example, the CICIDS2017 to classify the network traffic [1], [6]. On the other hand, unsupervised and hybrid models utilize

clustering and deep architectures to learn non-linear association in high-dimensional data [4], [7]. Both these learning paradigms have their own strengths while DT provide the simple interpretation, RF inherit the robust ensemble and SVM assure maximum-margin decision boundaries, applicable for smaller and homogeneously distributed dataset.

2.1 Network Security and Intrusion Detection Systems (IDS)

Network security is a collection of technologies, protocols, and operational practices aimed at protecting communication systems, data integrity, and confidentiality against unauthorized access, cyberattacks, or abuse [4]. In this context, Intrusion Detection Systems (IDS) are recognized as a critical proactive defence type that constantly monitors traffic data as well as host activities to discover abnormal behaviours or violations of defined policies [8].

Based on types of its technologies, IDS can be divided into two main categories, that is, signature-based and anomaly-based. Signature: Signature-based IDS use known signatures of attacks to identify incoming threats with high accuracy low false positives. Nevertheless, they become impotent to new or emerging attacks that do not have their respective signatures [2]. In contrast, anomaly-based IDS establish a baseline of "normal" behaviour and identify deviations as likely intrusions. Although those systems are capable of recognizing zero-day or unknown attack, they usually achieve high false-positive rate because they model normal patterns from sample data, which suffer from diverse (natural) variations of network data and require huge computational resources [3], [9].

In practice, each has its own strengths and weaknesses in terms of accuracy, flexibility, and interpretability. While signature-based systems provide high accuracy and very low chance of false alarm they can be easily bypassed with new attacks, conversely, anomaly-based systems manage to detect new attack patterns but with the expense of stability and false alarm rate. Motivated by these limitations, the integration of machine learning (ML) and artificial intelligence (AI) techniques within IDS frameworks has provided dynamic detection which learns from the historical data and consequently adapting to newer threat patterns in real-time [1], [10].

Recent work adjusts how we assess the need for intelligent IDS that can scale to complex, high-throughput network environments without sacrificing accuracy and efficiency. ML-based systems utilizing ensembles methods, deep learning architectures, and features selection methods achieved significant gains in detection of known and unknown intrusion respectively [11], [12].

2.2 Machine Learning in IDS

Machine learning (ML) techniques have become a key feature for improving the capability of intrusion detection systems (IDS) on how to detect, classify, and prevent complex cyber threats. In contrast to static rule-based schemes, IDS driven by machine-learning models can automatically learn from network data, capture important traffic patterns, and generalize across attack variants without pre-defined signatures [1]. Not only do these models increase speed and reduced the reliance on human beings but also boost scalability when deployed in a large evolving network environment.

Various algorithms such as Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM) fall under used for traffic classification problems on well-known datasets like CIC-IDS2017. Moreover, DT algorithms are also the most intuitive regarding the interpretability that comes from a series or hierarchy of splits through features and decision lines, while RF is an ensemble of many DT that usually have a very high robustness and resistance to overfitting [6], [12]. SVM models in contrast, excel in distinguishing non-linear data boundaries, achieving high accuracy in binary classification problems though often at the expense of computational efficiency in high-dimensional spaces [13].

Similarly, feature engineering and feature selection are also important in achieving better performance for ML-based IDS. Information Gain (IG), Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA) are few techniques that are helping in the process to identify the most important traffic features with reduced redundancy to increase the model accuracy [7], [9] .Some studies, where these methods were used, have reported faster training and fewer false positives, thus increasing the applicability of ML models for real-time network monitoring.

Deep learning (DL) models have emerged as vigorous successors to traditional machine-learning (ML) algorithms, retaining the ability to apply ML algorithms to large data and also automating the process of feature extraction and recognizing temporal patterns. When adequate amounts of computing power and large datasets are available, CNNs, LSTMs, and hybrid CNN-LSTM frameworks consistently outperform classical models [7]. However, even though these models generally have a high detection accuracy, they require a lot of computational resources and are usually not interpretable. Therefore, recent works recommend light-weight hybrid models that combine shallow machine learning classifiers with deep feature extractors as a potential solution that are able to achieve an optimal trade-off between interpretability, scalability, and performance [14].

This hybrid perspective underscores the enduring relevance of traditional ML algorithms such as Decision Tree, Random Forest, and Support Vector Machine within modern IDS research. Their distinct trade-offs in learning complexity, computational cost, and detection accuracy are summarized in Table 1 which highlights the comparative strengths and weaknesses of these core algorithms [15].

Table 1: Key strengths and weaknesses the algorithms as applied to IDS

Algorithm	Strength	Weakness
Decision Tree	Fast learning and inference with interpretable hierarchical rules; provides high precision for known attack categories [1], [6].	Prone to overfitting and limited in detecting previously unseen or zero-day attacks [3].
Random Forest	A collection of trees gives you great accuracy and robustness and works well with large amounts of or missing data [11], [12].	Heavy on computations (many trees) and less clear (hard to interpret ensemble decisions) [16].
Support Vector Machine	Achieves high accuracy in separating non-linear data by maximizing decision margins and performs effectively on moderate-to-large datasets [13].	Long training times on large datasets and sensitivity to kernel and hyperparameter selection [17].

But there are still a few problems that keep ML and DL from being widely used in IDS. The high cost of computing is a big problem because training and using these models often requires a lot of processing power and memory, which makes them less useful for real-time applications in places where resources are limited. Scalability is another problem because IDS needs to be able to handle the rapidly growing amount of traffic on the network without losing accuracy in detection. Moreover, the lack of interpretability in complex ML and especially DL models make it hard for security analysts to understand how these systems work. This is because they often act like "black boxes," giving little information about how detection decisions are made [18].

To provide a structured comparison of recent studies in machine learning-based intrusion detection systems, Table 2 summarizes key approaches, datasets, advantages, and limitations identified in the literature.

Table 2: Comparative Summary of Recent Studies on Machine Learning-Based IDS using CIC-IDS2017.

Reference	Approach Used	Dataset Used	Advantages	Limitations
Maseer et al. (2021) [6]	Decision Tree, Random Forest, SVM	CIC-IDS2017	High detection accuracy (DT and RF >99%). Comprehensive benchmarking across multiple ML models and strong evaluation using standard metrics	SVM showed lower performance that shows limited focus on real-time deployment. Have potential overfitting to dataset
Talukder et al. (2024) [11]	Decision Tree, Random Forest, Ensemble Models with Oversampling	CIC-IDS2017	Extremely high accuracy (~99.99%). Effective handling of class imbalance using oversampling and feature embedding	High computational cost which the results may not generalize to real-world traffic and reliance on heavy preprocessing
Xu et al. (2025) [19]	CNN, MLP, One-Class SVM (OCSVM)	CIC-IDS2017	Good balance between precision and recall. Effective for anomaly detection and unseen attacks	Deep learning models require high computational resources. Lower interpretability and dependent on feature quality
Saidane et al. (2024) [20]	Extra Trees, VGG19 Ensemble (Deep Learning + ML)	CIC-IDS2017	High detection performance (~99.26%); hybrid approach improves feature representation and classification	High model complexity. Resource-intensive and limited interpretability for security analysts
Rodríguez (2022) [21]	Random Forest, SVM, Decision Tree	CIC-IDS2017	Clear demonstration of classical ML effectiveness. Simple and interpretable models and also reproducible preprocessing workflow	Limited innovation. Does not address dataset imbalance deeply and lacks advanced optimization techniques

The comparison highlights that while many studies achieve high detection accuracy on CIC-IDS2017, challenges such as computational complexity, dataset imbalance, and limited real-world generalization remain significant.

2.3 Deep Learning in IDS

Recent years have seen a growing focus on deep-learning (DL) approaches to underpin next-generation intrusion detection systems (IDS), moving the focus away from handcrafted (low-dimensional) feature to automated (high-dimensional) pattern learning. Various types of neural architectures including Convolutional Neural Networks (CNNs), Long Short-Term Memory networks (LSTMs), hybrid CNN-LSTM architectures, and more recently Transformer or Graph Neural Network (GNN) models have been proposed to extract spatial, temporal, and topological features from traffic data on scales [22],

[23], [24]. While these models achieve excellent detection performance on standard datasets, this is only true in scenarios with enough labelled data and available computational resources.

DL-based IDS, while having the advantage of high accuracy, can have a number of operational drawbacks: they are generally resource-intensive and have high training/ inference time, and also suffer from a low interpretability and transparency of decision logic. This makes the real-time deployment across high-throughput enterprise networks non-trivial[11].Recently, many studies focus on lightweight hybrid models that combine (shallow)-ML classifiers (e.g., RF, SVM, etc.) with deep feature extractors so that it tries to improve the trade-off between accuracy, resource consumption, and interpretability of the detection [16].

A new extension of DL-IDS research includes the use of Graph Neural Networks (GNNs), which treat network traffic data and host relationships as graph-structured data. In comparison to traditional DL-based models, these systems capture inter-flow or inter-session dependencies better, and can also be potentially employed for detection of complex, multi-vector attacks and IoT/industrial-scale scenarios [25][26]. Meanwhile, the joint exploration on adversarial robustness, continual learning, and cross-domain transferability shows that there is a strong desire for IDS models enabled to cope with changing environments (e.g., changing tactics by attackers, or, shifting network distributions) [27].

Together these advances suggest that though deep-learning will continue to be a cornerstone of future IDS architecture, the best course forward is to create efficient, interpretable and adaptable hybrid systems. These will enabled deep feature learning with traditional classifiers, graph-oriented modelling and online adaptation in order to fulfil the requirements for real-world deployments.

2.4 Benchmark Datasets in IDS Research

To develop and evaluate ML/DL methods, researchers rely on publicly available intrusion detection datasets. Widely used benchmarks include NSL-KDD, CIC-IDS2017, and UNSW-NB15 datasets, which provide standardized evaluation environments for comparing model performance [4], [13].

NSL-KDD dataset (2009)-NSL-KDD dataset is an enhanced version of the KDD'99 dataset preventing the problems of duplicate records and has therefore balanced training and testing subsets. It comprises 41 features at the connection level e.g., protocol type, service and duration [28]. Although it is still a widely-used baseline for traditional ML models, its main limitation is that it lacks encrypted traffic and modern attack profiles, making it not very representative of real-world conditions [29].

The CIC-IDS2017 dataset (2017), created by the Canadian Institute for Cybersecurity, abstract realistic network behaviour and involves both benign and malicious traffic across different modern attacking classes such as DoS, DDoS, brute force, infiltration and botnets [10]. Based on packet captures, it provides a very rich dataset with 83 flow-based features (e.g., timestamps, packet counts, port numbers). But this has a heavy class imbalance and large data volume, which is costly in terms of computation resources to process [17], [27].

The UNSW-NB15 dataset (2015)- The UNSW-NB15 dataset has been created in 2015 and developed at the Australian Centre for Cyber Security (UNSW Canberra) and the dataset includes both actual and synthetic network traffic containing 49 flow attributes covering nine attack categories including exploits, worms and reconnaissance [4]. Although it still suffers from problems such as synthetic traffic generation and class imbalanced dataset, it is become a popular contemporary substitute for older datasets [22]. Table 3 summarizes the main strength and weakness of dataset that were review for this research.

Table 3: Summary of key attributes from dataset used

Dataset	Year	Features	Strength	Weakness
NSL-KDD	2009	41	Cleaned version of KDD'99 with no redundant records, balanced and widely used benchmark [4], [28].	Lacks modern attack varieties, no raw packet data and known to be outdated [29].
CIC-IDS2017	2017	83	Realistic, large-scale traffic with up-to-date attack types (DDoS, infiltration, brute-force); comprehensive flow features from real packet captures and widely adopted for ML/DL evaluation [10], [17].	Contains extreme class imbalance and very large size makes training costly [22], [27].
UNSW-NB15	2015	49	Includes modern attack types and normal traffic generated in a hybrid real/synthetic environment and captures exploits, worms, and reconnaissance activities [4].	Attack data includes synthetic elements, exhibits class imbalance and medium-scale data volume [22], [28].

2.5 Research Gaps and Challenges

Despite substantial progress in ML-driven Intrusion Detection Systems (IDS), several critical challenges remain that hinder their effectiveness and scalability in real-world deployments. These challenges span issues of generalization, data imbalance, computational efficiency, and dataset realism.

1. **Generalization to Real-World Traffic**
   Numerous models perform very well on benchmark datasets but cannot generalize to different network environments, indicating possible overfitting to dataset distributions. Layeghy and Portmann illustrate that no model under test maintained strong performance on multiple datasets, pointing included asymmetrical generalization. Their investigation revealed that models often harness characteristics specific to the dataset that do not generalize well to new domains [28].

2. **Class Imbalance and Uneven Detection Performance**
   Typically, datasets used for intrusion detection contain far more benign instances than malicious ones, making it difficult to remember attacks that are less common in the training dataset. The MDPI survey about DL in IDS, as well as other studies, demonstrate how severe class imbalance can be, when bias in the more commonly occurring class can lead to incorrect classifications of important instance, especially in situations when balancing or data augmentation are not carefully applied [22].

3. **Computational Efficiency, Scalability, and Model Drift.**
   To achieve real-time deployment of IDS, efficient models are needed to process high-speed, high-volume packets of network traffic with minimum latencies. Unfortunately, many ML and DL models (whether deep or textbook shallow architecture) are computationally cumbersome and out of reach for both large-scale deployments or the edge. As noted by Alashjaee (2025) and Azab (2024), to train complex deep learning models like CNN-LSTM architectures on CIC-IDS2017 datasets typically requires a huge amount of GPU resources, enough that it can be prohibitively expensive in both time and money [4], [23].And yet there is no route towards change. So lightweight, adaptive models coupled with online learning strategies might be essential in order to improve reliability and scalability under ever-changing network conditions.

4. **Dataset Limitations**
   While the benchmark datasets, like NSL-KDD, CIC-IDS2017, and UNSW-NB15 still play a significant role as the ground truth for evaluating IDS solutions. However, they themselves are not immune from vulnerabilities. NSL-KDD is obsolete, and CIC-IDS2017 and UNSW-NB15 has been enriched with synthetic traffic that delays generalization capability [29]. Layeghy et al. (2024) and Catillo et al. (2023) both claim these datasets, still used frequently today, do not reflect the current encrypted, IoT and 5G traffic forensics characteristics [10], [28]. Therefore, the scenario of continuous dataset evolution including periodical update and hybrid real-world data incorporation is presented.

5. **Generalization Across Different Setting**
   Models trained on one dataset, or in a certain environment, generally do not generalize well to other network conditions. This over-specialization undermines practical applicability. Recent reviews by Azab et al. (2024) and Xu & Liu (2025) prove that IDS models trained on CIC-IDS2017 achieve high accuracy in the corresponding domain while suffering heavy performance loss when dealing with UNSW-NB15 or real traffic data [4], [19]. This shows that developing transferable and domain-adaptative IDS architectures is a significant open problem. Promising directions for future work include federated learning, transfer learning and cross-domain adaptation to improve robustness and scalability under varying deployment conditions.

3. METHODOLOGY

3.1 Research Question

The purpose of this research is to provide a systematic overview of the most recent findings from machine learning-based network intrusion detection systems (NIDS) using the decision tree (DT), random forest (RF), and support vector machine (SVM) classifiers that were assessed using the CIC-IDS2017 dataset. Rather than proposing or creating a new detection model, the goal of this review was to collect all the previous works and synthesise them into one cohesive document. This will allow researchers and practitioners to see how researchers are trending toward using machine learning in NIDS, what methodologies researchers are using to evaluate these classifiers, and what future challenges need to be addressed in the literature.

Therefore, the following questions guide the review:

- **RQ1:** Which supervised machine learning models, specifically Decision Tree, Random Forest, and Support Vector Machine, demonstrate the strongest detection performance on the CIC-IDS2017 dataset as reported in prior studies? [30]

- **RQ2:** What trade-offs are identified in the literature between these models in terms of accuracy, interpretability, computational complexity, and scalability? [31]
- **RQ3:** What limitations are reported regarding generalisation, class imbalance, and real-world deployment of these models? [32]

These questions are consistent with recent systematic reviews that emphasise comparative evaluation and synthesis of existing experimental results rather than the development of new classifiers [1], [4].

I. **Detection performance:** A number of studies recently conducted indicate that classifier technology is capable of providing particularly high detection rates for intrusion detection systems (IDS) using data set such as CIC-IDS2017 and similar data sets. Long (2025) proposed the "Threat Aware" Support Vector Machine (SVM) and established a 95.6% success rate for detecting non-linear attacks within the CIC-IDS2017 dataset [33]. Well optimised Random Forests (RF), XGBoost and similar tree-based techniques generally achieve between 98%-99% accuracy on CIC-IDS2017, while advanced SVM approaches can achieve over 95% detection rates.

II. **Feature Relevance:** Feature reduction is a common process for enhancing the performance of classifiers. In their work, Halim (2021) created an algorithm based on genetic algorithms for feature selection (called GbFS) for intrusion detection systems, which selects the least number of features that still provide an accurate representation of an attack and performed algorithm testing was performed using CIRA-CIC-DoHBrw-2020, UNSW-NB15, and Bot-IoT, making GbFS capable of producing results with up to 99.80% accuracy for classifier development [34]. Achbarou (2025) performed an experimentation of feature selection via IoT Intrusion Detection Systems (CIC-IDS2017), it was found that application of XGBoost with 40 selected features provided an accuracy of approximately 99.95% [35].

III. **Imbalance Mitigation:** Achbarou (2025) applied an explicit two-phase balancing strategy for CIC-IDS2017 where they oversampled the minority class before splitting using SMOTE; then class weighting was used during training [35]. The findings from their research support the conclusion that both oversampling techniques, such as SMOTE, and class weighting methods assist IDS datasets. For example, according to Achbarou, by combining SMOTE and the use of class weights, the authors were able to achieve a very high level of accuracy with CIC-IDS2017 [35]. On the other hand, Hamidou obtained optimal levels of performance through SMOTE balancing of IDS datasets [36].

IV. **Model Comparison:** Recent findings have highlighted the importance of intense tuning and comparison between classifiers. Hamidou and Mehdi's (2025) comparison of Random Forest (RF), Extreme Gradient Boosting (XGBoost) and Deep Neural Networks (DNN) using IDS datasets demonstrated an effectiveness of 99.80% for Random Forest (RF) indicating superiority over all other models, which were optimized using Optuna [36].

3.2 Research Flow Diagram

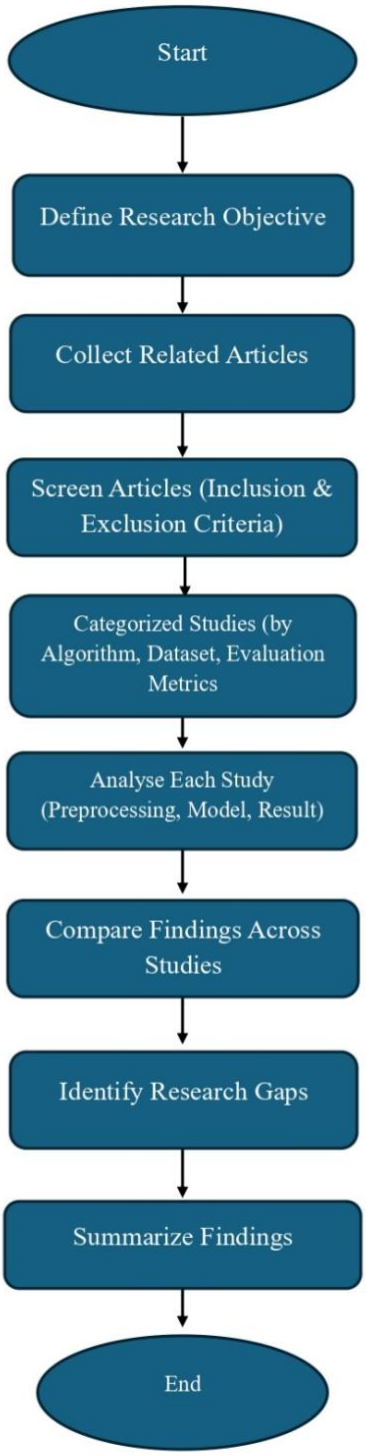

Figure 1: Research flow diagram.

This study follows a systematic procedure, as shown in Figure 1, to conduct the review of literature. The first step was to provide the research objective, which is an evaluation of machine learning algorithms and their application to network intrusion detection systems. Next, journals of interest were

recruited by databases and DOI articles. The journal articles were then reviewed based on the inclusion criteria and exclusion criteria chosen for recent publications. Next, the relevant articles selected were analysed based on research themes, either preprocessing method, features for model selection, algorithms, and evaluation measures. Each article was reviewed for in-app detail to examine the performance trends, differences in methods, and findings for each study. A comparative analysis was made across each selected study to determine the intersections of advantages and disadvantages, as well as the gap in research as it applied to the measures of the dataset imbalance, the certification of generalization to unseen attack, and model interpretability. Finally, a measured summary of the research was conducted to draw supporting conclusions to future directions in researching machine learning based intrusion detection systems.

3.3 Review Process and Synthesis Workflow

The process of reviewing these works was structured and transparent, ensuring that all works were covered in a systematic manner, analytical consistency across all works, and the reproducibility of the results. This workflow is consistent with previously established methodologies used in recent systematic literature reviews of intrusion detection systems based on machine learning [1], [2].

I. **First Phase:** In defining the objectives and scope of the review, it was decided to restrict this review to studies using supervised machine learning techniques that use Decision Tree, Random Forest, and Support Vector Machine models for the purpose of detecting network intrusions, and subsequently the CIC-IDS2017 dataset used in most of these studies. This scoping makes it possible to maintain comparability of the studies included in this review and helps to minimise fragmentation of methodology due to heterogeneous datasets and dissimilar learning paradigms [4].

II. **Second Phase:** The review process was to conduct a comprehensive search of the available literature using major academic databases. Such searches were performed using IEEE Xplore, SpringerLink, Elsevier ScienceDirect, and MDPI by combining keywords associated with the various types of intrusion detection systems, machine learning techniques, CIC-IDS2017 data set, and the classifiers chosen. The search was limited to peer-reviewed journal articles and conference proceedings to provide a degree of academic integrity for the retrieved literature, based on the recommendations provided by previous reviews [2], [3].

III. **Third Phase:** The actual screening and evaluation of the studies selected for the review based upon the predetermined inclusion/exclusion criteria. In this process, duplicate studies were removed from consideration, and the titles, abstract, and full text were reviewed to determine if each study met the appropriate evaluation criteria. Studies were removed from consideration if they did not apply machine learning-based intrusion detection systems, to the CIC-IDS2017 data set, or if empirical performance assessment did not take place on the study's findings. This filtering process reduced the potential for selection bias in the review process and improved the internal validity of the analysis of the review [1].

IV. **Fourth Phase:** Conducting a thematic analysis involved analysing each of the selected studies according to a set of predetermined dimensions, such as data preprocessing, selection of features, handling of class imbalance, model set up and evaluation metrics and drawing out commonalities between the studies' methods and advantages/disadvantages across those studies. The goal of conducting a thematic analysis was not to replicate the actual experimentation conducted by the authors of the studies but to identify the commonalities in the way the authors approached and solved their respective IDS problems, and the advantages and disadvantages of the methods used by the authors. This type of thematic analysis has been used in the past to inform the creation of IDS systematic reviews [6], [11].

V. **Fifth Phase:** A comparative synthesis was completed to synthesise the findings across all of the papers identified in the literature search and to determine how well they worked against each other. Specifically, the generalised detection performance of Decision Tree, Random Forest, and Support Vector Machine detectors was compared, as well as the resulting trade-offs between the three types of detectors with respect to accuracy versus interpretability versus computational cost, and the remaining unsolved challenges regarding generalisation, dataset dependence, and scalability were identified. This step of the synthesis allowed for the development of broader perspectives than just the results of each individual study and supported creating conclusions based on the accumulated evidence [28].

3.4 Rationale for Method and Model Selection

In the creation of an intrusion detection model for the CIC-IDS2017 dataset, this research utilizes Decision Tree (DT), Random Forest (RF) and Support Vector Machine (SVM) classifiers as part of developing the intrusion detection model. The decision to use these models is based on the complementary strengths, the high degree of usage of these models in the area of security research, and the historical record of effectiveness on other public IDS benchmarks.

1. **Random Forest (RF):** By combining a large number of low-quality decision trees together into one classifier called a Random Forest (RF), RF uses the strengths of each of those decision trees as well as their weaknesses to give us the best possible prediction result. Elashmawi et al. (2025) found that RF was significantly better performing than both DT and SVM for intrusion detection on an NSL-KDD dataset, achieving multi-class classification accuracy of 85.42% [37].
2. **Support Vector Machine (SVM):** Chosen for their theoretical strength in processing high-dimensional data and establishing optimal separating margins being highly ineffective when dealing with significantly imbalanced or noisy intrusion datasets, despite these challenges, Mari et al (2023), have established that SVMs can still be competitive, when feature tuning and adversarial testing are used correctly, in achieving detections of above 90% for the NSL-KDD Dataset Example[38]. SVMs, such as LS-SVM or one-class SVM versions, have strong decision boundaries and work well with data that has several dimensions. Xu et al. have investigated the use of One-Class SVMs for anomaly detection. They have determined that an OCSVM effectively balances precision and recall on both known and novel attacks [19].
3. **Decision Tree (DT):** Chosen for its transparent structure, low computational overhead, and strong classification accuracy in intrusion detection. As shown in Yadav et al. (2023), when implemented with a recursive feature elimination technique, DT models can achieve nearly 100% accuracy on public intrusion detection datasets such as KDD-99 and NSL-KDD. The advantages of DT's transparent tree-based logic were highlighted as being particularly advantageous to security analysts because they allow for the creation of usable and understandable models [39].

Furthermore, Genuario (2024) emphasize that although deep learning techniques are being widely utilized, conventional techniques for supervised learning, such as DT, RF, and SVM, have competitive performance, as well as the added advantage of being easier to interpret and implement in general operational environments [40].

4. BENCHMARK DATASETS IN INTRUSION DETECTION RESEARCH

4.1 Dataset Description and Attack Scenarios (CIC-IDS2017)

Table 4: Dataset description for CIC-IDS2017.

General Description	The Canadian Institute for Cybersecurity has developed a comprehensive network traffic dataset known as CICIDS-2017. It comprises approximately 2.83 million flow records, each of which is classified as benign or one of nine attack categories, and includes 79 features (packet/flow statistics) [41].
Attacks	The dataset contains many different types of attacks, such as Denial-of-Service (DoS), DDoS variations including Hulk, Slowloris, and GoldenEye, port scans, botnet traffic, FTP/SSH brute-force (Patator), SQL injection, XSS, infiltration, Heartbleed, and more [42].
Train/Test Splits	Researchers often divide CIC-IDS2017 into User Defined Training / Test Sets (custom), to evaluate the efficacy of machine learning in Practice. In this regard, researchers have considered the advantages and disadvantages of stratified splits and independence between training and test folds when conducting practical evaluations of IDS models [29].

4.2 Preprocessing Technique

Most studies apply extensive preprocessing to clean and normalize CICIDS-2017 data before modelling:

- **Data Cleaning and Noise Removal:** A common initial procedure when processing data sets includes removing any records (or files) that have an invalid or inconsistent entry (or value) in them. An example of this may occur for flow-based data sets where there might be records that were created due to an error during the capturing of network packets or because of packet loss. One study (Singh et al., 2024) [43] looking at what preprocessing can do for intrusion detection data sets found that preprocessing actions (like removing missing values, dealing with outliers, ensuring uniformity, etc.) greatly enhance the performance of any resulting machine learning processes/models.

- **Feature Reduction:** Many methods exist for reducing features to reduce the size of a data set by eliminating duplicate or less relevant attributes. The benefit of dimensionality reduction for network intrusion detection is to help reduce both computational costs and overfitting. Most of the literature uses PCA, correlation-based filters and Embedded feature selections as methods for reducing features prior to training the classifier [11].

- **Scaling and normalization:** Normalisation and Scaling are operations that adjust numerical feature representations to a single scale of measurement, which may be necessary for most models, including Support Vector Machines and classifiers based on distance. In addition, many of the preprocessing workflows reported in the literature include normalisation steps (for example, among the several normalisation methods available, min-max scaling or z-score) to reduce the impact of the feature with the largest numerical range on a model's learning process [44].

- **Dealing with Imbalance:** CIC-IDS2017 experiences a notable class imbalance in which benign traffic greatly outnumbers its attack classes. As a result, class imbalance is one of the most significant challenges faced when building machine learning models to classify network traffic. To address this issue, the majority of studies on CIC-IDS2017 report that various oversampling, undersampling, and hybrid balancing techniques were used to prevent biased models towards the most frequent classes of benign network traffic. Use of advanced sampling techniques such as SMOTE and hybrid clustering based sampling has been demonstrated to result in improved performance of classifiers due to adding additional instances of minority classified classes while also decreasing the amount of skew in training data [20].

- **Data type optimization:** Numerous studies have examined improving preprocessor efficiency, including resource optimization. For example, a preprocessor can reduce the amount of numeric precision used (i.e. from 64-bit float to 32-bit float) and therefore also reduce the amount of memory consumed during the model training process. With no real impact on classifier performance, these techniques have been included in the vast majority of preprocessor workflows for large-scale IDS deployments to enhance the scalability of these models [11].

Raw CIC-IDS2017 CSV logs (about 2.83 million entries by 79 columns) are ingested and sanitized (eliminating invalid or duplicate flows), encoded into numerical values, and standardized or normalized. Then, the data is divided into training and test sets, frequently in a way that groups similar data points together. To lower the number of dimensions, feature selection (such eliminating associated features, Information Gain ranking, or permutation importance) is used [12], [13]. Finally, cross-validation or holdout splits are used to train the models (DT, RF, SVM, etc.), and the results are checked. This procedure (based on Abdelaziz et al.) highlights the importance of cleaning and preparing data in a methodical way before classifying it [12].

4.3 Feature Selection Strategies

Researchers use various feature selection techniques to improve efficiency and accuracy:

- **Filter methods (information-based):** Independent of the classifier, filter management algorithms evaluate feature variables by utilizing statistical techniques (e.g. mutual information, Information Gain) to assess the quality of individual features. Recently, an approach utilizing RFE and Mutual Information identified the optimal subset of features to improve IDS performance. Results indicate the combination of Filter Rank and Wrapper Eliminate can produce a smaller set of more informative features whilst significantly decreasing both the number of features (dimensionality) and costs associated with computation across CIC-IDS2017 and additional datasets of a similar nature [45].

- **Embedded significance (permutation):** Embedded methods incorporate features selection during the model training phase using classifier-generated feature importance statistics. For example, in a recent study, it was discovered that Random Forest generated feature importance could be used, along with permutation-based evaluation, to determine which attributes were most relevant for intrusion detection. These procedures also evaluate how much each feature contributes to the prediction as the model is developed, thus discarding those features that provide little contribution to prediction [46].

- **Wrapper methods:** Exhaustive and repeated searches have been employed when computationally viable. Waghmode et al. utilize an Exhaustive Feature Selection technique that assesses all feature combinations (via nested classifiers) and selects the subset that produces optimal accuracy[13]. This method consistently identifies the optimal subset, albeit at a combinatorial cost. The top five subsets are input into classifiers, resulting in an accuracy of 99.58% on CIC-IDS-2017 [13].

- **Correlation-based filtering:** One easy way to do this is to get rid of features that are very similar or redundant before modelling. For example, removing one of any pair of features with a correlation over a certain level made the dimension smaller without lowering the accuracy [12].

In short, feature selection in different research might be anything from light filtering to intensive wrapper searches. IG-based and permutation-importance approaches are the most frequent because they can cut down on features (to a few dozen) without hurting detection performance.

4.4 ML Model Configurations and Training Protocols

- **Decision Tree (DT):** Most implementations of DecisionTreeClassifiers (e.g., scikit-learn) typically use either Gini impurity or entropy to make split decisions. In most cases, researchers do not refine the specification of DTs very much but instead evaluate things such as the depth of trees or the use of ensemble methods. In that work, the models were trained and evaluated using stratified data partitions to ensure balanced class representation during training and testing [47].

- **Random Forest (RF):** People usually utilize standard RF settings, like 100 trees, but they change the class weights. Abdelaziz et al. conduct a randomized search over RF hyperparameters and include class-weight adjustment to address imbalance. They also use different types of Weighted Random Forest that punish mistakes in classifying minority classes. Cross-validation (for example, 5-fold stratified) is used to find the best hyperparameters and to guess how well the model will work [12].

- **Support Vector Machine (SVM):** SVMs usually need scaling and a choice of kernel when they are deployed. Waghmode et al. LS-SVM (least-squares SVM) uses a Gaussian (RBF) kernel, and its parameters are adjusted together with the feature subset[13]. Xu et al. employ One-Class SVM (for anomaly detection) with the default v-SVM settings to model just "normal" data. All SVM types need to have their inputs carefully normalized (see preprocessing) [19].

- **Training protocols**: Most authors split their datasets into training and testing portions with stratified sampling to preserve the distribution of both attack classes and non-attack classes. There are common ratios that are used for split portions of data. Typically, these are 70/30 and 80/20. In addition, many authors use K-Fold Cross Validation (most frequently 5-Folds or 10-Folds) to estimate the stability of their models and the capability of generalising beyond the training set. Stratified K-fold Cross Validation helps to ensure that, for every fold, each fold maintains a proportionally representative distribution of attack and non-attack classes, thereby providing an indication of the actual performance of the classifier [48].

- **Hyperparameter settings:** The specifics differ. Abdelaziz et al. discuss the adjustment of RF tree count and depth by randomized search [12]. In their thorough search, Waghmode et al. adjusted the LS-SVM regularization and kernel parameters. These configurations are not always fully reported, but programs like scikit-learn simplify the process of grid and random search [13]**.**

4.5 Evaluation Metrics

Studies consistently report standard classification metrics:

i. **Accuracy:** The accuracy measure reflects the Overall Ratio of Correctly Classified Instances (TP + TN) / Total Instances. It is a common metric reported in IDS (Intrusion Detection System) research due to its ease of calculation. Since benign network traffic typically accounts for more than 90% of any dataset, it is important to look at accuracy along with other metrics that provide a better perspective when looking at class imbalances between attack and benign classes. Recent survey research shows that many researchers are still using the accuracy as a comparison metric for their Intrusion Detection Models that have been tested against Public Datasets such as CIC-IDS2017 and UNSW-NB15[49].

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

ii. **Precision and Recall (Sensitivity):** Precision is defined as TP/(TP+FP), while Recall is defined as TP/(TP+FN). These are frequently documented by class (attack versus benign) and averaged using either macro or weighted methods. Xu et al. delineate accuracy, precision, recall, and F1 for each model about known attacks. Precision and recall illustrate trade-offs. Xu et al. discovered that their CNN exhibited good precision (about 0.93) but marginally inferior recall (around 0.90)[19].

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

$$Recall = \frac{True\ Positive}{True\ Positive\ +\ False\ Positive}$$

iii. **F1-Score:** The harmonic mean of precision and recall metrics. Numerous studies underscore the importance of the F1 score to equilibrate false positives and false negatives. For example, Abdelaziz et al. optimized the weighted F1 score and reported a value of 99.8% for the weighted F1 score[12].

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

iv. **The False Positive Rate (FPR):** Known as the False Alarm Rate, is often reported, particularly in deep learning research; nevertheless, the majority of machine learning studies emphasize precision and recall instead[35].

$$FPR = \frac{FP}{FP + TN}$$

v. **Confusion matrix analysis:** Many IDS studies present confusion matrices or class-wise performance tables to illustrate where classifiers succeed or struggle, particularly when multiple attack categories are evaluated. This level of detail complements overall summary metrics and aids in interpreting model strengths and weaknesses[49].

4.6 Tools and Technology

The reviewed studies predominantly use Python and common ML libraries:

- **Python & scikit-learn:** An open-access research paper recently published on evaluating various supervised learning algorithms for intrusion detection has evaluated them by using Python and scikit-learn to preprocess the data, select features, train models and evaluate performance. The authors of this research paper noted that many of the built-in utilities in scikit-learn help the user to create a consistent way to compare all classifiers applied to datasets like CIC-IDS2017[7].
- **Deep learning libraries:** While the emphasis is on classical machine learning, several studies employ deep networks constructed with Keras/TensorFlow, primarily for comparative purposes[15], [19].
- **Feature selection tools:** Some studies mention using specialized toolkits (e.g. exhaustive search libraries) or algorithms, but many implement FS via scikit-learn or custom code[46].

Table 5: Summary of methodological differences.

Study (Year)	Preprocessing	Feature Selection	Models / Algorithms	Key Results
Xu et al. (2025) [19]	Implied preprocessing (cleaning, scaling) before modelling; evaluated on known vs unseen attacks.	XGBoost feature importance discussed for feature representation.	MLP, CNN, OCSVM, LOF	Known attacks: MLP F1 ≈ 0.94; OCSVM balanced precision/recall.
Talukder et al. (2024) [11]	Addressed class imbalance via oversampling (RO); used PCA for dimensionality reduction.	PCA and clustering-based embedding as feature reduction.	DT, RF, ET & others	On CIC-IDS2017: DT & RF ~99.99% accuracy; ensemble models ~99.94%.
Saidane et al. (2024) [20]	SMOTE + PCA + advanced data processing before modelling.	PCA + FCBF (fast correlation-based filter).	Extra Trees, VGG19 ensemble	VGG19 Ensemble ≈99.26% on CIC-IDS2017.
Rodríguez (2022) [21]	Cleaning, scaling, train/test splits; explained classical ML preprocessing workflows.	Correlation-based filtering discussed.	RF, SVM, DT	Provides comparative performance context for multiple classifiers.

Based on the Table 5, it summarises recent studies on CIC-IDS2017, showing that high detection performance is consistently achieved through careful preprocessing, feature selection, and robust evaluation protocols. Across the literature, classical models such as Random Forest remain highly competitive, while deep and anomaly-based models provide complementary strengths, particularly for unseen attacks [11], [19], [20], [21].

5. RESULTS & DISCUSSIONS

5.1 Overview Review Studies

Several recent studies have assessed Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM) classifiers using the CIC-IDS2017 dataset. For instance, Maseer et al. (2021) conducted an extensive evaluation of 10 machine learning methods (including DT, RF, and SVM) on the CIC-IDS2017 dataset [6]. The tuned DT classifier had an accuracy of around 99.49% (with the same precision, recall, and F1 score), the RF classifier had an accuracy of about 99.54% (with an F1 score of about 99.55%), and the SVM classifier had an accuracy of about 96.72% (with an F1 score of about 97.89%) [6]. Talukder et al. (2024) employed advanced preprocessing methods (oversampling, feature embedding) and demonstrated that the DT, RF, and Extra-Tree models attained about 99.99% accuracy on CIC-IDS2017 [11]. Bacevičius and Paulauskaitė-Ţarasevičienė (2023) examined raw, significantly imbalanced flows (including CIC-IDS2017) and discovered that a CART decision tree frequently surpassed RF, achieving average F1-scores between 96.9% and 99.9% across 15 classes [16]. Other studies support these trends that RF frequently achieves detection rates approaching near 100% (approximately 99.96% accuracy recorded in one research) outperforming DT and SVM in raw accuracy [12]. The studies that have been evaluated thoroughly reveal that DT and RF, especially RF, have very high detection rates on CIC-IDS2017. On the other hand, SVM usually does worse than the tree-based models. In experiments that utilized different datasets, the comparative performance of DT, RF, and SVM remained relatively uniform. Talukder et al. assessed their models using UNSW-NB15 and CIC-IDS2017/2018, discovering that RF achieved an accuracy of 99.59–99.95% on UNSW-NB15 and around 99.99% on CIC-IDS2017 (DT, RF, Extra-Trees), with a slight drop to 99.94% (DT and RF) on CIC-IDS2018. This indicates that the exceptional performance of DT/RF is not only coincidental, it is consistent across all current NIDS benchmarks [11]. Further research (e.g., Pelletier & Abualkibash 2020) indicates that older datasets (NSL-KDD) are more manageable, whereas CIC-IDS2017 provides a more authentic representation of multi-class traffic [50]. Generally, tree-based models maintain near-perfect accuracy on these contemporary datasets. Conversely, SVM's reduced sensitivity to CIC-IDS2017 is seen in analogous intrusion sets. Although our primary emphasis is on CIC-IDS2017, other comparisons indicate that the accuracy of RF/DT remains superior on analogous intrusion datasets [11].

5.2 Analysis on CIC-IDS2017 Dataset

Figure 2 presents the most important performance metrics (accuracy, precision, recall, and F1-score) for the top-performing Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM) models on CIC-IDS2017, as obtained from the reviewed literature. The optimal Decision Tree and Random Forest models achieve classification rates of 99–100%, although Support Vector Machine is significantly slower. For example, Maseer et al. say that a tuned DT has 99.49% accuracy (precision = recall = F1 = 99.49%) while an RF has 99.54% accuracy (precision 99.56%, recall 99.54%, F1 99.55%) [6]. Their best SVM model, on the other hand, only gets 96.72% accuracy (precision 99.27%, recall 96.72%, F1 97.89%). The statistics obtained through thorough cross-validation indicate that Random Forest (RF) and Decision Tree (DT) have nearly eliminated the disparity between benign and malicious activities in CIC-IDS2017, however Support Vector Machine (SVM) fails to detect a significant proportion of attacks. Figure 2 illustrates these tendencies, which are consistent across studies.

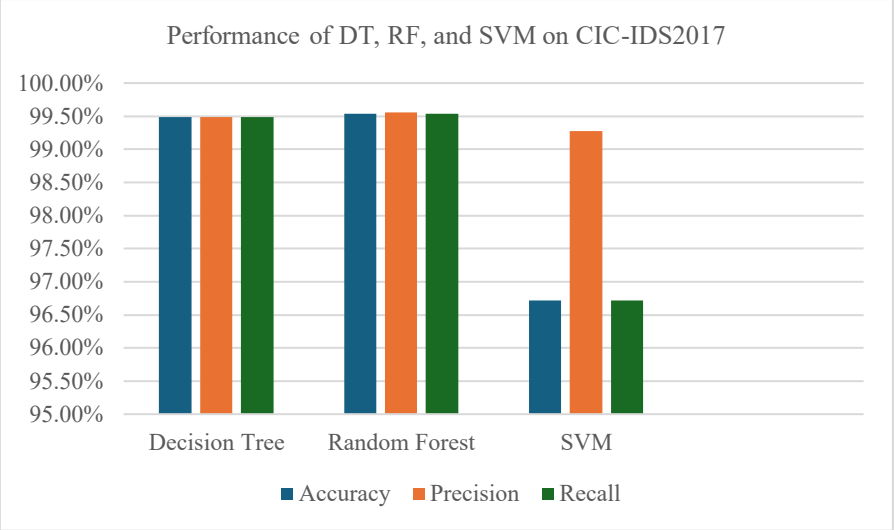

Figure 2: Performance of DT, RF, and SVM on CIC-IDS2017.

These values from Figure 2 demonstrate the efficacy of each strategy. Decision Trees (DT) and Random Forests (RF) show almost perfect detection rates (accuracy between 99.5 and 99.6%) with minimal missed attacks compared to the accuracy and recall for Support Vector Machines (SVM), which decline by multiple percentage points. The recall rate of SVM declines because its threshold is more conservative, but its precision rate is high due to the low number of false positives. This trend can be seen in the published literature as well. RF/DT achieves accuracy/F1 scores above 99% on CIC-IDS2017 while SVM often averages in the mid-90% [6].

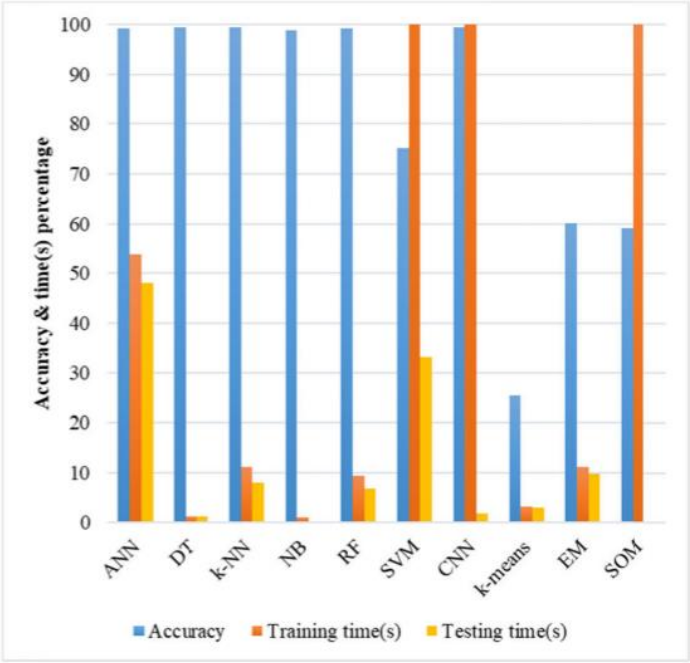

Figure 3: Overall accuracy and runtime performance [6].

Figure 3 compares several machine learning algorithms based on performance metrics, including accuracy, time taken for training, and time taken for testing. Several tree-based algorithms (for example, Decision Tree, k-NN (k-Nearest Neighbours), Naïve Bayes, and Random Forest), achieved very close to complete accuracy, while training and testing took less time than other algorithms in this comparison.

SVM (Support Vector Machines) and CNN (Convolutional Neural Networks) required much longer to train but achieved a higher degree of accuracy than the tree-based algorithms. Therefore, they carried a greater computational load. Unsupervised Learning (for example k-means and EM) did not perform as well as the tree-based algorithms, with no Better than hit-and-miss results on accuracy. Overall, the findings demonstrate that traditional supervised learning, particularly tree-based methods are the best choices for balancing performance for detection with runtimes.

5.3 Comparative and Strength

The experiments evaluated demonstrated a number of trade-offs among Decision Trees (DT), Random Forests (RF) and Support Vector Machines (SVM) with respect to interpretability, scalability, and generalisation. Decision trees are uniquely interpretable since they produce clear conditional rules. Whereas, a Random Forest (RF), which is a collection of several trees, is much more complicated, and SVM's decision boundary is similarly difficult to interpret. However, as the authors observe "it is important to note that RF can be computationally intensive, perform poor on high-dimensional data, and be less interpretable than simpler models," even when RF generally has better accuracy. Therefore, decision trees are a unique advantage for interpretability, while random forests may lose interpretability for only small improvements in accuracy. SVM similarly provides no feature importance or straightforward guidelines, rendering it the least interpretable of the three methods.

Table 6: Summary of performance metrics from reviewed studies.

Study	Dataset	Model	Accuracy	Precision	Recall	F1-score	Notes
Maseer et al. (2021) [6]	CIC-IDS2017	DT	99.49%	99.49%	99.49%	99.49%	Fast training
Maseer et al. (2021) [6]	CIC-IDS2017	RF	99.54%	99.56%	99.54%	99.55%	Strong but slower
Maseer et al. (2021) [6]	CIC-IDS2017	SVM	96.72%	99.27%	96.72%	97.89%	Long training time
Talukder et al. (2024) [11]	CIC-IDS2017	DT/RF	~99.99%	~99.99%	~99.99%	~99.99%	With oversampling
Bacevičius & Paulauskaitė-Ţarasevičienė (2023) [16]	CIC-IDS2017	CART (DT)	96.9 - 99.9% F1	-	-	-	Outperforms RF
Pandit & Ghimire (2023) [51]	CIC-IDS2017	SVM	99.0%	-	-	-	Baseline SVM (no autoencoder) achieved 99% accuracy.

There exists a significant difference in training duration. The optimal decision tree in Maseer et al. [6] was trained in approximately 1.23 seconds, but the corresponding random forest required roughly 9.38 seconds on the same hardware. The SVM exhibited significantly slower performance, requiring around 343.6 seconds for training and 33 seconds for testing. SVM is ineffective with CIC-IDS2017's about 3 million flow records, although single-tree decision trees perform optimally (almost linear time), and random forests exhibit satisfactory performance (linear time multiplied by the number of trees). In summary, Decision Trees (DT) are fast in training and testing, Random Forests (RF) are comparatively slower however still beneficial, while Support Vector Machines (SVM) exhibit significant latency, rendering them less advantageous for extensive datasets without optimisation.

Table 6 shows a comparison of how well different ML models performed when it came to classifying the CIC-IDS2017 dataset. Maseer et al. (2021) reported that the Decision Tree (DT) model achieved a precision, accuracy, recall, and F1-score of 99.49%, indicating its efficiency and comprehensibility during training [6]. Random Forest (RF) outperformed Decision Tree (DT) slightly, achieving an accuracy of 99.54% and an F1-score of 99.55%. Nonetheless, the training duration was extended. The Support Vector Machine (SVM) score a high precision rate of 99.27%, although its overall accuracy was 96.72% and its F1-score was 97.89%, both of which were lower. This was primarily due to the extended training duration and heightened sensitivity to extensive datasets.

Alternative studies have shown better benefits in some contexts. For instance, Talukder et al. (2024) used oversampling techniques on CIC-IDS2017 with near-perfect results, 99.99% accuracy, precision, recall, and F1-score [11]. Similarly, Bacevičius & Paulauskaitė-Ţarasevičienė (2023) showed that the CART version of Decision Trees achieved an F1-score range of 96.9-99.9%, exceeding Random Forest in some cases [16]. These results are promising and suggest that, in this case, classical models, such as Decision Trees (DT) and Random Forest (RF), remain highly competitive. In particular, these results would improve significantly with either data balance or possible iterative algorithm improvements. However, SVM has recurring scaling issues while working with large imbalanced intrusion detection datasets.

All three models perform well on the CIC-IDS 2017 dataset. Still, it is important to test how well they can generalise. Both Random Forest and Decision Trees preserve the characteristics of this dataset, which raises concerns about overfitting. In some cases, a study shows that good results on CIC-IDS2017 "cannot be achieved under real-world constraints". Feature selection and hyper-parameter tuning might make the model fit CIC better, but these changes could expose hidden weaknesses when new inputs are used. To put it another way, RF/DT works well with the current data, but it might not work as well in the future when conditions change. SVM may make bias worse even though it is a very advanced model because it often doesn't represent minority groups well. Some people have said that this is why their research on Yahoo and DDoS data sets had good results, while the performance of active classifiers was more difficult to compare. In summary, RF/DT works well on the CIC-IDS2017 attack portfolio, but it's not clear how well it will work in other situations.

5.4 Synthesis and Implication

The results address our research questions directly. For RQ1 (best performing classifier), Decision Trees (DT) and Random Forest (RF) greatly outperform Support Vector Machine (SVM) on CIC-IDS2017 with nearly perfect accuracy and recall (~99.5–99.9%), whereas SVM performs approximately at best 99.5 and worst 97% [6]. This suggests that tree based and ensemble approaches are the most reliable classifiers on this dataset.

With respect to RQ2 (trade-off between models), for DT and RF, the accuracy is nearly the same, even though it is differentiating in practices. Decision Trees (DT) result in fast training and better interpretability while Random Forest is more robust to noise through bagging but comes with more

computational cost [11]. The SVM, on the other hand, has lower training time and a less memory requirement which makes it less suited for large-scale traffic classification and to some extent may not even make it worth using (there would be no appreciable gain).

For RQ3 (would models perform the same in any dataset err being tested), throughout previous research, a caution (warning) is that such high performance might not be transferrable to other datasets or realistic traffic [16]. This emphasizes the need to assess models as applicable on different datasets or attack approaches to generalize their robustness. The results reveal that DT or RF would be the most suitable for implementation in intrusion detection systems, which are highly agreeable levels of accuracy and word being classified balance. While it performed well on the CIC-IDS2017 dataset, to be sure it does have application in other datasets (CIC-IDS2018 or even in realistic traffic) would need to be evaluated to confirm outside of additional research.

5.5 Key Findings

RQ1 - Best Performing Classifier

- In several tests utilizing CIC-IDS2017, Decision Tree (DT) and Random Forest (RF) consistently demonstrate superior accuracy and recall compared to Support Vector Machine (SVM). The results suggest that DT and RF have about 99.5 - 99.9% accuracy and F1-scores, whereas SVM has about 96 - 97% accuracy. This shows that tree-based classifiers work well with this dataset [6], [11].

RQ2 - Trade-offs and Practical Considerations

- DT and RF have equal detection rates, but DT is easier to understand, faster to train, and simpler to use, which makes it better for situations when explainability is very important. RF, on the other hand, is more resistant to noise and variance because it is made up of many different models, but this makes it more expensive to run. SVM is quite accurate on structured data, but it takes longer to train and has a lower recall rate, which makes it less useful for large-scale IDS deployment [16].

RQ3 - Generalization and Applicability

- Many studies have shown that achieving nearly perfect performance on CIC-IDS2017 can be inconclusive when considering other datasets or real network traffic. Specifically, models built on CIC-IDS2017 often demonstrate some degree of overfitting to its distribution caused by the way it had so much data and a number of weeks of different attack data. Models trained on CIC-IDS2017 can often result in decreased performance when tested on new datasets like CIC-IDS2018 or real-time traffic. This emphasizes the importance of testing models on multiple datasets and considering some adaptive techniques for responding to ongoing attack patterns [52].

6. CONCLUSIONS

This research was an examination and synthesis of earlier studies that used machine learning methods (Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM)) to identify intrusions in the CIC-IDS2017 dataset. The study seeks to comparatively assess the classification performance of the selected machine learning methods, delineate the strengths and weaknesses of each method, and consider their subsequent applications in intrusion detection systems in practice. The review found that DT and RF outperformed SVM every time, in terms of accuracy, recall, and F1-score. Performance measures attained from DT and RF were sometimes very close to the ideal metric range (around 99.5–

99.9%). DT was employed because it is simple to train and understand quickly, whereas RF was superior in noise and variability when implemented with an ensemble structure. SVM, on the other hand, was generally weaker in terms of recall and scalability, since it is more costly to deploy. Nevertheless, the results imply important research gaps and challenges. Most importantly, generalizability is weak, and while it is assumed that performance is good for CIC-IDS2017, it is not clear how it will perform for future datasets (for example, CIC-IDS2018) or for real-time data. In addition, several studies have not yet explored limitations related to computational scalability, model drift, and class imbalance. These limitations emphasize the need for cross-dataset validation, adaptive learning techniques, and efficiency-focused strategies for effective deployment of Intrusion Detection Systems (IDS).

Ultimately, the evidence strongly supports the use of tree-based models (DT and RF) for Intrusion Detection, in controlled benchmark settings. More importantly, the need for further evaluation beyond on CIC-IDS2017 is crucial to ensure robustness in a dynamic cybersecurity environment. Thus, future research should focus on validating these models in diverse real-world circumstances, and on developing hybrid or ensemble methods that bring together accuracy, interpretability, and scalability.

The use of advanced and adaptive approaches to intrusion detection, beyond traditional machine learning models, has increased significantly in recent years. The use of advanced deep learning techniques such as Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and hybrid models composed of CNN and LSTM has become increasingly popular due to their capability to extract complex patterns from traffic automatically, thereby improving detection rates. Likewise, hybrid models that utilize deep learning techniques in conjunction with classical algorithms such as Random Forests and Support Vector Machine (SVM) have been recognized for their ability to achieve a balance among performance, interpretability, and computational efficiency. There has also been a strong focus on new techniques such as Graph Neural Networks (GNN), which enhance the capability for detecting intrusions by modelling network traffic as structured relationships that can help to effectively identify complex and multi-stage attacks. Other areas of research have focused on class imbalance issues, improving the real-time detection capabilities of intrusion detection systems, and developing lightweight models for use in resource-constrained and high-throughput environments. Collectively, these trends indicate the future research direction of intrusion detection systems will be to develop scalable, adaptive, and explainable intrusion detection systems that will be able to successfully detect intrusions in dynamic real-world network environments.

ACKNOWLEDGEMENT

The authors would like to thank the Faculty of Computing and Informatics and Universiti Malaysia Sabah (UMS) for providing the resources to complete this research.

REFERENCES

[1] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, Jan. 2021, doi: 10.1002/ETT.4150.

[2] O. H. Abdulganiyu, T. Ait Tchakoucht, and Y. K. Saheed, “A systematic literature review for network intrusion detection system (IDS),” *International Journal of Information Security 2023 22:5*, vol. 22, no. 5, pp. 1125–1162, Mar. 2023, doi: 10.1007/S10207-023-00682-2.

[3] T. Saranya, S. Sridevi, C. Deisy, T. D. Chung, and M. K. A. A. Khan, "Performance Analysis of Machine Learning Algorithms in Intrusion Detection System: A Review," *Procedia Comput. Sci.*, vol. 171, pp. 1251–1260, Jan. 2020, doi: 10.1016/J.PROCS.2020.04.133.

[4] A. Azab, M. Khasawneh, S. Alrabaee, K. K. R. Choo, and M. Sarsour, "Network traffic classification: Techniques, datasets, and challenges," *Digital Communications and Networks*, vol. 10, no. 3, pp. 676–692, Jun. 2024, doi: 10.1016/J.DCAN.2022.09.009.

[5] L. F. Sikos, "Packet analysis for network forensics: A comprehensive survey," *Forensic Science International: Digital Investigation*, vol. 32, p. 200892, Mar. 2020, doi: 10.1016/J.FSIDI.2019.200892.

[6] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset," *IEEE Access*, vol. 9, pp. 22351–22370, 2021, doi: 10.1109/ACCESS.2021.3056614.

[7] Y. Zhang, H. Zhang, and B. Zhang, "An Effective Ensemble Automatic Feature Selection Method for Network Intrusion Detection," *Information 2022, Vol. 13, Page 314*, vol. 13, no. 7, p. 314, Jun. 2022, doi: 10.3390/INFO13070314.

[8] L. Diana, P. Dini, and D. Paolini, "Overview on Intrusion Detection Systems for Computers Networking Security," *Computers 2025, Vol. 14, Page 87*, vol. 14, no. 3, p. 87, Mar. 2025, doi: 10.3390/COMPUTERS14030087.

[9] K. Kurniabudi, D. Stiawan, M. Yazid bin Idris, B. Kerim, and R. Budiarto, "Important Features of CICIDS-2017 Dataset For Anomaly Detection in High Dimension and Imbalanced Class Dataset," *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, vol. 9, no. 2, pp. 498–511, 2021, doi: 10.52549/ijeei.v9i2.3028.

[10] M. Catillo, A. Del Vecchio, A. Pecchia, and U. Villano, "A Case Study with CICIDS2017 on the Robustness of Machine Learning against Adversarial Attacks in Intrusion Detection," *ACM International Conference Proceeding Series*, Aug. 2023, doi: 10.1145/3600160.3605031/ASSETS/HTML/IMAGES/ARES2023-83-FIG3.JPG.

[11] M. A. Talukder *et al.*, "Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction," *J. Big Data*, vol. 11, no. 1, pp. 1–44, Dec. 2024, doi: 10.1186/S40537-024-00886-W/TABLES/16.

[12] M. T. Abdelaziz *et al.*, "Enhancing Network Threat Detection with Random Forest-Based NIDS and Permutation Feature Importance," *Journal of Network and Systems Management*, vol. 33, no. 1, pp. 1–36, Mar. 2025, doi: 10.1007/S10922-024-09874-0/FIGURES/3.

[13] P. Waghmode, M. Kanumuri, H. El-Ocla, and T. Boyle, "Intrusion detection system based on machine learning using least square support vector machine," *Sci. Rep.*, vol. 15, no. 1, p. 12066, Dec. 2025, doi: 10.1038/S41598-025-95621-7.

[14] E. C. P. Neto, S. Iqbal, S. Buffett, M. Sultana, and A. Taylor, "Deep learning for intrusion detection in emerging technologies: a comprehensive survey and new perspectives," *Artificial Intelligence Review 2025 58:11*, vol. 58, no. 11, pp. 340-, Aug. 2025, doi: 10.1007/S10462-025-11346-Z.

[15] M. Catillo, A. Del Vecchio, A. Pecchia, and U. Villano, "Transferability of Machine Learning Models Learned from Public Intrusion Detection Datasets: the CICIDS2017 Case Study," Mar. 2022, doi: 10.1007/s11219-022-09587-0.

[16] M. Bacevicius and A. Paulauskaite-Taraseviciene, "Machine Learning Algorithms for Raw and Unbalanced Intrusion Detection Data in a Multi-Class Classification Problem," *Applied Sciences 2023, Vol. 13, Page 7328*, vol. 13, no. 12, p. 7328, Jun. 2023, doi: 10.3390/APP13127328.

[17] K. Kurniabudi, D. Stiawan, Darmawijoyo, M. Y. Bin Idris, B. Kerim, and R. Budiarto, "Important features of CICIDS-2017 dataset for anomaly detection in high dimension and imbalanced class dataset," *Indonesian Journal of Electrical Engineering and Informatics*, vol. 9, no. 2, pp. 498–511, Jun. 2021, doi: 10.52549/IJEEI.V9I2.3028.

[18] R. Patgiri, U. Varshney, T. Akutota, and R. Kunde, "An Investigation on Intrusion Detection System Using Machine Learning," *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018*, pp. 1684–1691, Jul. 2018, doi: 10.1109/SSCI.2018.8628676.

[19] Z. Xu and Y. Liu, "Robust Anomaly Detection in Network Traffic: Evaluating Machine Learning Models on CICIDS2017," *2025 10th International Conference on Electronic Technology and Information Science, ICETIS 2025*, pp. 475–482, Jun. 2025, doi: 10.1109/ICETIS66286.2025.11144025.

[20] S. Saidane, F. Telch, K. Shahin, and F. Granelli, "Optimizing Intrusion Detection System Performance Through Synergistic Hyperparameter Tuning and Advanced Data Processing," pp. 141–160, Aug. 2024, doi: 10.5121/csit.2024.141411.

[21] M. Rodríguez, Á. Alesanco, L. Mehavilla, and J. García, "Evaluation of Machine Learning Techniques for Traffic Flow-Based Intrusion Detection," *Sensors (Basel)*, vol. 22, no. 23, p. 9326, Dec. 2022, doi: 10.3390/S22239326.

[22] Y. Zhang, R. C. Muniyandi, and F. Qamar, "A Review of Deep Learning Applications in Intrusion Detection Systems: Overcoming Challenges in Spatiotemporal Feature Extraction and Data Imbalance," *Applied Sciences 2025, Vol. 15, Page 1552*, vol. 15, no. 3, p. 1552, Feb. 2025, doi: 10.3390/APP15031552.

[23] A. M. Alashjaee, "Deep learning for network security: an Attention-CNN-LSTM model for accurate intrusion detection," *Scientific Reports 2025 15:1*, vol. 15, no. 1, pp. 21856-, Jul. 2025, doi: 10.1038/s41598-025-07706-y.

[24] M. Zhong, M. Lin, C. Zhang, and Z. Xu, "A survey on graph neural networks for intrusion detection systems: Methods, trends and challenges," *Comput. Secur.*, vol. 141, p. 103821, Jun. 2024, doi: 10.1016/J.COSE.2024.103821.

[25] Z. Sun, A. M. H. Teixeira, and S. Toor, "GNN-IDS: Graph Neural Network based Intrusion Detection System," *ACM International Conference Proceeding Series*, Jul. 2024, doi: 10.1145/3664476.3664515;SUBPAGE:STRING:ABSTRACT;CSUBTYPE:STRING:CONFERENCE.

[26] Y. Wang, Z. Han, Y. Du, J. Li, and X. He, "BS-GAT: a network intrusion detection system based on graph neural network for edge computing," *Cybersecurity 2025 8:1*, vol. 8, no. 1, pp. 27-, Apr. 2025, doi: 10.1186/S42400-024-00296-8.

[27] C. Xu, D. Li, Z. Liu, J. Yang, Q. Shen, and N. Tong, "Few-shot network intrusion detection method based on multi-domain fusion and cross-attention," *PLoS One*, vol. 20, no. 7, p. e0327161, Jul. 2025, doi: 10.1371/JOURNAL.PONE.0327161.

[28] S. Layeghy and M. Portmann, "On Generalisability of Machine Learning-based Network Intrusion Detection Systems," *Computers and Electrical Engineering*, vol. 108, May 2022, doi: 10.1016/j.compeleceng.2023.108692.

[29] J. Hesford *et al.*, "Expectations Versus Reality: Evaluating Intrusion Detection Systems in Practice," Mar. 2024, Accessed: Aug. 30, 2025. [Online]. Available: https://arxiv.org/pdf/2403.17458v2

[30] E. E. Abdallah, W. Eleisah, and A. F. Otoom, "Intrusion Detection Systems using Supervised Machine Learning Techniques: A survey," *Procedia Comput. Sci.*, vol. 201, no. C, pp. 205–212, Jan. 2022, doi: 10.1016/J.PROCS.2022.03.029.

[31] M. M. Rahman, S. Al Shakil, and M. R. Mustakim, "A survey on intrusion detection system in IoT networks," *Cyber Security and Applications*, vol. 3, p. 100082, Dec. 2025, doi: 10.1016/J.CSA.2024.100082.

[32] V. Shanmugam, R. Razavi-Far, and E. Hallaji, "Addressing Class Imbalance in Intrusion Detection: A Comprehensive Evaluation of Machine Learning Approaches," *Electronics (Switzerland)*, vol. 14, no. 1, Jan. 2025, doi: 10.3390/ELECTRONICS14010069.

[33] Y. Long, "Enhanced SVM-based model for predicting cyberspace vulnerabilities: Analyzing the role of user group dynamics and capital influx," *PLoS One*, vol. 20, no. 7, p. e0327476, Jul. 2025, doi: 10.1371/JOURNAL.PONE.0327476.

[34] Z. Halim *et al.*, "An effective genetic algorithm-based feature selection method for intrusion detection systems," *Comput. Secur.*, vol. 110, Nov. 2021, doi: 10.1016/J.COSE.2021.102448.

[35] O. Achbarou, T. Datsi, O. Bourkoukou, and A. M. El kiram, "Enhanced intrusion detection system using feature selection and hybrid learning models for high performance and efficiency in an IOT environment," *Journal of Engineering Research*, Oct. 2025, doi: 10.1016/J.JER.2025.10.016.

[36] S. T. Hamidou and A. Mehdi, "Enhancing IDS performance through a comparative analysis of Random Forest, XGBoost, and Deep Neural Networks," *Machine Learning with Applications*, vol. 22, p. 100738, Dec. 2025, doi: 10.1016/J.MLWA.2025.100738.

[37] W. H. Elashmawi, A. Sheta, and A. Al-Qerem, "Intelligent Intrusion Detection System Using RF, SVM, and DT: A Comparison-Based KDD Data Set," *Journal of Computer Science*, vol. 21, no. 8, pp. 1749–1759, Aug. 2025, doi: 10.3844/JCSSP.2025.1749.1759.

[38] A.-G. Mari, D. Zinca, and V. Dobrota, "Development of a Machine-Learning Intrusion Detection System and Testing of Its Performance Using a Generative Adversarial Network," *Sensors 2023, Vol. 23, Page 1315*, vol. 23, no. 3, p. 1315, Jan. 2023, doi: 10.3390/S23031315.

[39] N. Singh Yadav, V. Prakash Sharma, D. Sikha Datta Reddy, and S. Mishra, "An Effective Network Intrusion Detection System Using Recursive Feature Elimination Technique," *Engineering Proceedings 2023, Vol. 59, Page 99*, vol. 59, no. 1, p. 99, Dec. 2023, doi: 10.3390/ENGPROC2023059099.

[40] F. Genuario, G. Santoro, M. Giliberti, S. Bello, E. Zazzera, and D. Impedovo, "Machine Learning-Based Methodologies for Cyber-Attacks and Network Traffic Monitoring: A Review and Insights," *Information 2024, Vol. 15, Page 741*, vol. 15, no. 11, p. 741, Nov. 2024, doi: 10.3390/INFO15110741.

[41] D. Pinto, I. Amorim, E. Maia, and I. Praça, "A review on intrusion detection datasets: tools, processes, and features," *Computer Networks*, vol. 262, p. 111177, May 2025, doi: 10.1016/J.COMNET.2025.111177.

[42] S. Farhat, M. Abdelkader, A. Meddeb-Makhlouf, and F. Zarai, "Evaluation of DoS/DDoS Attack Detection with ML Techniques on CIC-IDS2017 Dataset," vol. 1, 2023, doi: 10.5220/0011605700003405.

[43] A. Singh, J. Prakash, G. Kumar, P. K. Jain, and L. S. Ambati, "Intrusion Detection System: A Comparative Study of Machine Learning-Based IDS," *Journal of Database Management*, vol. 35, no. 1, 2024, doi: 10.4018/JDM.338276.

[44] M. Mbow, H. Koide, and K. Sakurai, "Handling class Imbalance problem in Intrusion Detection System based on deep learning," *International Journal of Networking and Computing*, vol. 12, no. 2, pp. 467–492, 2022, doi: 10.15803/IJNC.12.2_467.

[45] S. Wanjau and G. Kamau, "Ensemble Feature Selection for Network Intrusion Detection: Combining Information Gain and Random Forest with Recursive Feature Elimination," *International Journal of Computer and Information Technology(2279-0764)*, vol. 13, no. 4, Jan. 2025, doi: 10.24203/262J5109.

[46] L. ALkahla, M. Khalaf Hussein, and A. Alqassab, "Feature Selection Techniques in Intrusion Detection: A Comprehensive Review," *Iraqi Journal for Computers and Informatics*, vol. 50, no. 1, pp. 46–53, Jun. 2024, doi: 10.25195/IJCI.V50I1.462.

[47] T. H. Chua and I. Salam, "Evaluation of Machine Learning Algorithms in Network-Based Intrusion Detection Using Progressive Dataset," *Symmetry 2023, Vol. 15, Page 1251*, vol. 15, no. 6, p. 1251, Jun. 2023, doi: 10.3390/SYM15061251.

[48] Md. A. Uddin, S. Aryal, M. R. Bouadjenek, M. Al-Hawawreh, and Md. A. Talukder, "A Dual-Tier Adaptive One-Class Classification IDS for Emerging Cyberthreats," Mar. 2024, Accessed: Jan. 05, 2026. [Online]. Available: http://arxiv.org/abs/2403.13010

[49] V. Z. Mohale and I. C. Obagbuwa, "Evaluating machine learning-based intrusion detection systems with explainable AI: enhancing transparency and interpretability," *Front. Comput. Sci.*, vol. 7, p. 1520741, May 2025, doi: 10.3389/FCOMP.2025.1520741/BIBTEX.

[50] Z. Pelletier and M. Abualkibash, "Evaluating the CIC IDS-2017 Dataset Using Machine Learning Methods and Creating Multiple Predictive Models in the Statistical Computing Language R," *International Research Journal of Advanced Engineering and Science*, vol. 5, no. 2, pp. 187–191, 2020.

[51] R. Kumar Pandit and S. Kumar Ghimire, "Proceedings of 13 th IOE Graduate Conference Performance Enhancement of Support Vector Machine Using Autoencoder For Network Intrusion Detection System," vol. 13, Apr. 2023.

[52] J. Halvorsen, C. Izurieta, H. Cai, and A. Gebremedhin, "Applying Generative Machine Learning to Intrusion Detection: A Systematic Mapping Study and Review," *ACM Comput. Surv.*, vol. 56, no. 10, p. 257, Jun. 2024, doi: 10.1145/3659575.