Intelligent Intrusion Detection Systems – A comprehensive overview of applicable AI Methods with a Focus on IoT Security

Olivér Hornyák

Abstract—**The rapid advancement of technology and the increasing complexity of cyber threats have necessitated the development of more sophisticated security measures. This paper presents a structured analysis of how artificial intelligence (AI) methods enhance the accuracy, adaptability, and efficiency of Intrusion Detection Systems (IDS). Different AI approaches, including machine learning, deep learning, and reinforcement learning are categorized and evaluated, highlighting their practical applications and limitations. The main focus is on enhancing the detection capabilities of IDS. By examining supervised, unsupervised, and reinforcement learning approaches, the study highlights how these methods can improve the accuracy, efficiency, and adaptability of IDS in identifying both known and novel threats. Additionally, the paper addresses the challenges associated with AI-based IDS, such as the need for extensive datasets, computational demands, and vulnerability to adversarial attacks. The findings underscore the transformative impact of AI on IDS and suggest directions for future research to further advance the field. With the exponential growth of Internet of Things (IoT) devices, securing networked environments has become increasingly challenging due to their resource constraints, diverse communication protocols, and exposure to cyber threats. Lightweight IDS models may provide solutions for the computational overhead, the scalability and privacy issues. This overview aims to serve as a valuable resource for researchers and practitioners seeking to leverage AI to bolster cybersecurity defenses. This paper not only provides a historical perspective but also critically analyzes current advancements and future research directions with a particular focus on IoT security and lightweight intrusion detection models.**

Index Terms—**Information security. Intrusion detection, Artificial intelligence, Machine learning**

I. Introduction

A. Background

IN cybersecurity, "intrusion" refers to any unauthorized access or attempt to gain access to a computer system, network, or data. This can include exploiting vulnerabilities to access sensitive information, installing malicious software, or disrupting normal operations. Intrusions can range from simple unauthorized logins to sophisticated, multi-stage attacks involving malware, phishing, or advanced persistent threats. The goal of Intrusion Detection Systems (IDS) is to identify and alert on such activities to help protect against data breaches and other security incidents.

The concept of detecting intrusions has been part of computer security efforts since the early days of computer networking. One of the first recorded intrusion detection systems, called "Cops" (Computer Oracle and Password System) was developed in the 1980s [1]. It was a collection of programs to warn the users of potential problems.

The IDS field matured after this period as computer security specialists increasingly understood the need to detect and respond to malicious activities in networks, ultimately leading to the more advanced and intelligent systems used today.

In the past decades, the rapid advancement of information technology has significantly transformed various industries and aspects of human life. Computer networks are necessary in business, industry, and everyday activities, necessitating the development of reliable and secure networks. However, this technological progress has also introduced numerous challenges, particularly in ensuring the availability, integrity, and confidentiality of network resources.

Among the various threats to network security, Denial of Service (DoS) attacks [63] stand out as particularly damaging. DoS attacks aim to disrupt the availability of services to end users by overwhelming network resources and systems with excessive, illegitimate requests. This type of attack first gained widespread attention in 2000 when Yahoo became one of the earliest high-profile victims. Today, web services and social media platforms are frequent targets of such attacks.

In addition to DoS attacks, other forms of cyber threats, such as Remote to Local (R2L) and User to Root (U2R) attacks, pose significant risks. R2L attacks involve an external attacker gaining local access rights to network resources that are typically restricted to local users, often exploiting vulnerabilities in services like file servers. U2R attacks, on the other hand, involve granting privileges from a normal user to a root user, providing full access to system resources to the attacker.

Cyber threats' dynamic and evolving nature makes it challenging for all attack types to use fixed, traditional security measures. Consequently, Intrusion Detection Systems have become an essential part of network security, developed to monitor network traffic and issue alerts upon detecting suspicious activities. IDS can be implemented as host-based systems, monitoring specific devices, or as network-based

This paragraph of the first footnote will contain the date on which you submitted your paper for review. This work was supported in part by University of Miskolc, grant No....

O. Hornyák is with the Institute of Information Science, University of Miskolc, Hungary, (e-mail: oliver.hornyak@uni-miskolc.hu).

DOI: 10.36244/ICJ.2025.5.8

systems, overseeing all network traffic. These systems are further categorized into anomaly-based and misuse-based IDS. Anomaly-based IDS detects attacks by comparing current traffic patterns against established baselines of normal behavior, offering the advantage of identifying novel attacks but often generating higher false positive rates. Misuse-based IDS rely on known attack signatures, effectively identifying known threats but potentially missing new, unknown attack vectors.

The structure of the rest of the paper is as follows: an overview of the historical development of IDS is presented, highlighting key milestones and technological shifts that have shaped modern security approaches. Following this, various AI methodologies applied in IDS, including supervised, unsupervised, and reinforcement learning techniques, are examined, with their strengths, limitations, and practical applications discussed.

A significant portion of the paper is dedicated to addressing the challenges and emerging trends in AI-driven IDS, particularly within the context of IoT security. The increasing demand for lightweight IDS models is analyzed, emphasizing their importance in resource-constrained environments. Critical factors such as computational efficiency, energy consumption, scalability, and real-time performance are evaluated. Recent advancements in edge computing, federated learning, and adversarial defense mechanisms are also explored, demonstrating their role in improving IDS effectiveness in modern cyber-physical systems. The paper concludes with a discussion on future research directions and open challenges in AI-driven IDS.

B. *The goal of the paper*

The primary goal of this paper is to provide a comprehensive and structured analysis of how artificial intelligence (AI) methods enhance the detection capabilities, adaptability, and efficiency of Intrusion Detection Systems (IDS), with a specific focus on IoT security. The paper aims to categorize and evaluate different AI approaches—including machine learning, deep learning, and reinforcement learning—highlighting their practical applications, strengths, and limitations in intrusion detection.

TABLE I
MILESTONES OF IDS

Year	Milestone
1980	First IDS concept by James Anderson
1987	Development of IDES at SRI International
1990	First commercial IDS: Haystack
1994	Deployment of Network-based IDS (NIDS)
1999	DARPA IDS Evaluation Dataset
2000	Launch of Snort, an open-source IDS
2003	Introduction of anomaly-based IDS
2005	Commercial use of machine learning in IDS
2010	Widespread adoption of SIEM systems
2015	Rise of AI-driven IDS solutions
2020	Integration of IDS with cloud security

A key objective is to analyze the challenges and emerging trends in AI-driven IDS, particularly in resource-constrained IoT environments. The paper discusses the necessity of lightweight IDS models to address issues related to computational overhead, scalability, and privacy concerns, ensuring that AI-based IDS solutions remain viable for IoT networks. Additionally, the study compares various AI techniques and examines their effectiveness in identifying both known and novel cyber threats.

Another objective is to establish a set of evaluation criteria for IoT-based IDS, ensuring that security solutions can be measured against essential performance and efficiency benchmarks.

II. HISTORY OF IDS

A. *Main milestones*

Intrusion Detection Systems have emerged as a fundamental component of cybersecurity, evolving significantly since their inception. The historical development of IDS provides a valuable overview of how these systems have adapted to the ever-changing landscape of cyber threats. From the early days of simple anomaly detection methods to the sophisticated, multi-layered defense mechanisms we see today, the history of IDS reflects the growing complexity and sophistication of both cyber-attacks and the technologies developed to counter them. Table I gives an overview of the significant milestones of IDS.

The concept of intrusion detection systems was first introduced by James Anderson in his seminal paper [3], which laid the foundation of understanding how monitoring system logs could help to detect unauthorized access to computer systems.

B. *Early ages*

This early concept focused on investigating audit trails to identify anomalies indicative of security breakings. The Intrusion Detection Expert System (IDES) was developed at SRI International's Computer Science Lab by Denning and Neumann in 1987 [4]. IDES was one of the first practical implementations of an IDS, designed to detect intrusions in real-time by analyzing system logs and network traffic for suspicious activity. It used statistical methods for reducing and analyzing audit trails [5]. After reengineering the prototype, the so-called Next-Generation Intrusion Detection Expert System (NIDES) was created to reach the production quality of the system. The observed behavior of the individual's system usage was compared to a profile-based value [6].

C. *Commercial systems*

„*Haystack*" was the first commercial IDS developed for the US Air Force in 1990. It marked the transition of IDS from research to practical application, providing real-time monitoring and alerting for potential security breaches within network environments [2].

D. *Network-based IDS*

1994 saw the Deployment of Network-based IDS (NIDS) which started to become widely recognized in the mid-1990s. These systems monitor network traffic in real-time, analyzing packet data to detect suspicious patterns that may indicate an attack [7]. The deployment of NIDS expanded the scope of IDS from individual host monitoring to entire network segments.

NIDS are typically deployed at strategic points within the network, such as at the gateway or in front of critical servers. They continuously capture and inspect packets traversing the network in real-time. NIDS uses predefined signatures or patterns of known threats to identify potential attacks. These signatures are similar to virus definitions used in antivirus software. When a packet matches a known signature, an alert is generated. In addition to signature-based methods, some NIDS employ anomaly-based detection. This approach involves creating a baseline of normal network behavior and flagging deviations from this norm as potential threats. Anomaly detection can help identify novel or previously unknown attacks.

There are challenges to NDIS. The biggest one is the high volume of data. NIDS needs to process large volumes of network traffic, which can be resource-intensive and may result in performance bottlenecks. Anomaly-based detection methods can generate false positives, where normal traffic is incorrectly flagged as malicious. The increasing use of encryption for network traffic can block NIDS's ability to inspect packets' content.

E. DARPA

The DARPA Intrusion Detection Evaluation Dataset, created by MIT Lincoln Laboratory in 1999, became a benchmark for testing and evaluating IDS performance. This dataset provided a standardized set of network traffic data containing both normal and malicious activities, enabling researchers to assess the effectiveness of various IDS approaches. The dataset records all network traffic, including the entire payload of each packet, in tcpdump format to enable comprehensive evaluation. It includes sniffed network traffic, Solaris BSM audit data, Windows NT audit data (for the DARPA 1999 dataset), and file system snapshots to identify intrusions against a test network composed of real and simulated machines. Background traffic was artificially generated, while attacks targeted real machines [8].

The dataset categorizes attacks into five main classes: Probe/Scan attacks, which scan networks to find valid IP addresses, active ports, OS types, and vulnerabilities; Denial of Service (DoS) attacks, which disrupt host or network services; Remote to Local (R2L) attacks, where an attacker gains local access without an account; User to Remote (U2R) attacks, where a local user obtains superuser or administrator privileges; and Data attacks, which involve exfiltration of sensitive files.

Despite its value, the DARPA dataset has faced criticism for its synthetic nature [12], which does not fully represent real-world traffic, and its limited representation of attack types, which may not reflect recent or diverse attack vectors. Additionally, performance evaluated with the DARPA 1999 dataset may not predict IDS effectiveness against modern threats or different network infrastructures.

Nevertheless, the DARPA dataset remains a significant tool for IDS research due to its detailed attack scenarios and comprehensive traffic records. It underscores the challenges of modeling network traffic and the need for continuous updates to reflect evolving threats and user behaviors. While more realistic datasets are needed for future research, the DARPA dataset's availability has been crucial for developing and evaluating IDS technologies.

F. SNORT

Snort, developed by Martin Roesch and released in 2000, revolutionized the IDS by providing a flexible, and open-source IDS tool. Snort's rule-based detection engine allowed users to write custom rules for identifying specific attack patterns, making it widely adopted in both academic and commercial environments. Snort is also recognized for its significant prevention capabilities, being the pioneer of the Intrusion Detection and *Prevention* System (IDPS) that supports both IDS and IPS modes, with significant prevention capabilities. As a Network Intrusion Detection and Prevention System (NIDPS). Snort is easy to configure and can effectively monitor network traffic. It compares received packets against known attack signatures and logs of detected attacks. In its IPS mode, Snort not only detects but also actively blocks malicious packets, preventing potential threats from causing harm to the network.

Snort uses Libpcap for packet capturing, followed by a decoder to interpret the captured packets. The preprocessor normalizes these packets, converting the traffic into a form that the detection engine can understand. The detection engine then applies predefined rules to identify and respond to malicious packets. In IPS mode, Snort’s ability to block malicious packets in real time enhances its prevention capabilities, making it a crucial tool for network security.

Developed in 1998 and continually updated by an active community, Snort [9] remains relevant in modern network security. Despite lacking a Graphical User Interface, this limitation can be addressed with open-source visualization tools. With the introduction of the multi-threaded variant, Snort 3 further enhances its efficiency and capability in preventing network intrusions.

G. Anomaly-based IDS

Anomaly-based IDS approaches were introduced to detect unknown attacks by identifying deviations from established normal behavior patterns [10]. These systems use statistical models, machine learning, and other techniques to learn what constitutes normal activity and flag any deviations as potential threats. This approach is particularly effective against novel attacks that do not match any known signatures.

H. Machine Learning

Commercial use of Machine Learning (ML) in IDS appeared in 2005 [11]. By this time, machine learning techniques began to be integrated into commercial IDS products. These techniques improved the accuracy of intrusion detection by enabling systems to learn from historical data [13] and adapt to evolving threat landscapes. Machine learning-based IDS [12] [14] could better identify complex attack patterns and reduce false positive rates. The following chapter will give an overview of the most common methods

I. SIEM

Another milestone in 2010 was the widespread adoption of Security Information and Event Management (SIEM) systems: tools in modern cybersecurity, designed to provide comprehensive monitoring, detection, and response capabilities [15]. They work by aggregating, correlating, and analyzing security data from various sources within an IT infrastructure. The primary goal of SIEM systems is to detect potential security threats, ensure compliance with regulatory requirements, and offer a centralized view of an organization's security posture.

SIEM systems collect data from multiple sources such as firewalls, intrusion detection/prevention systems (IDS/IPS), antivirus software, and other security devices. This data is normalized to maintain a consistent format, facilitating easier analysis. Once collected, the data is correlated to identify patterns that might indicate security threats. This correlation process links events from different systems to provide context and identify potential incidents, often based on predefined rules and policies.

Real-time monitoring is a critical function of SIEM systems, which continuously monitor network traffic and system activities for signs of malicious behavior. When potential threats are detected, the system generates alerts based on predefined thresholds and anomaly detection mechanisms. These alerts prompt security personnel to investigate further.

Incident response is another vital component of SIEM systems. They can automate responses to certain types of incidents by executing predefined actions such as blocking IP addresses or isolating affected systems. For more complex incidents, security teams use SIEM systems to investigate alerts, analyze the context, and determine appropriate response actions. SIEM systems also provide robust log management capabilities. They store logs from various sources for extended periods, which is crucial for compliance audits, forensic investigations, and trend analysis. Advanced search functionalities allow security analysts to quickly query logs and retrieve relevant information.

SIEM systems enhance an organization’s ability to detect, respond to, and manage security threats, playing a crucial role in maintaining a robust security posture.

J. AI-driven solutions

The rise of artificial intelligence (AI) and advanced analytics led to the development of AI-driven IDS solutions. These systems leverage deep learning, neural networks, and other AI technologies to detect sophisticated attacks with higher accuracy. AI-driven IDS can identify patterns and anomalies that traditional methods might miss, providing enhanced security insights. The following section will overview these methods.

K. Cloud security

With the increasing adoption of cloud computing, IDS solutions began to integrate with cloud security platforms around 2020. These integrated solutions provide comprehensive security monitoring across on-premises and cloud environments [16]. They address the unique challenges of cloud security, such as elastic scaling and dynamic infrastructure, ensuring continuous protection against cyber threats.

These milestones highlight the significant advancements in IDS technology over the years, reflecting the ongoing efforts to enhance network security and protect against evolving cyber threats.

III. Machine Learning and Artificial Intelligence

A. ML Methods

Machine learning and artificial intelligence approaches in intrusion detection systems comprise a wide range of techniques. ML methods rely on labeled data to train models that can classify network traffic as normal or malicious. Unsupervised learning techniques help identify patterns and anomalies in data without requiring labeled examples. Semi-supervised learning uses a small amount of labeled data to make use of a larger unlabeled dataset. Reinforcement learning is used to perform optimal actions for maximizing cumulative rewards in dynamic environments.

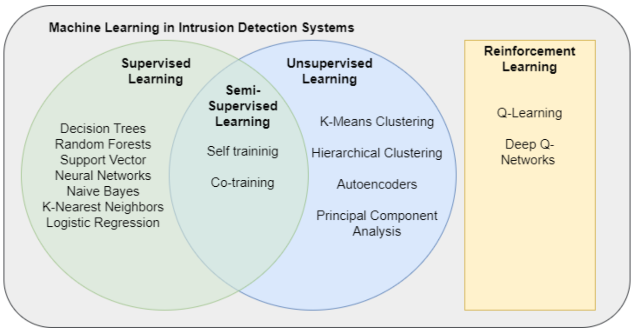


Fig. 1. Machine Learning methods in IDS

1) Decision trees

Th use of Decision trees technique IDS is popular due to their simplicity, interpretability, and efficiency in classifying network traffic. Decision trees operate by recursively partitioning the data into subsets based on the value of input features. This creates a tree-like model of decisions. Each node in the tree represents a feature, each branch represents a decision rule, and each leaf node represents an outcome or class label.

In the context of IDS, decision trees are used to analyze network traffic and identify patterns that distinguish normal activity from malicious activity. The process typically begins with the collection of network data, which is then pre-processed to handle missing values, noise, and irrelevant features. The decision tree algorithm is trained on this pre-processed data, learning the decision rules that best separate normal traffic from various types of attacks.

One of the key advantages of using decision trees in IDS is their ability to handle both categorical and numerical data, making them versatile for analyzing different types of network features. Moreover, decision trees can be easily visualized, allowing security analysts to understand the decision-making process and interpret the results.

The application of a C4.5-based decision tree for detecting intrusions in imbalanced datasets is demonstrated by [22]. Their approach involves using a supervised relative random sampling technique to balance the data before training the decision tree, which helps improve detection accuracy for minority attack classes. This method achieved high accuracy on benchmark datasets like NSL-KDD and CICIDS2017, proving the effectiveness of decision trees in handling real-world IDS challenges. This approach underscores the adaptability of decision trees in different IDS scenarios, from simple binary classification of attacks to more complex multi-class problems. Additionally, [23] highlights the use of decision trees for both binary and multiclass classification in IDS.

Decision trees can provide a robust framework for intrusion detection by leveraging their inherent interpretability and ability to model complex decision boundaries. Their application in IDS continues to evolve, integrating with advanced techniques to improve detection rates and reduce false positives.

2) Random forest

In IDS applications, Random Forests are particularly effective in classifying network traffic and detecting anomalies, which helps in identifying potential security threats. It is valuable for its accuracy, and ability to handle large datasets with a high number of dimensions. Random Forest is a collective learning method that builds multiple decision trees during training and merges their results to improve classification accuracy and control overfitting. Each decision tree in the forest is trained on a random subset of the training data with replacement (bootstrap sampling), and a random subset of features is used for splitting at each node. The final classification is determined by majority voting among the trees, which enhances the model's generalization capability and stability.

In the field of IDS, Random Forests have shown significant improvements in detection accuracy and reduction in false positives. For example, research conducted on the NSL-KDD dataset demonstrated that Random Forest models achieve high accuracy and stability in detecting various types of network attacks [19]. The study revealed that the model could handle imbalanced datasets effectively, which is crucial for IDS where the number of normal traffic samples often far exceeds that of attack samples.

Enhanced versions of Random Forest, combined with techniques like the Synthetic Minority Over-sampling Technique (SMOTE), further improve the detection performance by addressing the class imbalance. These enhancements help in generating more balanced training datasets, leading to better classification of minority classes (i.e., attack types). Additionally, studies have optimized Random Forest models by integrating feature selection methods to identify the most relevant features for intrusion detection, which boosts the model's accuracy and reduces computational complexity. In [20] Boruta feature selection was used with RF, providing some worse accuracy but at reduced memory usage. Boruta aims to find all features that are relevant for prediction can be computationally expensive, especially with large datasets [21].

3) Support Vector Machines

Support Vector Machine (SVM) is a machine learning technique used in intrusion detection systems in classifying network traffic. SVMs are particularly effective for IDS because they can handle high-dimensional data and provide a clear decision boundary between normal and malicious activities. SVMs are employed to detect intrusions by learning the characteristics of both normal and attack traffic. The SVM algorithm works by mapping input features into a high-dimensional space and finding the hyperplane that best separates the different classes of data. This makes SVMs highly effective for binary classification tasks commonly found in intrusion detection, where the goal is to distinguish between benign and malicious traffic.

One significant application of SVMs in IDS is detailed in a comprehensive survey of their use on the KDDCUP'99 and NSL-KDD datasets. This research highlights how SVMs can achieve high detection accuracy and low false positive rates by efficiently learning the distinctions between normal and attack behaviors in network traffic. The study emphasizes that SVMs perform well in various IDS scenarios, from detecting known attack patterns to identifying novel threats by modeling normal behavior and spotting deviations [24].

Another notable application is the use of One-Class Support Vector Machines (OCSVM) in anomaly-based Intrusion Detection Systems. OCSVM is particularly useful when only normal data is available for training. It works by constructing a hyperplane that separates normal data from potential anomalies. This method is advantageous in environments where labeled attack data is scarce or hard to obtain. Research combining OCSVM with autoencoders for feature extraction has shown improved detection rates, demonstrating the effectiveness of SVM in semi-supervised learning scenarios for IDS [25].

4) Neural Networks

Traditional neural networks, specifically Multi-Layer Perceptrons (MLPs), are widely used in intrusion detection systems that rely on labeled datasets. These networks consist of multiple layers of neurons, including an input layer, one or more hidden layers, and an output layer. For each layer each neuron is connected to every neuron in the next layer, with each connection having a weight that is adjusted during training.

In IDS applications, MLPs are trained on labeled datasets where each instance of network traffic is marked as either normal or malicious. The network learns to map input features to the correct labels through a process of forward propagation, where inputs are passed through the network to produce an output, and backpropagation, where errors are propagated back through the network to adjust the weights.

For example, [26] utilized an MLP trained on the KDD CUP 99 dataset, achieving high accuracy in detecting various types of network intrusions. Another study [28] demonstrated that MLPs, when trained on labeled datasets like NSL-KDD, could effectively distinguish between different types of attacks, providing a reliable method for intrusion detection. [27] compared MLP networks having different structures and achieved a detection accuracy of 95.6%.

These studies show that traditional neural networks, when properly trained on labeled datasets, can achieve significant performance in identifying and classifying network intrusions, thereby enhancing the security of network systems.

5) Naive Bayes

Naive Bayes is a probabilistic classifier based on Bayes' theorem, assuming independence among predictors. In intrusion detection systems using labeled datasets, Naive Bayes is employed to classify network traffic as normal or malicious by calculating the probability of each class given the input features. [29] investigated Naive Bayes' efficacy in IDS by training on the NSL-KDD 99 dataset. The classifier effectively identified various types of intrusions with high accuracy due to its simplicity and efficiency in handling large datasets. [30] highlighted the use of Naive Bayes in conjunction with feature selection methods, which improved the detection performance on the NSL-KDD dataset. The classifier's ability to handle categorical and numerical data made it suitable for various IDS applications, ensuring quick and accurate detection of network threats. [31] underscore Naive Bayes' role in IDS, leveraging its simplicity and effectiveness in processing labeled datasets to provide reliable network security.

6) K-Nearest Neighbors

K-Nearest Neighbors (kNN) is a simple, instance-based learning algorithm used in intrusion detection systems to classify network traffic using labeled datasets. The algorithm classifies a data point based on the majority class of its *k* nearest neighbors in the feature space.

The process starts with calculating the distance between the input data point and all points in the training set. Common distance metrics include Euclidean, Manhattan, and Minkowski distances. The *k* smallest distances are identified, and the class labels of these *k* nearest neighbors are retrieved. The input data point is then assigned the class that appears most frequently among these neighbors.

In IDS applications, kNN is effective for detecting various types of network intrusions. [32] investigated the algorithm's accuracy on the KDD CUP 99 dataset. The flexibility of KNN allows it to adapt to different types of network traffic and attacks, making it suitable for dynamic IDS environments. [33] highlighted KNN's ability to classify network activities efficiently using the NSL-KDD dataset, emphasizing the importance of selecting an appropriate *k* value and distance metric for optimal performance.

7) Logistic regression

Logistic regression is a statistical method used in IDS for binary classification tasks, distinguishing between normal and malicious network traffic using labeled datasets. The algorithm models the probability that a given input belongs to a particular class by applying a logistic function to a linear combination of input features.

In logistic regression, the algorithm learns the weights of input features during training to maximize the likelihood of correctly classifying the training data. The logistic function $L(x) = \frac{1}{1+e^{-x}}$, or – in neural network context often called as sigmoid function – maps the output to a probability between 0 and 1, which can be thresholded to decide the class label.

For IDS applications, logistic regression is employed to analyze network traffic features and predict the likelihood of an intrusion. [34] showed that logistic regression, when trained on the NSL-KDD dataset, effectively identified various types of network attacks with high accuracy. [35] demonstrated that logistic regression with multinominal regression model could enhance detection performance and reduce misclassification.

8) Semi-supervised learning techniques

Self-training and co-training are semi-supervised learning techniques used in intrusion detection systems to leverage both labeled and unlabeled data, enhancing detection performance without relying solely on extensive labeled datasets. Self-training involves an iterative process where an initial model is trained on a small, labeled dataset. This trained model then classifies the unlabeled data, and the high-confidence predictions are added to the labeled dataset as pseudo-labels. The model is retrained using this expanded dataset, iterating this process to gradually improve its accuracy. For example, [36] demonstrated that self-training improved IDS performance by generating more reliable pseudo-labels through uncertainty reduction techniques, such as using similarity graphs and graph convolutional networks to enhance the confidence and accuracy of predictions.

Co-training, on the other hand, uses two or more classifiers trained on different views or subsets of the features. Initially, each classifier is trained on a labeled dataset. Each classifier then labels the unlabeled data, and the most confident predictions from one classifier are added to the training set of the other classifier. This mutual reinforcement continues iteratively, improving the robustness of the model. Research has shown that co-training can significantly enhance IDS performance by combining classifiers' strengths and using ensemble methods to improve classification accuracy and reduce false positives [37].

These techniques are beneficial in IDS applications as they allow for the utilization of large amounts of unlabeled data, which is easier and cheaper to obtain than labeled data. By iteratively refining their models, self-training and co-training help IDS systems detect both known and unknown intrusions more effectively.

9) K-means clustering

K-means clustering is an unsupervised learning algorithm used for partitioning a dataset into distinct groups or clusters. In the context of intrusion detection systems, K-means clustering is applied to identify patterns in network traffic, distinguishing between normal and potentially malicious activities based on feature similarities. This method does not rely on labeled datasets; instead, it organizes the data into clusters where each data point belongs to the cluster with the nearest mean.

The K-means algorithm works as follows:

1. Initialize *k* cluster centroids randomly.
2. Assign each data point to the nearest centroid, forming k clusters.

3. Recalculate the centroids as the mean of all points in each cluster.
4. Repeat steps 2 and 3 until the centroids no longer change significantly or a maximum number of iterations is reached.

In IDS applications, K-means clustering helps in anomaly detection by grouping similar network behaviors. Points that fall into small or distant clusters may indicate anomalies or potential intrusions.

Comparing K-means clustering with K-Nearest Neighbors (KNN), we find that K-means is used for unsupervised learning, whereas KNN is a supervised learning algorithm. KNN classifies a data point based on the majority class of its *k* nearest neighbors, using labeled datasets to make predictions. K-means focuses on clustering similar data points without prior labels and is more suited for exploratory data analysis and anomaly detection.

[38] showed that while KNN excels in classification tasks with labeled datasets, K-means is more effective for initial pattern recognition and grouping of data in the absence of labels. Both methods can complement each other in IDS by using K-means to identify potential clusters of interest, which can then be further analyzed or classified using KNN with labeled data. These insights underscore the versatility of both algorithms in enhancing the detection capabilities of IDS through different approaches to data analysis.

10) Hierarchical clustering

Hierarchical clustering is a technique in cluster analysis aimed at constructing a hierarchy of clusters. It is especially valuable in intrusion detection systems for recognizing patterns and anomalies in network traffic. Unlike K-means clustering, hierarchical clustering does not require specifying the number of clusters as an input parameter of the algorithm. This method can be either agglomerative (bottom-up) or divisive (top-down). In agglomerative hierarchical clustering, each data point initially forms its own cluster. The algorithm then successively merges the nearest pairs of clusters until a single cluster is formed or a specified stopping criterion is reached. The steps are:

1. Compute the distance matrix for all data points.
2. Each data point forms a single cluster.
3. Merge the two closest clusters based on a chosen distance metric (e.g., single-linkage, complete-linkage).
4. Update the distance matrix to reflect the merger.
5. Repeat steps 3 and 4 until the desired number of clusters is achieved or all points are in one cluster.

In IDS applications, hierarchical clustering helps to discover the underlying structure of network traffic data, identifying groups of similar behaviors which can indicate potential intrusions. For example, [39] presented hierarchical clustering in identifying anomalies by analyzing the hierarchical structure of network connections and traffic patterns.

Hierarchical clustering, on the other hand, is unsupervised and does not use labeled data directly. It is more suited for exploratory data analysis and understanding the overall structure and relationships within the dataset, which can then be used to inform further analysis or classification tasks, potentially utilizing algorithms like KNN on labeled data.

11) Autoencoders

Autoencoders are a type of neural network used for unsupervised learning, particularly for feature learning and anomaly detection. In intrusion detection systems with labeled datasets, autoencoders are employed to detect anomalies by learning a compressed representation of the data and then reconstructing it [40]. The network consists of two main parts: the encoder, which compresses the input data into a latent space representation, and the decoder, which reconstructs the input data from this representation. The autoencoder is trained to minimize the reconstruction error, which is the difference between the input data and its reconstruction [41]. For IDS applications, the training is typically done on normal (benign) data. When the autoencoder encounters new data, it will reconstruct normal data well but will struggle with anomalous (malicious) data, resulting in higher reconstruction errors. These errors can then be used to flag potential intrusions.

12) Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction technique used in intrusion detection systems to transform high-dimensional data into a lower-dimensional space while preserving most of the variance in the data. PCA identifies the principal components, which are the directions of maximum variance in the dataset, and projects the data onto these new axes. This process reduces the number of features while retaining the most important information, making it easier to analyze and visualize the data. In IDS applications with labeled datasets, PCA is often used as a pre-processing step to enhance the performance of machine learning algorithms. By reducing the dimensionality, PCA helps in mitigating the curse of dimensionality, reducing computational costs, and improving the efficiency of the learning algorithms.

For example, [42] used PCA for feature reduction in a network Intrusion Detection System. The researchers of [43] applied PCA to the KDD CUP 99 dataset, reducing the number of features before applying classification algorithms like SVM and kNN. The results showed improved detection accuracy and reduced training time. NSL-KDD Dataset based on PCA-Fuzzy Clustering-KNN was used in [44].

13) Reinforcement learning methods

Reinforcement learning (RL) methods are used in Intrusion Detection Systems to enable models to learn optimal actions through trial and error by interacting with an environment. The primary goal is to maximize a cumulative reward signal. Unlike immediate rewards, which are given after each action, the cumulative reward is the total accumulated reward over a sequence of actions. The primary objective in RL is to develop a strategy (or policy) that maximizes this cumulative reward over time, not just the immediate reward.

$$G_t = \sum_{k=0}^{\infty} \gamma^k \cdot R_{t+k+1}$$

where

G_t is the cumulative reward starting from time step t

γ is the discount factor, which lies between 0 and 1 and

determines the importance of future rewards.
R_{t+k+1} is the reward received at time step t+k+1.

Two prominent RL methods in IDS are Q-learning and Deep Q-Networks (DQN). Q-learning is a model-free RL algorithm that aims to learn the optimal policy for an agent to take actions in a given environment. The core component of Q-learning is the Q-table, which stores the value of taking a particular action in a particular state. The algorithm updates the Q-values using the Bellman equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \cdot [r + \gamma \cdot max_{a'} Q(s',a') - Q(s,a)]$$

where
s is the current state,
a is the action taken,
r is the reward received,
s' is the next state,
a$'$ is the next action,
α is the learning rate, and
γ is the discount factor.

In IDS applications, Q-learning helps in adapting to new types of attacks by updating policies based on feedback from the network environment. [45] used the effectiveness of Q-learning in detecting and responding to network intrusions, showing that Q-learning could dynamically adjust its detection strategy based on observed network behaviors. Deep Q-Networks (DQN) extend Q-learning by integrating deep learning, allowing the algorithm to handle high-dimensional state spaces. Instead of maintaining a Q-table, DQN uses a neural network to approximate the Q-values. This enables DQN to scale to more complex environments that are impractical for Q-learning due to the large state-action space. The DQN algorithm involves training a neural network to predict Q-values and using experience replay to stabilize training.

In IDS applications, DQNs can detect complex patterns and adapt to evolving threats more effectively than traditional Q-learning. [46] applied DQNs to network intrusion detection, demonstrating the deep neural network's ability to learn intricate features from raw network traffic data. The study showed that DQN outperformed traditional Q-learning and other Machine Learning methods in identifying sophisticated attacks.

Comparing Q-learning and DQN, Q-learning is simpler and more straightforward, suitable for environments with smaller state-action spaces. DQN, on the other hand, is more powerful and scalable, capable of handling high-dimensional data and complex scenarios at the cost of increased computational resources and training time.

B. *AI methods*

Artificial intelligence approaches in IDS include expert systems and rule-based systems that use predefined rules to detect known threats. Signature-based detection, a form of expert systems, recognizes patterns that match known threat signatures. Anomaly detection uses statistical methods, heuristic methods, and behavioral analysis to identify deviations from normal behavior, which might indicate potential threats. Deep learning techniques, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory (LSTM) networks, are employed to analyze complex data patterns and sequential data like log files or time-series data. Hybrid systems combine machine learning with traditional rule-based methods, integrating multiple models to enhance detection accuracy and reduce false positives.

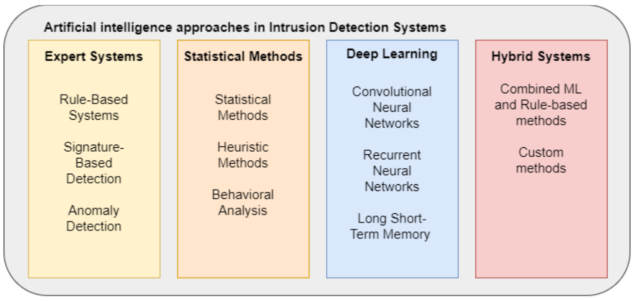


Fig. 2. AI approaches in IDS

1) *Expert systems*

Expert systems are artificial intelligence systems that use knowledge and inference procedures to solve problems that typically require human expertise. In the context of Intrusion Detection Systems, expert systems include rule-based systems, signature-based detection, and anomaly detection. Rule-based systems operate by applying predefined rules to the data being analyzed. These rules are created by domain experts and represent the knowledge about what constitutes normal and abnormal behavior in a network. The system compares incoming data against these rules to detect potential intrusions. For example, if a rule specifies that multiple failed login attempts within a short period indicate a brute-force attack, the system will flag such occurrences. Signature-based detection is a subtype of rule-based systems where the rules, or signatures, are patterns of known threats. These signatures are derived from previous attack patterns and behaviors. When network traffic matches a known signature, the system raises an alert. Signature-based IDS are highly effective at detecting known attacks but struggle with new or evolving threats. [47] demonstrated the efficacy of signature-based IDS in quickly identifying known malware based on established signatures.

Anomaly detection, on the other hand, builds a model of normal network behavior and identifies deviations from this model as potential intrusions. This method can detect previously unknown threats by recognizing unusual patterns of activity. Anomaly detection can use statistical methods, machine learning algorithms, or a combination of these to define what constitutes normal behavior [48].

Comparing these approaches, rule-based systems are easy to understand but can be limited by the completeness and accuracy of the rules. Signature-based detection is very effective for known threats but cannot detect new or modified attacks. Anomaly detection offers the advantage of identifying novel threats by focusing on deviations from normal behavior, but it can generate false positives if the model of normal behavior is not accurately defined or if legitimate activities deviate from the norm.

2) *Statistical methods*

Statistical methods for Intrusion Detection Systems rely on mathematical models to analyze network data and detect anomalies indicative of potential intrusions [49]. These methods can be broadly categorized into heuristic methods and behavioral analysis. Heuristic methods use simple, rule-of-thumb strategies based on prior knowledge and experience to detect anomalies in network traffic. These methods often involve setting thresholds for various metrics, such as the number of failed login attempts or the volume of traffic from a single IP address. When these thresholds are exceeded, the system flags the activity as suspicious. Heuristic methods [51] are relatively easy to implement and understand but can generate false positives if the thresholds are not set accurately or if normal behavior varies widely. Behavioral analysis [50] involves creating profiles of normal network behavior and monitoring for deviations from these profiles. This approach uses statistical techniques to model the typical patterns of activity for users and systems. Any significant deviation from these patterns is considered an anomaly and potentially an intrusion. Behavioral analysis can adapt to changes in network behavior over time, reducing the likelihood of false positives. However, it requires a substantial amount of historical data to build accurate models and can be computationally intensive.

In IDS applications, heuristic methods provide quick and straightforward detection mechanisms. Behavioral analysis, on the other hand, offers a more dynamic and comprehensive approach. Comparing these methods, heuristic methods are simpler and faster to deploy but may lack the sophistication needed to handle complex or evolving threats. Behavioral analysis provides a deeper and more adaptive understanding of network activity but requires more data and computational resources to implement effectively.

3) *Convolutional Neural Networks*

Convolutional Neural Networks (CNNs) belong to deep learning models particularly effective for analyzing grid-like data structures, such as images and time-series data. In Intrusion Detection Systems, CNNs are employed to detect complex patterns in network traffic data, enhancing the ability to identify intrusions [52][53].

A CNN consists of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers apply a series of filters (kernels) to the input data to extract high-level features, such as edges or patterns in network traffic. The pooling layers down-sample the spatial dimensions of the data, reducing the computational load and emphasizing the most critical features. The fully connected layers, usually at the end of the network, perform classification based on the extracted features. The CNN algorithm operates as follows:

- Input Layer: Raw network traffic data is fed into the network. In IDS applications, this data can be represented as a matrix where rows represent different time steps and columns represent various traffic features.
- Convolutional Layer: Filters slide over the input data, performing convolution operations to detect local patterns. Each filter generates a feature map that highlights the presence of specific patterns.
- Activation Function: An activation function, typically Rectified Linear Unit (ReLU) [56], is applied to introduce non-linearity into the model.
- Pooling Layer: Feature maps are down-sampled using operations like max pooling or average pooling, which reduce dimensionality while retaining important information.
- Fully Connected Layer: The pooled feature maps are flattened into a vector and passed through one or more fully connected layers, which combine the features to classify the input data.
- Output Layer: The final layer produces a probability distribution over the class labels, determining whether the input data represents normal traffic or an intrusion.

In IDS applications, CNNs can learn to recognize intricate patterns in network traffic that may indicate malicious activity. For example, [54] applied CNNs to network traffic data from the NSL-KDD dataset. The CNN model demonstrated superior performance in detecting various types of network intrusions compared to traditional machine learning methods, thanks to its ability to automatically extract relevant features from raw data. [55] used CNNs to analyze time-series data from network traffic. The researchers found that CNNs could effectively capture temporal patterns and detect anomalies with high accuracy, reducing false positives and improving overall IDS performance.

CNNs provide a powerful tool for IDS by leveraging their deep learning capabilities to automatically learn and detect complex patterns in network traffic, making them highly effective for identifying sophisticated intrusions.

4) *Recurrent Neural Networks*

Recurrent Neural Networks (RNNs) are a type of deep learning model designed to handle sequential data, making them suitable for intrusion detection systems that analyze time-series network traffic data. RNNs are particularly effective in capturing temporal dependencies and patterns, which are crucial for detecting anomalies and intrusions in network behavior over time.

RNNs operate by maintaining a hidden state that captures information from previous time steps, allowing them to process sequences of data. The hidden state is updated at each time step based on the current input and the previous hidden state, enabling the network to retain information about past inputs. The RNN algorithm functions as follows:

- Input Layer: Sequential network traffic data is fed into the RNN. This data can include various features such as packet size, inter-arrival time, and protocol type.
  - Hidden Layer: At each time step t the RNN computes the hidden state h_t using the current input x_t and the previous hidden state h_{t-1}. The hidden state is updated using the formula:

$$h_t = \sigma \cdot (W_h \cdot h_{t-1} + W_x \cdot x_t + b)$$

where W_h and W_x are weight matrices,
b is the bias, and
σ is an activation function such as hyperbolic tangent (tanh) or ReLU.

- Output Layer: The output at each time step is computed based on the current hidden state. For classification tasks in IDS, a softmax layer is typically used to produce a probability distribution over the possible classes (e.g., normal traffic or different types of attacks).
- Training: The network is trained using backpropagation through time (BPTT), which involves unrolling the RNN for a number of time steps and applying the backpropagation algorithm to update the weights.

RNNs are well-suited for IDS because they can model the temporal dynamics of network traffic, capturing patterns that may indicate an ongoing intrusion. For instance, a study in the IEEE Access journal applied RNNs to the NSL-KDD dataset, demonstrating that RNNs could effectively detect intrusions by learning the temporal relationships in network traffic data.

[58] highlighted the use of Long Short-Term Memory (LSTM) networks, a variant of RNNs, to improve IDS performance. LSTMs address the vanishing gradient problem in standard RNNs by incorporating memory cells that can retain information over longer periods, enhancing the model's ability to detect slow, evolving attacks. RNNs [57] and their variants, such as LSTMs, provide powerful tools for IDS by leveraging their capability to learn from sequential data, capturing temporal patterns that are essential for accurate intrusion detection.

IV. IDS for IoT

The Internet of Things (IoT) represents a revolutionary shift in how devices interact and communicate with each other, promising unprecedented levels of convenience, efficiency, and automation. IoT encompasses a wide range of devices, from smart home appliances and wearable fitness trackers to industrial sensors and connected vehicles. However, this interconnected landscape also introduces significant security challenges, making Intrusion Detection Systems crucial for ensuring the integrity, confidentiality, and availability of IoT networks.

IoT devices often operate in a distributed environment, connected to the internet and various other networks, making them susceptible to a wide array of cyber threats. These threats include unauthorized access, data breaches, denial of service (DoS) attacks, malware infections, and more. The sheer number of IoT devices, many of which have limited processing power and memory, further complicates security measures. Traditional security solutions like firewalls and antivirus software are often inadequate for IoT due to their resource-intensive nature and the diverse range of devices and protocols in use.

Intrusion Detection Systems play an important role in safeguarding IoT environments by monitoring network traffic, detecting anomalies, and identifying potential security breaches. IDS can be broadly categorized into two types: Network-based IDS (NIDS) and Host-based IDS (HIDS). NIDS monitors network traffic for suspicious activity, while HIDS monitors the activities within individual devices. Both types are essential for comprehensive IoT security.

One of the primary functions of IDS is to provide real-time monitoring and alerting for potential security threats. This capability is particularly important in IoT environments where rapid detection and response can prevent minor incidents from escalating into significant security breaches. For instance, in smart home networks, an IDS can detect unusual traffic patterns that may indicate a compromised device attempting to communicate with external servers.

IoT networks can have a variety of behaviors depending on the devices and their usage patterns. IDS equipped with anomaly detection can learn the normal behavior of the network and flag deviations that may signify an intrusion. For example, an IDS can detect an increase in network traffic from a typically dormant device, suggesting that the device might have been hijacked for a botnet attack.

IoT environments face a broad spectrum of threats, ranging from simple brute-force attacks to sophisticated malware designed to exploit specific vulnerabilities. IDS can help mitigate these threats by identifying known attack signatures and using heuristic methods to detect new, previously unknown threats. This dual approach ensures that both common and novel attacks can be detected and mitigated.

IoT devices often have constrained resources, making it challenging to implement heavy security protocols directly on each device. IDS can offload the burden of security monitoring to more capable devices or cloud services, ensuring that security measures do not impede the performance of the IoT devices themselves.

As IoT devices become more prevalent in sectors like healthcare, finance, and critical infrastructure, compliance with regulatory standards becomes essential. IDS can help organizations meet these requirements by providing detailed logs and reports of network activity, facilitating audits, and ensuring that security measures are in place to protect sensitive data.

Implementing IDS in IoT also presents several challenges. The heterogeneous nature of IoT devices means that they often run on different platforms and operating systems, making it difficult to create a one-size-fits-all security solution. Additionally, the resource constraints of many IoT devices limit the types of security measures that can be implemented directly on the device. Network-based IDS solutions must be capable of handling high volumes of data from numerous devices without becoming a bottleneck.

Another challenge is the need for continuous updates. As new threats emerge, IDS must be constantly updated with the latest signatures and detection algorithms to remain effective. This requires a robust infrastructure for delivering updates and patches to IoT devices and IDS solutions.

Privacy concerns also come into play. IDS involves monitoring and analyzing network traffic, which can raise privacy issues, especially in environments where sensitive personal data is transmitted. Ensuring that IDS solutions comply with privacy regulations and protect user data is paramount.

Despite these challenges, the benefits of implementing IDS in IoT environments are substantial. By providing real-time monitoring, detecting anomalies, mitigating diverse threats, and ensuring compliance with regulatory requirements, IDS plays a critical role in protecting IoT networks. As IoT continues to expand and evolve, the importance of robust, adaptive IDS solutions will only grow, making them an indispensable component of modern cybersecurity strategies.

Research and development in IDS for IoT are ongoing, with many promising approaches being explored [60]. Machine learning and artificial intelligence are increasingly being integrated into IDS to enhance their ability to detect and respond to new and sophisticated threats. [60] demonstrated that a Raspberry PI can fulfill the role of IDS in an IoT environment. A system designed for preventing botnet attacks is discussed in [62]. The state-of-the-art IDS methods in IoT networks are reviewed in [61].

In an IoT environment, a lightweight IDS system needs to be used with optimized algorithms that reduce computational overhead while maintaining effective threat detection capabilities. These models should extensively use techniques such as feature selection, dimensionality reduction (e.g., Principal Component Analysis), and low-complexity classifiers (e.g., decision trees, Naïve Bayes, and lightweight neural networks) to enhance efficiency. Additionally, hybrid approaches that combine signature-based detection with anomaly-based methods can improve detection accuracy while minimizing false positives. To further optimize performance, edge computing-based IDS solutions distribute detection tasks across IoT gateways reducing the need for constant cloud communication and enhancing real-time threat response. By adopting these lightweight strategies, IDS systems can effectively safeguard IoT networks against evolving cyber threats without imposing excessive computational demands.

A. Evaluation criterions

Intrusion Detection Systems (IDS) are evaluated based on several key factors that determine their effectiveness, efficiency, and applicability in different environments. While generic IDS and IoT IDS share some common evaluation criteria, IoT-specific IDS must address additional challenges due to the unique constraints and characteristics of IoT environments. The table below summarizes the key evaluation factors for both types of IDS.

TABLE II
IDS EVALUATION CRITERIONS

Evaluation Factor	Generic IDS	IoT IDS
Detection Accuracy	Measures the system's ability to correctly identify intrusions and minimize false positives/negatives.	Equally important but must also consider lightweight detection techniques to maintain efficiency.
False Positive Rate	Essential to minimize unnecessary alerts that make the security team overload	False positives can disrupt normal IoT operations and lead to unnecessary energy consumption.
False Negative Rate	IDS should avoid missing actual attacks, as this can compromise security.	More critical in IoT, as undetected attacks can disrupt real-time operations, e.g., healthcare or industrial IoT.
Real-time Performance	Response time is important but may tolerate slight delays in non-critical systems.	Extremely crucial in IoT environments where real-time threat detection is necessary (e.g., autonomous vehicles, industrial control systems).
Scalability	Must handle large networks but is generally deployed on powerful infrastructure	Must efficiently manage thousands of resource-constrained IoT devices across distributed environments.
Computational Overhead	Can be relatively high, especially for AI-based IDS, since enterprise systems have powerful computing resources	Must be **low**, as many IoT devices have limited CPU, memory, and energy constraints.
Network Overhead	IDS may introduce moderate network overhead for monitoring and logging.	Must **minimize communication overhead**, as IoT devices rely on low-bandwidth networks.
Adaptability to New Attacks	AI-based IDS and rule-based IDS must be regularly updated to detect evolving threats.	Requires lightweight, adaptive models that can learn new threats with minimal retraining, as frequent updates may not be feasible for IoT.
Energy Efficiency	Not a primary concern in traditional IDS.	**Highly important**, as IoT devices often run on battery power and cannot support continuous, power-intensive monitoring.
Privacy & Data Sensitivity	Monitors user/system activity but typically operates within a secure infrastructure.	Critical in IoT healthcare, smart homes, and industrial IoT, where personal/sensitive data must be protected from breaches.
Deployment Model	Typically centralized in enterprise networks with a dedicated security team.	Often decentralized, relying on edge computing or fog computing to distribute detection closer to the IoT devices.
Robustness Against Adversarial Attacks	IDS must handle sophisticated attack strategies like polymorphic malware.	IoT IDS is more vulnerable to adversarial ML attacks, sensor spoofing, and firmware-based exploitation.
Integration with Security Frameworks	Works alongside firewalls, SIEM systems, and endpoint security tools.	Needs **lightweight security integration**, often in resource-constrained environments where traditional firewalls may not be available.

Lightweight Security Integration refers to the incorporation of security mechanisms, into resource-constrained environments while maintaining minimal impact on computational resources, energy consumption, and network performance, which focuses on efficient, adaptive, and decentralized security measures that ensure protection without degrading device functionality. It utilizes event-driven detection (e.g., anomaly detection triggered by specific behaviors) instead of continuous monitoring. It is important to use lightweight cryptographic protocols (e.g., elliptic curve cryptography) instead of computationally expensive security methods.

V. Sample Datasets for IDS Development

Intrusion Detection Systems datasets are essential for researchers as they provide a standardized and reliable foundation for developing, testing, and validating IDS algorithms. These datasets typically contain a mix of normal and malicious network traffic, enabling researchers to simulate real-world scenarios and measure the effectiveness of their detection methods. By using these datasets, researchers can benchmark their solutions against existing techniques, identify strengths and weaknesses, and iteratively improve their models. Furthermore, standardized datasets facilitate reproducibility in research, allowing different researchers to compare their results and advancements consistently.

The relevance of IDS datasets lies in their ability to represent a wide range of attack vectors and network behaviors, making them invaluable for developing robust and adaptive IDS solutions. Historical datasets like KDD Cup 1999 and DARPA [17] have laid the groundwork for intrusion detection research, while more recent datasets such as CIC-IDS2017 and TON_IoT reflect contemporary network environments and sophisticated attack strategies. These datasets not only help in understanding the evolution of cyber threats but also in developing next-generation IDS that can protect against emerging threats. By providing diverse and comprehensive data, IDS datasets empower researchers to push the boundaries of cybersecurity and enhance the resilience of networked systems.

IDS datasets are provided in various formats, each serving different purposes and offering unique advantages for data analysis and intrusion detection research, see Table II. Understanding these formats is crucial for effectively utilizing the datasets in IDS development and evaluation.

PCAP (Packet Capture) files contain raw network traffic data captured at the packet level. Each packet includes details such as source and destination IP addresses, port numbers, protocols, and payload data. PCAP is widely used in network analysis because it allows researchers to reconstruct network sessions and analyze the detailed behavior of network traffic.

NetFlow is a network protocol developed by Cisco for collecting IP traffic information. NetFlow records summarize flows of network traffic, providing aggregated information about the source and destination addresses, ports, protocols, and the volume of data transferred. This format is useful for analyzing traffic patterns and detecting anomalies over longer periods.

Logs are text-based files that record events generated by network devices, operating systems, and applications. Each log entry typically includes a timestamp, the source of the log, and a message describing the event. Logs are invaluable for forensic analysis and for identifying patterns of behavior that may indicate security incidents.

TABLE III
IDS Datasets

Name	Description/ Location	Year	Labe lled	Format
MAWI	Traffic traces from the WIDE project, used for traffic analysis and anomaly detection https://mawi.wide.ad.jp/mawi	2006 -	No	PCAP
B TON_IoT	IoT and network traffic data from a simulated smart city environment. https://research.unsw.edu.au/projects/toniot-datasets	2019	Yes	CSV, JSON
CIC-DDoS2019	Data from various types of DDoS attacks for detection and mitigation research. https://www.unb.ca/cic/datasets/ddos-2019.html	2019	Yes	CSV, PCAP
AAGM2013	Network traffic data for evaluating anomaly detection methods. https://csr.lanl.gov/data/audit/	2013	Yes	CSV
BoT-IoT	Synthetic IoT traffic with normal and attack scenarios for intrusion detection.	2018	Yes	CSV, PCAP
UGR'16	Real traffic captures from UGR, including benign and malicious traffic. https://nesg.ugr.es/nesg-ugr16	2016	Yes	CSV, NetFlow
CTU-13	Botnet traffic mixed with normal traffic from the Czech Technical University. https://www.stratosphereips.org/datasets-ctu13	2011	Yes	PCAP, CSV
Twente	Network traffic data with various network events for traffic analysis and anomaly detection. https://data.4tu.nl/articles/dataset/TNTS/Twente_University_Network_Traffic_Dataset/12781370	2015	Yes	NetFlow, CSV
SUEE	Dataset from Sharif University of Technology with a variety of attack scenarios. http://ocslab.hksecurity.net/Datasets/suee	2012	Yes	CSV
CACTI	Comprehensive archive of cyber threat intelligence, including network traffic and log files. https://github.com/CACTI-dataset	2020	Yes	JSON, CSV
DARPA98	Network traffic data with simulated normal and attack activities, foundational for IDS research. https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset	1998	Yes	PCAP, Logs
KDD Cup 1999	Derived from DARPA98, this dataset is used for network intrusion detection research. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html	1999	Yes	CSV

VI. Further Discussion

In conclusion, the integration of AI methods into Intrusion Detection Systems represents a significant advancement in the field of cybersecurity. AI techniques, including machine learning, deep learning, and neural networks, offer robust and adaptive solutions for detecting a wide range of cyber threats. These methods enhance the capability of IDS to identify both known and novel attacks with greater accuracy and efficiency compared to traditional detection approaches.

The application of machine learning algorithms has proven effective in analyzing large volumes of network traffic data, identifying patterns, and detecting anomalies that may indicate potential security breaches. Techniques such as supervised learning, unsupervised learning, and reinforcement learning have been successfully employed to improve detection rates and reduce false positives in IDS.

Deep learning, particularly through the use of convolutional neural networks (CNNs) and recurrent neural networks (RNNs), has shown exceptional promise in handling complex and high-dimensional data. These models are capable of learning intricate features from raw data, enabling more precise identification of sophisticated attack vectors. Additionally, the ability of deep learning models to continually learn and adapt to new threats makes them invaluable in the ever-evolving landscape of cybersecurity.

Despite the advancements, there are challenges associated with the deployment of AI-based IDS. The need for large, high-quality datasets for training, the computational resources required for model training and deployment, and the potential for adversarial attacks on AI models are critical issues that need to be addressed. Ongoing research and development are essential to overcome these challenges and to further enhance the performance and reliability of AI-driven IDS.

In summary, AI methods have revolutionized intrusion detection systems, providing more intelligent, adaptable, and efficient tools for safeguarding network security. Future research should focus on refining these techniques, addressing their limitations, and exploring new AI paradigms to keep pace with the advancing threat landscape. The continuous evolution of AI-driven IDS will play a crucial role in protecting digital infrastructures and ensuring the integrity and confidentiality of information in the digital age.

The computational demands of Intrusion Detection Systems are a significant consideration, particularly with the integration of advanced artificial intelligence (AI) techniques. AI-driven IDS, such as those utilizing machine learning and deep learning algorithms, require substantial processing power to analyze vast amounts of network traffic data in real-time. Training complex models, especially deep learning networks, involves intensive computations that necessitate the use of high-performance computing resources, including GPUs and distributed computing frameworks. Additionally, the deployment of these models in operational environments demands continuous monitoring and analysis, which can strain the computational resources of a network. The need for high-speed data processing, large-scale storage, and efficient memory management further adds to the computational burden. Consequently, the infrastructure supporting AI-based IDS must be robust and scalable to handle the high computational requirements, ensuring that the system can operate effectively without compromising performance or security. Addressing these computational challenges is crucial for the successful implementation and operation of intelligent IDS.

In the context of IoT-based Intrusion Detection Systems (IDS), selecting the most suitable machine learning and artificial intelligence methods requires balancing accuracy, computational efficiency, adaptability, and real-time processing. Supervised learning methods are widely used for detecting known attack patterns. Decision trees provide a simple and interpretable approach with minimal computational overhead, making them suitable for IoT devices with limited processing power. Random forests improve upon decision trees by aggregating multiple classifiers, offering higher detection accuracy while remaining efficient. Naïve Bayes classifiers, due to their probabilistic nature, are extremely fast and can work effectively in low-resource environments, making them a viable choice for IoT IDS with minimal training data. Support vector machines perform well in binary classification tasks and are particularly useful when labeled attack data is available, though they can be computationally demanding when dealing with large datasets.

For detecting unknown threats, unsupervised learning methods play a critical role. K-means clustering is effective for identifying anomalies by grouping network behaviors into clusters, though its performance depends on selecting an optimal number of clusters. Autoencoders offer a more advanced approach by learning normal network behavior and identifying deviations as potential attacks. These models reduce the feature space while preserving important data characteristics, making them efficient for IoT environments. One-class support vector machines provide another effective anomaly detection technique by modeling normal traffic and flagging deviations, which is particularly useful when labeled attack data is scarce.

Hybrid AI approaches enhance intrusion detection by combining multiple techniques to improve both detection accuracy and efficiency. Federated learning is increasingly relevant in IoT security as it enables decentralized model training across multiple devices without transferring sensitive data to a central server. This method enhances privacy while allowing IoT devices to collaboratively improve their IDS models. Reinforcement learning introduces an adaptive mechanism that enables IDS to continuously learn from its environment and adjust detection strategies dynamically. This approach is particularly valuable for evolving IoT security threats, as it does not rely on pre-labeled datasets.

To minimize latency and energy consumption, lightweight AI models optimized for edge computing environments are essential. Lightweight neural networks, designed with quantization and model compression techniques, allow IDS to detect threats directly on IoT gateways or edge devices without relying on cloud-based processing. TinyML [64], a specialized subset of machine learning designed for microcontrollers, further enables on-device intrusion detection with ultra-low power consumption, making it a promising solution for battery-operated IoT systems.

IoT IDS must integrate models that not only achieve high detection accuracy but also maintain low computational complexity and real-time responsiveness. Decision trees and Naïve Bayes are ideal for signature-based detection, while autoencoders and one-class SVMs are effective for anomaly detection. Federated learning and reinforcement learning offer adaptive, privacy-preserving solutions, while lightweight neural networks and TinyML enable real-time, energy-efficient intrusion detection at the network edge. The combination of these techniques ensures that IoT IDS can provide robust security without overburdening constrained devices and networks.

Acknowledgment

The authors would like to express their sincere gratitude to the University of Miskolc, Institute of Information Science, for their generous funding and support. This research would not have been possible without their financial assistance and the valuable resources they provided. We are deeply appreciative of the opportunities and facilities made available to us, which have greatly contributed to the advancement of this study.

References

[1] D. Farmer. *Cops (computer oracle and password system)* (1989) [Online]. Available: http://coast.cs.purdue.edu/pub/tools/unix/scanners/cops/

[2] S. E. Smaha. (1988). Haystack: an intrusion detection system. *[Proceedings 1988] Fourth Aerospace Computer Security Applications*, 37–44. DOI: 10.1109/acsac.1988.113412

[3] Anderson, J. P. (1980). "Computer Security Threat Monitoring and Surveillance." *Technical report*, James P. Anderson Co.

[4] Denning, Dorothy, and Peter G. Neumann. Requirements and model for IDES-a real-time intrusion-detection expert system. Vol. 8. Menlo Park: SRI International, 1985.

[5] Denning, D. E. (1987). "An Intrusion-Detection Model." *IEEE Transactions on Software Engineering*.

[6] Amoroso, E. G. (1999). "Intrusion Detection: An Introduction to Internet Surveillance, Correlation, Trace Back, Traps, and Response." Intrusion. Net Books.

[7] Northcutt, S., & Novak, J. (2002). "Network Intrusion Detection." New Riders Publishing. DOI: 10.1201/1079/43253.27.7.20000101/30304.4

[8] Lippmann, R. P., et al. (2000). "Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation." *Proceedings DARPA Information Survivability Conference and Exposition*. DOI: 10.1109/discex.2000.821506

[9] Roesch, M. (1999). "Snort: Lightweight Intrusion Detection for Networks." *Proceedings of the 13th USENIX Conference on System Administration*.

[10] Patcha, A., & Park, J. M. (2007). "An overview of anomaly detection techniques: Existing solutions and latest technological trends." *Computer Networks*. DOI: 10.1016/j.comnet.2007.02.001

[11] Kotsiantis, S. B., Zaharakis, I. D., & Pintelas, P. E. (2006). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26, 159–190. DOI: 10.1007/s10462-007-9052-3

[12] Sommer, R., & Paxson, V. (2010). "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection." *IEEE Symposium on Security and Privacy*. DOI: 10.1109/sp.2010.25

[13] Chuvakin, A., Schmidt, C., & Phillips, D. (2013). "Logging and Log Management: The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management." *Syngress*. DOI: 10.1016/b978-1-59-749635-3.00014-2

[14] Buczak, A. L., & Guven, E. (2016). "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection." *IEEE Communications Surveys & Tutorials*. DOI: 10.1109/comst.2015.2494502

[15] Muhammad, Adabi Raihan, Parman Sukarno, and Aulia Arif Wardana. "Integrated security information and event management (SIEM) with intrusion detection system (ids) for live analysis based on machine learning." *Procedia Computer Science* 217 (2023): 1406–1415. DOI: 10.1016/j.procs.2022.12.339

[16] Modi, C., Dhiren Patel, Bhavesh Borisaniya, Hiren Patel, Avi Patel, Muttukrishnan Rajarajan (2013). "A survey of intrusion detection techniques in Cloud." *Journal of Network and Computer Applications*. DOI: 10.1016/j.jnca.2012.05.003

[17] Thomas, C., Sharma, V., & Balakrishnan, N. (2008). Usefulness of DARPA dataset for intrusion detection system evaluation. SPIE Defense + Commercial Sensing. DOI: 10.1117/12.777341

[18] Negandhi, P., Trivedi, Y., & Mangrulkar, R. (2019). Intrusion detection system using random forest on the NSL-KDD dataset. In *Emerging Research in Computing, Information, Communication and Applications: ERCICA* 2018, Volume 2 (pp. 519–531). Springer Singapore. DOI: 10.1007/978-981-13-6001-5_43

[19] Wu, T., Fan, H., Zhu, H., You, C., Zhou, H., & Huang, X. "Intrusion detection system combined enhanced random forest with SMOTE algorithm." *EURASIP Journal on Advances in Signal Processing* 2022.1 (2022): 39. DOI: 10.21203/rs.3.rs-270201/v1

[20] Hasan, M. A. M., Nasser, M., Ahmad, S., & Molla, K. I. (2016). Feature selection for intrusion detection using random forest. *Journal of Information Security*, 7(3), 129–140. DOI: 10.4236/jis.2016.73009

[21] Subbiah, S., Anbananthen, K. S. M., Thangaraj, S., Kannan, S., & Chelliah, D. (2022). Intrusion detection technique in wireless sensor network using grid search random forest with Boruta feature selection algorithm. *Journal of Communications and Networks*, 24(2), 264–273. DOI: 10.23919/jcn.2022.000002

[22] Panigrahi, R., Borah, S., Bhoi, A. K., Ijaz, M. F., Pramanik, M., Kumar, Y., & Jhaveri, R. H. (2021). A consolidated decision tree-based intrusion detection system for binary and multiclass imbalanced datasets. *Mathematics*, 9(7), 751 DOI: 10.3390/math9070751

[23] Rai, Kajal, M. Syamala Devi, and Ajay Guleria. "Decision tree based algorithm for intrusion detection." International Journal of Advanced Networking and Applications 7.4 (2016): 2828.

[24] Ngueajio, M. K., Washington, G., Rawat, D. B., & Ngueabou, Y. (2022, September). Intrusion detection systems using support vector machines on the KDDCUP'99 and NSL-KDD datasets: A comprehensive survey. In *Proceedings of SAI Intelligent Systems Conference* (pp. 609–629). Cham: Springer International Publishing. DOI: 10.1007/978-3-031-16078-3_42

[25] Mhamdi, L., McLernon, D., El-Moussa, F., Zaidi, S. A. R., Ghogho, M., & Tang, T. (2020, October). A deep learning approach combining autoencoder with one-class SVM for DDoS attack detection in SDNs. In *2020 IEEE Eighth International Conference on Communications and Networking (ComNet)* (pp. 1–6). IEEE. DOI: 10.1109/comnet47917.2020.9306073

[26] Ji, H., Kim, D., Shin, D., & Shin, D. (2018). A study on comparison of KDD CUP 99 and NSL-KDD using artificial neural network. In *Advances in Computer Science and Ubiquitous Computing: CSA-CUTE* 17 (pp. 452–457). Springer Singapore. DOI: 10.1007/978-981-10-7605-3_74

[27] Mohammed, A. J., Arif, M. H., & Ali, A. A. (2020). A multilayer perceptron artificial neural network approach for improving the accuracy of intrusion detection systems. *IAES International Journal of Artificial Intelligence,* 9(4), 609. DOI: 10.11591/ijai.v9.i4.pp609-615

[28] Devi, R. R., & Abualkibash, M. (2019). Intrusion Detection System Classification Using Different Machine Learning Algorithms on KDD- 99 and NSL-KDD Datasets - A Review Paper. *International Journal of Computer Science and Information Technology*. DOI: 10.2139/ssrn.3428211

[29] Muda, Z., Yassin, W., Sulaiman, M. N., & Udzir, N. I. (2011, July). Intrusion detection based on K-Means clustering and Naïve Bayes classification. In *2011 7th international conference on information technology in Asia* (pp. 1–6). IEEE. DOI: 10.1109/cita.2011.5999520

[30] Deshmukh, Datta H., Tushar Ghorpade, and Puja Padiya. "Intrusion detection system by improved preprocessing methods and Naïve Bayes classifier using NSL-KDD 99 Dataset." *2014 International Conference on Electronics and Communication Systems (ICECS)*. IEEE, 2014. DOI: 10.1109/ecs.2014.6892542

[31] Panda, M., & Patra, M. R. (2007). Network intrusion detection using Naive Bayes. International journal of computer science and network security, 7(12), 258–263.

[32] Li, L., Zhang, H., Peng, H., & Yang, Y. (2018). Nearest neighbors based density peaks approach to intrusion detection. *Chaos*, Solitons & Fractals, 110, 33–40. **DOI**: 10.1016/j.chaos.2018.03.010

[33] Belgrana, F. Z., Benamrane, N., Hamaida, M. A., Chaabani, A. M., & Taleb-Ahmed, A. (2021, January). Network intrusion detection system using neural network and condensed nearest neighbors with selection of NSL-KDD influencing features. In *2020 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)* (pp. 23–29). IEEE. **DOI**: 10.1109/iotais50849.2021.9359689

[34] Besharati, E., Naderan, M., & Namjoo, E. (2019). LR-HIDS: logistic regression host-based intrusion detection system for cloud environments. *Journal of Ambient Intelligence and Humanized Computing*, 10, 3669–3692. **DOI**: 10.1007/s12652-018-1093-8

[35] Wang, Y. (2005). A multinomial logistic regression modeling approach for anomaly intrusion detection. *Computers & Security*, 24(8), 662–674. **DOI**: 10.1016/j.cose.2005.05.003

[36] Wang, Y., Huang, Y., Wang, Q., Zhao, C., Zhang, Z., & Chen, J. (2023). Graph-based self-training for semi-supervised deep similarity learning. *Sensors*, 23(8), 3944. **DOI**: 10.3390/s23083944

[37] Mao, C. H., Lee, H. M., Parikh, D., Chen, T., & Huang, S. Y. (2009, March). Semi-supervised co-training and active learning based approach for multi-view intrusion detection. In *Proceedings of the 2009 ACM symposium on Applied Computing* (pp. 2042–2048). **DOI**: 10.1145/1529282.1529735

[38] Aung, Y. Y., & Min, M. M. (2018, June). Hybrid intrusion detection system using K-means and K-nearest neighbors algorithms. In *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)* (pp. 34–38). IEEE. **DOI**: 10.1109/icis.2018.8466537

[39] Horng, S. J., Su, M. Y., Chen, Y. H., Kao, T. W., Chen, R. J., Lai, J. L., & Perkasa, C. D. (2011). A novel intrusion detection system based on hierarchical clustering and support vector machines. *Expert systems with Applications*, 38(1), 306–313. **DOI**: 10.1016/j.eswa.2010.06.066

[40] Mirsky, Y., Doitshman, T., Elovici, Y., & Shabtai, A. (2018). Kitsune: an ensemble of autoencoders for online network intrusion detection. *Network and Distributed Systems Security (NDSS) Symposium* 2018, San Diego, CA, USA, **DOI**: 10.14722/ndss.2018.23204

[41] Choi, H., Kim, M., Lee, G., & Kim, W. (2019). Unsupervised learning approach for network intrusion detection system using autoencoders. *The Journal of Supercomputing*, 75, 5597–5621.

[42] Syarif, I., Prugel-Bennett, A., & Wills, G. (2012). Unsupervised clustering approach for network anomaly detection. In Networked Digital Technologies: 4th International Conference, NDT 2012, Dubai, UAE, April 24-26, 2012. *Proceedings*, Part I 4 (pp. 135–145). Springer Berlin Heidelberg. **DOI**: 10.1007/s11227-019-02805-w

[43] Rajadurai, H., & Gandhi, U. D. (2021). An empirical model in intrusion detection systems using principal component analysis and deep learning models. *Computational Intelligence*, 37(3), 1111–1124. **DOI**: 10.1111/coin.12342

[44] Benaddi, H., Ibrahimi, K., & Benslimane, A. (2018, October). Improving the intrusion detection system for NSL-KDD dataset based on PCA-fuzzy clustering-KNN. In *2018 6th International conference on wireless networks and mobile communications (WINCOM)* (pp. 1–6). IEEE. **DOI**: 10.1109/wincom.2018.8629718

[45] Alavizadeh, H., Alavizadeh, H., & Jang-Jaccard, J. (2022). Deep Q-learning based reinforcement learning approach for network intrusion detection. *Computers*, 11(3), 41. **DOI**: 10.3390/computers11030041

[46] Suwannalai, E., & Polprasert, C. (2020, November). Network intrusion detection systems using adversarial reinforcement learning with deep Q-network. In *2020 18th International Conference on ICT and Knowledge Engineering (ICT&KE)* (pp. 1–7). IEEE. **DOI**: 10.1109/ictke50349.2020.9289884

[47] Hubballi, N., & Suryanarayanan, V. (2014). False alarm minimization techniques in signature-based intrusion detection systems: A survey. *Computer Communications*, 49, 1–17. **DOI**: 10.1016/j.comcom.2014.04.012

[48] Jabez, J., & Muthukumar, B. J. P. C. S. (2015). Intrusion Detection System (IDS): Anomaly detection using outlier detection approach. *Procedia Computer Science*, 48, 338–346. **DOI**: 10.1016/j.procs.2015.04.191

[49] Kabir, E., Hu, J., Wang, H., & Zhuo, G. (2018). A novel statistical technique for intrusion detection systems. *Future Generation Computer Systems*, 79, 303–318. **DOI**: 10.1016/j.future.2017.01.029

[50] Moon, D., Im, H., Kim, I., & Park, J. H. (2017). DTB-IDS: an intrusion detection system based on decision tree using behavior analysis for preventing APT attacks. *The Journal of supercomputing*, 73, 2881–2895. **DOI**: 10.1007/s11227-015-1604-8

[51] Devarajan, Rajagopal, and Padmanabhan Rao." An efficient intrusion detection system by using behaviour profiling and statistical approach model." *Int. Arab J. Inf. Technol*. 18.1 (2021): 114–124. **DOI**: 10.34028/iajit/18/1/13

[52] Mohammadpour, L., Ling, T. C., Liew, C. S., & Aryanfar, A. (2022). A survey of CNN-based network intrusion detection. *Applied Sciences*, 12(16), 8162. **DOI**: 10.3390/app12168162

[53] Azizjon, M., Jumabek, A., & Kim, W. (2020, February). 1D CNN based network intrusion detection with normalization on imbalanced data. In *2020 international conference on artificial intelligence in information and communication (ICAIIC)* (pp. 218–224). IEEE. **DOI**: 10.1109/icaiic48513.2020.9064976

[54] Ding, Y., & Zhai, Y. (2018, December). Intrusion detection system for NSL-KDD dataset using convolutional neural networks. In *Proceedings of the 2018 2nd International conference on computer science and artificial intelligence* (pp. 81–85). **DOI**: 10.1145/3297156.3297230

[55] Vinayakumar, R., Soman, K. P., & Poornachandran, P. (2017, September). Applying convolutional neural network for network intrusion detection. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (pp. 1222–1228). IEEE. **DOI**: 10.1109/icacci.2017.8126009

[56] Yin, W., Kann, K., Yu, M., & Schütze, H. (2017). Comparative study of CNN and RNN for natural language processing. *arXiv preprint arXiv:1702.01923*.

[57] Sohi, Soroush M. et al. "RNNIDS: Enhancing network intrusion detection systems through deep learning." *Comput. Secur.* 102 (2020): 102151. **DOI**: 10.1016/j.cose.2020.102151

[58] Hossain, M. D., Inoue, H., Ochiai, H., Fall, D., & Kadobayashi, Y. (2020). LSTM-based intrusion detection system for in-vehicle can bus communications. *IEEE Access*, 8, 185 489–185 502. **DOI**: 10.1109/access.2020.3029307

[59] Albulayhi, K., Smadi, A. A., Sheldon, F. T., & Abercrombie, R. K. (2021). IoT intrusion detection taxonomy, reference architecture, and analyses. *Sensors,* 21(19), 6432. **DOI**: 10.3390/s21196432

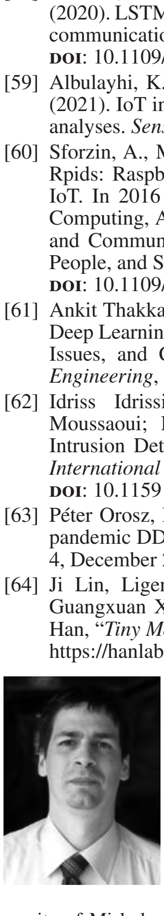
[60] Sforzin, A., Mármol, F. G., Conti, M., & Bohli, J. M. (2016, July). Rpids: Raspberry Pi IDS—a fruitful intrusion detection system for IoT. In 2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (pp. 440–448). IEEE. **DOI**: 10.1109/uic-atc-scalcom-cbdcom-iop-smartworld.2016.0080

[61] Ankit Thakkar; Ritika Lohiya; "A Review on Machine Learning and Deep Learning Perspectives of IDS for IoT: Recent Updates, Security Issues, and Challenges", *Archives Of Computational Methods In Engineering*, 2020. **DOI**: 10.1007/s11831-020-09496-0

[62] Idriss Idrissi; Mohammed Boukabous; Mostafa Azizi; Omar Moussaoui; Hakim El Fadili; "Toward A Deep Learning-based Intrusion Detection System for IoT Against Botnet Attacks", *IAES International Journal Of Artificial Intelligence*, 2021 **DOI**: 10.11591/ijai.v10.i1.pp110-120

[63] Péter Orosz, Balázs Nagy, Pál Varga, "Detection strategies for post-pandemic DDoS profiles", *Infocommunications Journal*, Vol. XV, No 4, December 2023, pp. 26–39., **DOI**: 10.36244/ICJ.2023.4.4

[64] Ji Lin, Ligeng Zhu, Wei-Ming Chen, Wei-Chen Wang, HanCai, Guangxuan Xiao, Haotian Tang, Shang Yang, Yujun Lin, and Song Han, "*Tiny Machine Learning Projects*" [Online]. Available: https://hanlab.mit.edu/projects/tinyml



Oliver Hornyak was born in Kazincbarcika, Hungary in 1973. He received the M.S. degree in Mechanical Engineering from the University of Miskolc, Hungary, in 1997, and the Ph.D. degree in Information Engineering from the University of Miskolc, Hungary, in 2002. From 2002 to 2006, he was an Associate Professor at the University of Miskolc, Hungary. From 2006 to 2012, he was a Senior Lecturer at the University of Miskolc. Since 2012, he has been an Associate Professor at the Department of Information Technology of the University of Miskolc. His research interests include network security, intrusion detection systems, data encryption, and cybersecurity applications. He is the author of about 100 articles and holds a patent.