Review

A Comprehensive Review: The Evolving Cat-and-Mouse Game in Network Intrusion Detection Systems Leveraging Machine Learning

Qutaiba Alasad [*] , Meaad Ahmed , Shahad Alahmed , Omer T. Khattab , Saba Alaa Abdulwahhab , Jiann-Shuin Yuan

Posted Date: 8 October 2025

doi: 10.20944/preprints202510.0564.v1

Keywords: Network security; network-based intrusion detection system; cybersecurity; adversarial machine learning attacks and defenses

Review

A Comprehensive Review: The Evolving Cat-and-Mouse Game in Network Intrusion Detection Systems Leveraging Machine Learning

Qutaiba Alasad [1,*], Meaad Ahmed [2], Shahad Alahmed [2], Omer T. Khattab [3], Saba Alaa Abdulwahhab [2] and Jiann-Shuin Yuan [4]

[1] Departments of Cybersecurity and Petroleum Systems Control Engineering, Tikrit University
[2] Department of Cybersecurity, Tikrit University
[3] College of Health and Medical Technologies-Al-Dour, Northern Technical University
[4] Department of Electrical and Computer Engineering, University of Central Florida, USA
* Correspondence: qutaibaeng@tu.edu.iq

Abstract

Machine learning (ML) techniques have significantly enhanced decision support systems to render the systems more accurate, efficient, and faster. ML classifiers in securing networks, on the other hand, face a disproportionate risk of mettle adversarial attacks as compared to other areas, including spam filtering and intrusion, and virus detections, and this introduces a continuous competition between malicious users and preventers. Attackers test ML models with inputs that have been specifically produced for evading these models and provide an inaccurate forecast. This paper presents a comprehensive review of attack and defensive techniques in ML-based NIDSs. It highlights the current serious challenges that the systems face in preserving robustness against adversarial attacks. Based on our analysis, with respect to their current superior performance and robustness, ML-based NIDS require rapid attention to develop more robust techniques to withstand such attacks. Finally, we discuss the current existing approaches in generating adversarial attacks and reveal the limitations of current defensive approaches. In this paper, the most recent advancements, such as hybrid defensive techniques that integrate multiple approaches to prevent adversarial attacks in NIDS, have effectively highlighted the ongoing challenges.

Keywords: Network security; network-based intrusion detection system; cybersecurity; adversarial machine learning attacks and defenses

1. Introduction

The increasing quantity, scale, and sophistication of network assaults have had a broad range of consequences for many individuals and victimized enterprises, ranging from large monetary expenditures to widespread power dissipations. The goal of these attacks is to illegally access to cloud flow or disrupt services offered to customers. Such attacks have significantly impact not only on the economic and financial affairs, but also on the national and cultural security [1–3]. Therefore, public and private sector institutes are anticipated to spend a very large amount of money to develop efficient techniques that can identify and eradicate breaches as a result of the significant negative impacts that such attacks generate [4]. Cyberattacks can cause serious damage to a country's money systems, economic stability, national security, and cultural protection. Because of these major risks, it is essential to block such attacks, whether they come from inside or outside the country, from either government or business organizations [5].

In computer servers and modern networks, firewalls and other rules-based security solutions have been commonly applied to prohibit active attacks. However, these solutions have not been able to completely detect malicious activities or contemporary threats [6]. As a result, Intrusion Detection

System (IDS) have been proposed for repeatedly observing the traffic and create alarm signals when any suspicious events take place. IDS analyzes the network by collecting sufficient data and information in order to detect unusual sensor node behavior, illegal access, improper usage, unauthorized users, hackers, cloaked malware, and many other risks [7].

IDS can broadly be categorized to: Network- and Host-based Intrusion Detection System (NIDS and HIDS). HIDS involves monitoring all the network ports and configuration parameters of the target device, which necessitates particular host-specific settings. Typically, HIDS, considered a passive technique [8], is implemented on single systems, and these single systems need a small program to monitor the operating systems and create alarms. NIDS, in contrast to the HIDS, keeps tracking all incoming and outgoing packets in a computer network, while HIDS only observe specific actions, such as which applications are allowed to employ and what files can be accessed [9]. This article reviews the current existing NIDS studies that focus on analyzing the flow of information between computers, such as network traffics. Accordingly, NIDS can essentially sniff the suspicious conduct and protect the network hosts from sophisticated attacks. Note that both forms of IDS use one of two ways to evaluate data, signature-based or anomaly-based, as shown in Figure 1. As of 2025, adversarial attacks have evolved, with new taxonomies from NIST emphasizing network-specific vulnerabilities and mitigations [10].

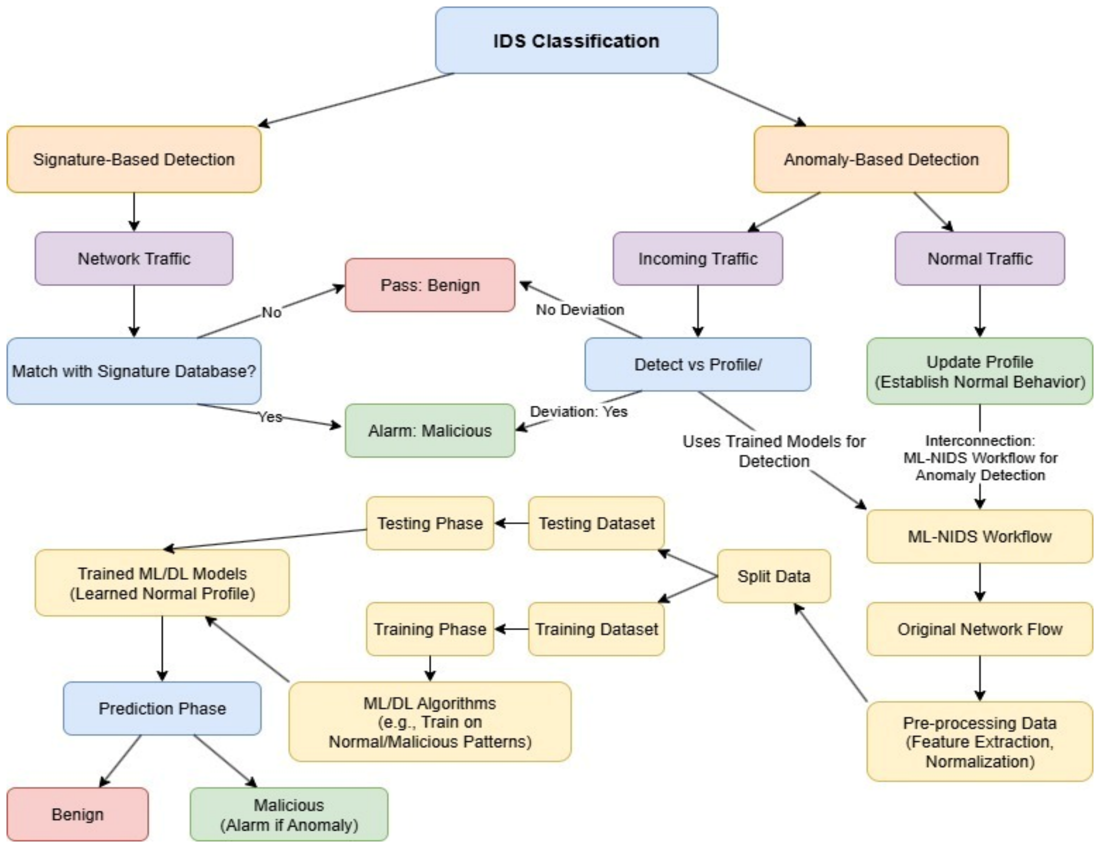
IDS Classification
Signature-Based Detection
Anomaly-Based Detection
Network Traffic
Pass: Benign
Incoming Traffic
Normal Traffic
No
No Deviation
Match with Signature Database?
Detect vs Profile/
Update Profile (Establish Normal Behavior)
Yes
Alarm: Malicious
Deviation: Yes
Uses Trained Models for Detection
Interconnection: ML-NIDS Workflow for Anomaly Detection
Testing Phase
Testing Dataset
ML-NIDS Workflow
Trained ML/DL Models (Learned Normal Profile)
Split Data
Training Phase
Training Dataset
Original Network Flow
Prediction Phase
ML/DL Algorithms (e.g., Train on Normal/Malicious Patterns)
Pre-processing Data (Feature Extraction, Normalization)
Benign
Malicious (Alarm if Anomaly)

Figure 1. Combination of IDS systems with Workflow of ML.

The signature-based IDS employs pre-determined and pre-configured attack signatures to discover malicious tasks and activities. It is appropriate for detecting predefined types of attacks but cannot be used to detect unexpected ones. As a result, an attacker can build special code to fool the system once attackers obtain access to the IDS. In contrast, the anomaly-based IDS techniques have been designed to recognize anomalous data patterns and are commonly employed to discover frauds, flaws, and invasions. More specifically, an anomaly-based IDS techniques observe user activity to determine what is legal and what is illegal through system logs. This method can be leveraged to prevent new types of attacks without any prior knowledge [6].

The NIDS serves as one of the best promising active security solutions against adversary attacks [11,12]. It is designed especially for network framework directors to detect and prevent unusual attacks in the network, in which important and sensitive data should be collected to protect the systems and prohibit any aberrant behaviors [13]. Since NIDS cannot prevent sophisticated attacks, especially in large networks, ML has been shown to effectively mitigate such attacks with low computational time. Figure 1 illustrates the combination of IDS with the workflow of ML.

1.1. Motivation

The increase in sophisticated adversarial attacks (AAs) on ML-based NIDS has motivated us to present this review paper. Since ML models have become more deeply incorporated to network security, they have also become more susceptible to adversarial manipulations. Given such sophisticated adversarial attacks, network security requires continuously arms race between assailants and defenders. 2025 studies show evasion rates up to 90% in IoT NIDS, as reviewed in a recent taxonomy of AAs on DL-based detection [14]. Adversaries are constantly developing their techniques to bypass detection systems, and therefore, researchers and scholars should always update and develop their approaches to prevent such attacks. More specifically, the main objective of this work is to offer an in-depth review on the current state of adversarial examples and defenses on ML-based NIDS. This work differs from the prior reviews in [15–28] as follows: We carefully review and analyze studies on attack and defense techniques, and we aim to reveal real current challenges in the existing works and provide possible future research. This paper also shows the necessity of leveraging real implemented attacks with modern datasets to successfully assess the ML-based NIDS. This article provides direct future research in this domain in detail in the discussion section.

1.2. Contributions of This Paper

The paper contributions are, as follows:

1. We give an in-depth review on existing ML-based NIDS attack and defense techniques, including a detailed overview of NIDS, its types, ML techniques, commonly used datasets in NIDS, attacker models, and possible detection methods.
2. The types of the powerful attack and defense techniques and the most realistic datasets in NIDS are carefully classified.
3. Existing attack and defensive approaches on adversarial learning in the NIDS are analyzed and evaluated.
4. We identify the real current challenges in detection and defensive techniques that need to be further investigated and remedied in order to produce more robust NIDS models and realistic datasets. Revealing current challenges guides us to provide insights about future research directions in the area of detecting AAs in ML-based NIDS. This has been explored in detail in the discussion section, highlighting our findings in revealing the limitations of current existing approaches.
5. A comprehensive analysis has been conducted to show the impact of adversarial breaches on ML-based NIDS, including black-, gray-, and white-box attacks.

2. Background

2.1. ML Techniques in NIDS

ML approaches have been utilized to recognize the valuable features in network traffic and detect zero-day attacks, e.g., new obscure threats, that are difficult to detect leveraging traditional signature-based methods. ML-based NIDS improves the generalization of the detection techniques to detect more sophisticated attacks [13]. Moreover, it has been shown that ML-based NIDS can significantly mitigate attacks targeted to bypass or fool the system [4]. ML algorithm is mainly classified into three types: Reinforcement Learning, Unsupervised Learning, and Supervised Learning [29,30].

Given the widely used of Artificial Intelligent (AI) in the network security domain, revealing the main weaknesses of ML classifiers in adversarial methods is essential [31]. In cyber defense, adversarial machine learning (AML) attacks pose significant challenges to the ML security [13]. These assaults exploit the intrinsic sensitivity in ML models to their internal parameters in order to create strong infected examples that impact the model to achieve the attackers' goal. Note that one of the strongest attacks that impacts the ML models is the adversarial perturbations [32]. In order to develop models and execute classification tasks, machine learning algorithms need sufficiently large datasets. These learned models can produce wrong decisions with high confidence due to skillfully created and managed perturbations introduced at the genuine inputs, called Adversarial Examples (AEs) [13].

2.2. Adversarial Machine Learning (AML)

AML requires incorporating perturbations to the given data input to deceive the ML model, and this in turn leads to providing inaccurate detection outcomes that are desired by the assaulter. The incorporated malicious data must be undetected by observers. In contrast, revealing this in other domains is easier. For instance, in computer vision, modifying only a few pixels can result in generating unexpected outcomes. However, detecting such threats in network traffic is more challenging due to the unique features in this domain [33]. To be more specific, AML can sit at the crossroads of ML and computer security, and it is usually a battle between two agents [34]. The first agent is a malicious payload intruder whose goal is to infiltrate a specific network, while the second agent's aim is to protect the network from risks caused by the payload [29]. Table 1 illustrates the factors that need to be considered with any attack, including descriptions, NIDS-specific examples, and relevant references. These factors primarily involve knowledge, timing, goals, and capability.

Table 1. Summary of Attacker Model Factors in Adversarial Machine Learning.

Description	Description	Examples in NIDS Context	Refs.
Knowledge	Level of knowledge on the attacked model that the assaulter knows.	Black-box: Only input-output access. Gray-box: Limited dataset access. White-box: Full access to model parameters.	[13]
Timing	When the attack occurs in the ML lifecycle.	Evasion: During the testing, to misclassify. Poisoning: During the training, to corrupt data.	[14,29,35]
Goals	Attacker's objective.	Targeted: Specific misclassification. Non-targeted: General error induction.	[36]
Capabilities	Attacker's access and actions on the system.	Full access: Modify model internals. Limited: Query outputs only.	[10,15–26,32]

2.2.1. Knowledge

AML threats are broadly partitioned into: black-, grey-, and white-box [13]:

- **Black box attack**: no information about the parameters or the classifier's structure is required for the malicious agents. They should be able to access the input and the output of the models, and the rest is considered as black-box.

- **Gray box attack**: limited information or access to the system should be known to an attacker, such as accessing the preparation of the dataset and the predicated labels (training dataset), or applied a limited number of queries to the model.
- **White box attack**: In this type, it has been assumed that the attacker has completely knowledge and in-formation about the classifier and the used hyperparameters.

2.2.2. Timing

Timing plays an excellent role in modeling AAs. A recent study showed that classification evasion is one of the most prevalent in NIDS, providing a taxonomy of DL-based adversarial attacks [14]. Generally, there are two types of attacks, called evasion and poisoning attacks, on NIDS-based ML. The details of these two attacks are demonstrated in Figure 2.

- **Evasion Attack:** The attack can be employed during the testing phase, where an attacker attempts to force the ML model to classify the observations wrongly. In the context of network-based IDS, the attacker tries to prevent the detection system from detecting malicious or unusual events, and therefore, the NIDS wrongly classifies the behavior as benign. To be more specific, four scenarios might occur, as follows: (1) Confidence reduction, which reduces the certainty score to cause wrongly classification, (2) Misclassification, in which the attacker modifies the result to produce a class not similar to the first class; (3) Intended wrong prediction, in which the malicious agent generates an instance that fools the model into classifying the behave as an objective or incorrect class, and (4) source or target Misclassification, in which the adversary changes the result class of a certain attack example to a particular target class [35].
- **Poisoning attack:** It occurs during the training phase, in which an assaulter tampers with the model or the training data to yield inaccurate predictions. Poisoning attacks mainly involve data insertions or injections, logic corruption, and data poisoning or manipulation. Data insertion or injection occurs when an assailant inserts hostile or harmful data inputs into the original data without changing its characteristics and labels. The phrase "data manipulation" refers to an adversary adjusting the original training data to compromise the ML model. Logic corruption refers to an adversary's attempt to modify the internal model structure, decision making logic, or hyperparameters, to malfunction the ML [29].

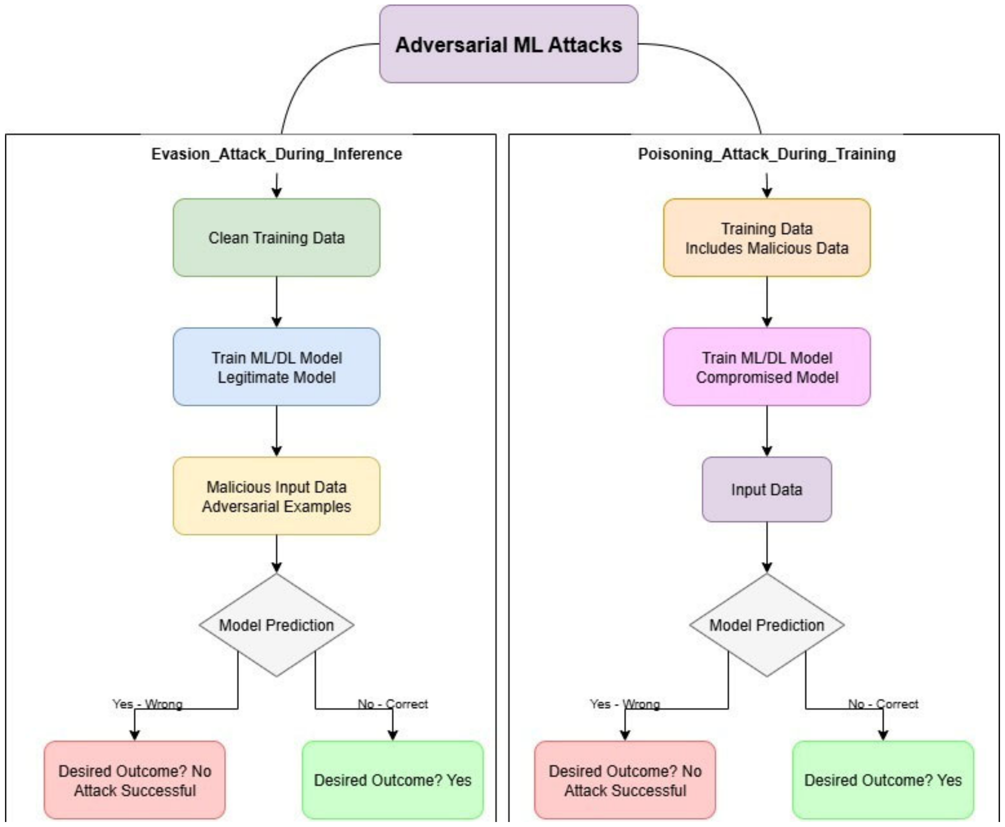
Adversarial ML Attacks
Evasion_Attack_During_Inference
Clean Training Data
Train ML/DL Model Legitimate Model
Malicious Input Data Adversarial Examples
Model Prediction
Yes - Wrong
No - Correct
Desired Outcome? No Attack Successful
Desired Outcome? Yes
Poisoning_Attack_During_Training
Training Data Includes Malicious Data
Train ML/DL Model Compromised Model
Input Data
Model Prediction
Yes - Wrong
No - Correct
Desired Outcome? No Attack Successful
Desired Outcome? Yes

Figure 2. The flow of the evasion and poisoning Attacks.

2.2.3. Goals

Attackers can use a specific algorithm for different reasons. However, in most cases, they either have a specific purpose in mind and require the algorithm to produce a specific result (a targeted attack), or they want to lower the trustworthiness of the algorithm via inducing mistakes (an untargeted attack) [36].

2.2.4. Capability

Refers to the activities an assailant may perform on the target system, including the AI system. The level of access should be shown to the assailant with respect to the digital finder: Limited access (can only access and read the output result), full access (can read and adjust the internal structure of the model and its outputs), or no access to any part [32].

With the increasing interest in leveraging ML techniques in the area of network security, many hostile or malicious attacks against these technologies have become prevalent. This highlights the need for extensive research solutions and mitigation techniques, as well as efforts to review papers in the field to address these assaults. AML has been presented in many publications in the image and text recognition fields. However, comprehensive studies are still needed to carefully address the issues of adversarial examples in the NIDS area. Even though there are many recent studies, surveys, and reviews in the area of NIDS, there are still many challenges that have not been explored yet. With respect to the efforts presented [15–19,21–25,27], this paper acts as an inspiration to provide a review on antagonistic assaults and their countermeasures in detail that can effectively reveal current challenges in literature studies and help set up future research directions. The main purpose of this survey is to audit and assess advancements in adversarial learning applied to the NIDS space and highlight the key challenges and areas that require further examination in order to achieve worthy improvements in the area of ML-based NIDS. AML exploits the ML sensitivities and vulnerabilities.

According to the 2025 NIST taxonomy, such attacks are categorized based on the lifecycle stages and attacker goals [10].

2.3. Adversarial Attacks Based on Machine Learning

The commonly utilized and adapted techniques in cybersecurity are provided in this section. These techniques serve a critical role in dealing with both defensive and offensive facets of managing systems and shielding sensitive data.

2.3.1. Generative Adversarial Networks (GANs)

They, presented by Goodfellow et al. in 2014, are a type of neural network model widely employed for creative tasks. They are one of the most promising tools in many applications, such as unsupervised learning, image creation, earthquake engineering, and data augmentation [37,38]. A discriminator (D) and a generator (G) are two components in any GAN model. G creates examples that accurately imitate the original ones, which exist in the main traffic [26,39]. When the adversarial learning process is utilized to train the network, the G imitates the data obtained from noisy random input, and the outcomes are bypass to D model in order to assess the generated samples and output the probability to find if the data was produced via original primary dataset or G. To maximize the probability value, G is retrained using the results generated by D. The G and D models function in a two-player, zero-sum, mini-max game, while the G model maximizes the probability value, the discriminator reduces the value [40]. Compared to ZOO, GANs: GANs are more effective in black-box NIDS, with 2025 reviews highlighting their impact on evasion success in network traffic [14].

The loss of the G is computed via the following equation (1):

$$L_G = E_{M \in S_{attack} N} D\big(G(M,N)\big) \qquad (1)$$

Where the n-dimension and m-dimension are represented by N (noise vector) and M (attack sample vector). D refers to the discriminator, and G refers to the generator. L_G is the creator loss, and S attack is the attack instance. E refers to the value used in all of the G's random inputs. Note that, it is very essential to decrease the value of L_G to elevate the accuracy of the G. Two processes are required to train the D model: i) adversarial data created via G, and ii) labels estimated by IDS. To calculate the loss function in the D, the following equation can be used:

$$L_D = E_{s \in B_{benign}} D(s) - E_{s \in B_{attack}} D(s) \qquad (2)$$

Where E is the approximated size of the created data, classified as malicious or benign. s is the data collection generated from the G model and utilized to train the D model. The benign sample is represented by the B benign, while the attack sample is represented by the B attack [41].

2.3.2. Zeroth-Order Optimization (ZOO)

ZOO, also known as bandit or gradient-free optimization, is used to estimate gradients of functions that are difficult to calculate with traditional methods. Instead of relying on direct gradient information, ZOO approximates gradients by evaluating the function itself, making it useful for optimizing complex problems [1]. Interestingly, ZOO can handle many challenging tasks where gradients are hard to calculate, e.g., attacking large neural networks, network control, reinforcement learning, and tuning deep neural network (DNN) hyperparameters [42].

This technique also approximates the function gradient through checking the function at some points and seeing how the output modifies. Additionally, it works well with black-box optimization methods, where the model's hyperparameters cannot be accessed [43]. ZOO can create AEs to attack the NNs in black-box-settings and make the NN models produce incorrect predictions [44], although the classifier structure is inaccessible by an attacker [42,43]. The optimization problem of ZOO can be calculated via the following equations (3) and (4):

$$\min_{x'} \; ||x' - x||_2 + c.f(x',t) \quad s.t. \quad x' \in [0,1]^p \qquad (3)$$

In which c refers to a regularization parameter when it is larger than 0. p refers to a dimensional column vector. X represents the original data input associated with the label l, X' is to the malicious example connected to the label t, e.g., $f\ (X') = t \neq f(\mathrm{x})\ l, where$ the loss function is given by $f(X' + t)$ and can be computed using equation (4):

$$f(X', t) = max\ \left\{\max_{l \neq t} log[F(X')]_l - log[F(X')]_{t'} - K\right\} \qquad (4)$$

Where the class' number is K, and when k represents larger than or equal to zero, it represents a tuning parameter to improve the transferability of the attack. $\hat{\mathrm{g}}_\mathrm{i}$, represents the estimated gradients and can be calculated using the finite differences approaches as follows (5):

$$\hat{\mathrm{g}}_i \coloneqq \frac{\partial\ \mathrm{f(x)}}{\partial\ \mathrm{x_i}} \approx \frac{\mathrm{f(x + he_i) - f(x - he_i)}}{\mathrm{2h}} \qquad (5)$$

Where the i-th part of the standard vector is referred to as $\mathrm{e_i}$, and h represents a small value. ZOO employs the Newton's technique with the Hessian approximation, $\hat{\mathrm{h}}\mathrm{i}$, as given in the following equation.

$$\hat{\mathrm{h}}_i\!:= \frac{\partial^2\ f(x)}{\partial\ x_{ii}^2} \approx \frac{f(x + he_i) - 2f(x - he_i)}{h^2} \qquad (6)$$

This approach is effective in approximating the gradient and Hessian while maintaining approximately the same performance in the C&W breach, and no training for the models or further knowledge about the target model is required. However, it requires additional significant computational time and costs, which may enable the detection mode to expose the attack in real applications since a large query number to the model is required [42].

2.3.3. Kernel Density Estimation (KDE)

KDE has been considered as a powerful statistical approach used to predict directly from observed data the probability density function (PDF) of a random variable. In contrast to parametric methods that require specific distributional assumptions, KDE does not make any assumptions about the data's distribution. This makes KDE especially useful for analyzing complex or unknown data patterns, helping to accurately capture the behavior of real-world datasets [45].

KDE smooths data points by placing a kernel function (like a symmetric, non-negative Gaussian) over each point, spreading its influence over a range. This creates a smooth density estimate that reveals the data's true distribution [46]. KDE uses this kernel to find the distribution of a variable without needing prior assumptions, making it a non-parametric method. The density is estimated purely from the observed data using the following equation:

$$f_h(x) = \frac{1}{n}\sum\nolimits_{i=1}^{n} k_h\ (x - \ x_i) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - x_i}{h}) \qquad (7)$$

In the equation, h represents the bandwidth (controlling smoothness), x is the input variable, and the kernel function is given by K, which is bounded, non-negative, and integrates to unity (normalized) [47]. Due to the KDE's flexibility, it is used in many areas, like seismology for analyzing image processing, the hybrid Kalman filter (EnKF), and microearthquake (MEQ) data [45]. In cybersecurity, this technique works with SVDD based on distance (Support Vector Data Description) to spot traffic adversaries, boosting performance, detection accuracy, and outlier identification—especially on the NSL-KDD dataset [48].

KDE is powerful because it can reveal data distributions, especially when they have complex shapes like multiple peaks that parametric methods struggle with. For instance, when applied to the National Alzheimer's Coordinating Center (NACC) Dataset, KDE successfully uncovered hidden patterns, showing multiple peaks in cognitive test score distributions [49]. However, KDE's performance depends heavily on the bandwidth, a smoothing parameter that controls how much detail is preserved. A large bandwidth oversmooths the data, hiding important features like multiple peaks, while a small bandwidth creates a noisy estimate that doesn't reflect the true distribution [50]. That's why choosing the right bandwidth is crucial for getting an accurate density estimate.

Recently, KDE has been combined with advanced techniques to assess changing dynamic geothermal system. It is used to approximate the probability value of such a system and compute its probability density function (PDF) for different situations, like thermal breakthrough limits and how long reservoirs might last. This approach gives a fuller picture of geothermal resources than traditional static methods [45]. Because of this, KDE is a powerful non-parametric tool for finding PDFs in datasets, making it useful for many applications. These include analyzing risks in renewable energy systems, spotting patterns in medical data, and evaluating geothermal resources. However, to get precise density estimates, it is important to carefully select both the kernel type and bandwidth [51].

2.3.4. DeepFool

DeepFool is an effective way to create adversarial examples that fool deep neural networks (DNNs). It works in white-box assaults; assaulters can completely bypass the classifier's design. Unlike targeted attacks, it simply looks for the smallest possible change to make the model misclassify input data by studying where the model switches between different classes [52]. While DNNs work very well in tasks like language processing, speech recognition, and image analysis [53], they can be tricked by tiny, hard-to-notice changes in input data that lead to wrong outputs. This weakness is especially dangerous in important areas like self-driving cars and security systems [54]. In cybersecurity, DeepFool can create "poisoned" data that fools the model but doesn't break it, using techniques like Pearson's correlation to filter out useless features - known as a poisoning attack [55]. Via studying how the model works and slowly changing the primary input, DeepFool iteratively applies minimal perturbations until the decision boundary is crossed to efficiently generate small but highly impactful adversarial changes [56]. Recent 2025 attacks adapt DeepFool for DL-NIDS, as categorized in a taxonomy of deep learning adversarial methods [14].

DeepFool begins with a zero perturbation vector. Each step calculates how close the input is to the classification boundary to force misclassification. It finds the smallest required modification to cross the decision threshold by approximating the model near the given input. DeepFool is used to keep finding tiny adjustments that change the model's prediction. The changes depend on gradient differences and perturbation direction—basically, DeepFool measures the input's distance to the decision threshold to add the minimum noise that fools the model [25]. It searches the input space for the nearest boundary and the shortest crossing path. A small overshoot is added to ensure the input fully crosses the boundary. The process repeats—adding tiny perturbations and checking results—until the model makes a mistake [25]. The DeepFool model first finds the minimum change required to reach the classification boundary. Afterwards, starting from the last obtained position, it repeats the process over and over until it successfully creates adversarial examples. Mathematically, the smallest required change to make an adversarial sample is shown in equation (8).

$$\delta(X|f) = \min_{r} ||r||_2 \quad s.t. \quad f(X+r) \neq f(X) \qquad (8)$$

Where the smallest needed change is given by r, while δ measures how resistant the linear modeler f is to the input X. The model works as $f(x) = W^T \cdot x + b$, where the bias and the weight are represented by b and W, respectively. The DFA technique provides an accurate and efficient way to test how strong ML models are against attacks. It creates adversarial examples with smaller changes than the FGSM and JSMA methods but tricks models more often. However, it requires more computing power than both [52].

2.3.5. Fast Gradient Sign Method (FGSM)

FGSM is an effective technique to produce infected instances [57]. These malicious samples are specially modified inputs where each pixel is changed enough to render the ML classifiers produce incorrect outputs. The method works by calculating how modifications on the input impact the model's error (the gradient), then adjusting each pixel in the direction that maximizes this error. The amount of change is controlled by a setting called epsilon (ε), which limits the change on the given input. The final changed image (the AE) can be calculated using Equation (9).

$$X_{adversial} = X + \varepsilon . Sign\left((\nabla x)(\theta, X, Y)\right) \qquad (9)$$

Where ε is a minimum fixed value, and ∇ indicates the loss function gradient J (how steep the slope is), the model's parameters, original input, and true label are θ, X, and Y, respectively. This method does its work in three stages: (1) first, it finds the loss's gradient, (2) scales it to a maximum size of ε, and (3) adds it to X to create the adversarial example $X_a{}^d{}_v$. While FGSM is fast at generating AEs, it's less effective than advanced methods because it produces only one AE per input and doesn't explore all possible attacks. Another limitation is that it is a white-box assault, restricting its use when attackers have limited access. However, it is still helpful for manufacturers to test their models' robustness or in insider attacks [25]

2.3.6. The Carlini and Wagner (C&W) Attack

In this method, the optimization techniques are employed to generate better assaults than the older L-BFGS approach. It works by [58]. Carlini et al. modified the objective function and removed L-BFGS's box constraints. They tested three attack types using different distance measures (L_0, L_2, and L_∞) and replaced cross-entropy loss with hinge loss. A key innovation was introducing variable k instead of directly optimizing perturbation δ, avoiding box constraints. The approach is mathematically defined in Equations (10) and (11):

$$\min_{\delta}(X, X+\delta) + c.f\,(X+\delta) \quad s.t. \quad X+\delta \in [0,1] \qquad (10)$$

$$X+\delta = \frac{1}{2}[\tanh(k)+1] \qquad (11)$$

The C&W attack uses a carefully chosen constant c > 0, where δ represents the malicious input, D(·,·) measures distances (L_0, L_2, or L_∞), and the loss function is given by f(X+δ) that succeeds when the model predicts the attack's target. As shown in Equation (8), k replaces δ as the optimized variable. Through a white-box attack, C&W can transfer between networks, enabling black-box attacks against ML security systems with partial knowledge. While more effective than L-BFGS at bypassing advanced defenses, such as distillation-based defense and adversarial training, it requires more computing power than FGSM or JSMA [24].

2.3.7. The Jacobian-based Saliency Map Attack (JSMA)

The JSMA was introduced by Papernot et al. [59] that fools DNNs. It works by utilizing the Jacobian matrix to identify which input features most affect the model's predictions, then modifies those features to fool the model. Unlike FGSM, which modifies all pixels slightly, JSMA changes only a few key pixels to create adversarial examples. It builds a saliency map from network gradients to identify which pixels will most effectively trick the model, then modifies them one by one. The process repeats until either the attack succeeds or reaches the maximum allowed pixel changes. For an input image X with label l (f(X)=l), the attack adds a small change δ to create X' that gets misclassified (f(X')=l*), as follows:

$$Arg\ min_{\delta_x} \|\delta_x\| \qquad s.t.\ f(X') = f(X+\delta_x) = l' \qquad (12)$$

For the input X, the Jacobian matrix and its positive derivatives are given in formula (13):

$$J_f(X) = \frac{\partial f(X)}{\partial X} = \left[\frac{\delta f_j(X)}{\delta X_i}\right]_{i\epsilon 1\ldots M; j\epsilon 1\ldots N} \qquad (13)$$

Compared to FGSM, JSMA requires more computing time because it calculates saliency values. However, it changes fewer features in the input, creating adversarial examples that look much more like the original samples [24].

2.3.8. Projected Gradient Descent (PGD)

PGD builds on BIM by adding a projection step to stay within allowed limits. Developed by Madry et al. (2018), this optimization method finds the minimum of a function while following rules like maximum perturbation size. It works by repeatedly moving in the direction that reduces the function most (following the negative gradient), then adjusting to stay within the permitted area. This projection step guarantees the solution always obeys the constraints. The method is summarized in Expression (14).

$$\prod C_\varepsilon(X') = Argmin_{z \in C_\varepsilon}\|Z - X'\|, \qquad X_{N+1}^{adv} = \prod_{C_\varepsilon}\left\{X_N^{adv} + \alpha.Sign\left(\nabla x\, J\,(X_N^{adv}, Y)\right)\right\} \qquad (14)$$

Here, Cε refers to the allowed set of changes, and $C_ε$ is equal to $\{z: d(x,z) < \varepsilon\}$, ∏_$C_ε$ represents projecting back into $C_ε$, and α controls the size of the step. For instance, when using the L∞ $d(x,z) = ||x - z||\infty$), the projection $\prod_{C_\varepsilon}(z)$ simply clips z to $[x-\varepsilon, x+\varepsilon]$. Y and J are the correct labels and the loss function, respectively. N counts iterations, X is a given image, and α controls the change in the size. PGD guarantees solutions stay within allowed limits. This is very good for applications with limited constraints, but the step of the projection is slow, especially in complicated constraints [24,60].

2.3.9. Basic Iteration Method (BIM)

It, introduced in 2017, improves on FGSM by using multiple small steps instead of one big change [61]. It repeatedly applies FGSM to an input, making tiny adjustments in the direction that most increases the classifier's error. The created adversarial example looks almost like the original but fools the model. BIM starts with an initial solution and gradually improves it using gradient descent. After each step, the modified image is adjusted to keep pixel changes within limits. The method is summarized in Expression (15):

$$X_0^{adv} = X, \quad X_{N+1}^{adv} = Clip_{X+\varepsilon}\left\{X_N^{adv} + \alpha.Sign\left(\nabla x\, J\,(X_N^{adv}, Y)\right)\right\} \qquad (15)$$

Where the loss function is given by J, Y represents its true label, X refers to the given image, N counts iterations, and α controls how much we change the input. The Clip function keeps the adversarial example within two limits: the ε range (x±ε) and valid input values. BIM starts with the original image, calculates how changes affect the loss (gradient), then makes small, controlled adjustments in that direction. After each change, it clips the values to stay within bounds. This process repeats either until successful or for N iterations [24]. Table 2 shows a comparison of adversarial attack techniques in NIDS, including threat type, strengths, weaknesses, tested datasets, references, and success rates from examples.

Table 2. Comparison of Adversarial Attack Techniques.

Attack Technique	Type (White/Black /Gray)	Strengths	Weaknesses	Success Rate in NIDS (Examples)	Datasets Tested	Refs.
GANs	Black/White /Gray	High evasion; realistic samples	Computationally intensive	98% evasion [41]	CICIDS2017	[14,26,37–41]
ZOO	Black	Gradient-free; black-box effective	High query count; slow	97% on DNNs [42]	NSL-KDD	[1,42,43,44]
KDE	White/Black	Non-parametric density estimation	Bandwidth sensitivity	95% outlier detection [48]	NSL-KDD	[45–51]
DeepFool	Black/White	Minimal perturbations	Compute-heavy; white-box only	90% misclassification [52]	KDDCup99	[14,26,52 -56]
FGSM	White	Fast generation	Less transferable	97% on CNN [57]	KDDCup99	[25,57]
C&W	White	Optimized for distances (L0/L2/L∞)	Resource-intensive	95% bypass [58]	Various	[24,58]
JSMA	White	Targets key features	Slow; feature-specific	92% targeted [59]	NSL-KDD	[24,59]

PGD	White	Constrained optimization	Iterative; compute-costly	96% robust test [60]	CICIDS2017	[24,60]
BIM	White	Multi-step improvements	Similar to PGD but basic	94% evasion [61]	CICIDS2017	[24,61]

Given the comparative analysis of the adversarial defensive and attack strategies, the evaluation between the GAN and DeepFool can be classified, as follows:

In terms of attacks, GANs generate more realistic infected samples, which can effectively bypass NIDSs but require hours to days for training purposes, and this, in turn, makes them impractical and infeasible in many applications. DeepFool carefully calculates minimal perturbations using mathematical operations, and based on our recent study analysis supported by experimental results [62], it has been shown that precise training of the DeepFool technique produces strong samples, which could be stronger than the ones in GAN attacks. Note that generating such samples can be done in seconds with high stealth advantages.

In terms of defenses, while GANs can be utilized to create unpredictable attack vectors, their randomness prevents systematic defense development by making the comprehensive countermeasures hard to be generated. DeepFool's deterministic approach allows defenders to understand the attack techniques accurately. In fact, this enables the targeted countermeasure and rapid security testing cycles. Further, the consistency in the DeepFool ensures that defensive approaches remain reliable on different samples and instances, unlike GANs, where different patterns can deceive the selective defenses. Therefore, this makes DeepFool outperform GAN.

We focused on comparing GANs and DeepFool since they are currently the two most effective adversarial techniques. GANs are known for generating highly evasive samples, while DeepFool excels in efficiency and precision. Our recent findings demonstrate that, with carefully and properly training, DeepFool can outperform the GANs while offering significant advantages in terms of the speed and practice.

3. Datasets used in NIDS

3.1. Type of Used Datasets

This subsection provides in detail the most commonly employed datasets in the cybersecurity field of NIDS. We also briefly mention other, less frequently utilized datasets in Table 3. 2025 critiques highlight KDD99's obsolescence; recommend LITNET-2020 updates, as per a taxonomy reviewing adversarial attacks on NIDS [14].

- **NSL-KDD dataset:** it represents the updated KDDCup 99 dataset implemented to solve the problems of the data imbalance and irrelevant records between abnormal and normal samples. It does have 41 features: 13 for content connection features, 9 for temporal features (found during the two-second time window), 9 for individual TCP connection features, and 10 for other general features [63].
- **UNSW-NB15 dataset:** It has been generated in 2015 by the cybersecurity research group at the Australian Centre of Cyber Security (ACCS). This dataset includes about data with 100 GB size, including both malicious and normal traffic records, where each sample has 49 features created by using many feature extraction techniques. It is mainly partitioned into test and training sets, involving, respectively, 82.332 and 175.341 instances [64].
- **BoT-IoT dataset:** It has been presented at the UNSW Canberra Cyber Center by the Cyber Range Lab. This dataset has been generated from a physical network incorporating botnet and normal samples from traffic. It consists the serious attacks, as follows:

  -DoS—This attack relies on the HTTP, TCP, and UDP protocols.

  -DDoS—Distributed DoS relies also on the HTTP, TCP, and UDP protocols.

-Information theft: it is for data theft and keylogging.

-Information gathering uses for OS fingerprinting and service scanning [65].

- **Kyoto 2006+ dataset:** It was established and revealed by Song et al. [66]. It is a collection of real traffic coming from 32 honeypots with various features from almost three years (Nov 2006 to Aug 2009), including a total of over 93 million samples [67]. Then, it has been expanded by adding more honeypots, for nine years of traffic (to 2015), to include DNS servers for creating benign traffic records. Each of these records has 24 features captured from network traffic flows, 14 of them are represented as KDD or DARPA, while the rest are added as new features.
- **CIC dataset:** It has been generated to assess the NIDS performance and train the models further. The first generation of these datasets was CICIDS2017. It was collected from a synthetic network traffic for 5 days, available on request (is protected). The collected data are in flow and packet formats with 80 different features [4,20].
- **CSE-CIC-IDS2018 dataset:** The Canadian Institute for Cybersecurity (CIC) and Communications Security Establishment (CSE) have implemented this on AWS (Amazon Web Services), based in Fredericton [68]. It represents the extended CIC-IDS2017 and was collected to reflect real cyber threats, including eighty features, network traffic, and system logs [69,70]. The network's infrastructure contains 50 attacking computers, 420 compromised machines, and 30 servers.
- **CIC-IDS2019 dataset:** This one includes new attacks and addresses some flaws discovered in the previous ones: CIC-IDS2018 and CIC-IDS2017 [71,72]. It contains 87 features classified into normal and abnormal ones, with numerous samples for DDoS-based attacks.
- **The ADFA-LD dataset:** It has been generated via the Australian Defence Force Academy (ADFA) at the New South Wales University, the Australian Centre for Cyber Security (ACCS). It was derived from the host-based IDSs and includes samples recorded from two operating systems. Many important attacks in this dataset have been collected from zero-day and malware-based attacks [73].
- **KDD CUP 99 dataset:** It has been produced in 1998 via the processing of the tcpdump segment of DARPA. It contains a total of 23 attacks with 41 features and an additional one, the 42nd feature, employed to divide the connection as either normal or attack [74].
- **CTU-13 dataset:** it is a synthetic dataset that concentrates mainly on botnet network traffic and contains a wide range of dataset analyses for anomaly detection on critical infrastructures, covering 153 attack types, including port scanning and DDoS. The dataset has 30 samples. In each one, a different botnet malware is executed in a controlled environment. This dataset was originally presented in two formats: a full Pcap of normal and malicious packets and a labelled bidirectional Netflow [75]. Table 3 illustrates the advantages and disadvantages of commonly used datasets in NIDS, highlighting strengths, limitations, and supporting references to evaluate the adversarial contexts.

Table 3. Pros and Cons of Commonly Used Datasets in NIDS.

Dataset	Pros	Cons	Refs.
NSL-KDD	Addresses KDD99 imbalances; 41 features for efficient processing.	Outdated (pre-2010); lacks modern attacks like IoT-specific threats.	[14,63]
UNSW-NB15	Realistic traffic; 49 features including raw data.	Large size (~100 GB); requires heavy preprocessing.	[14,64]
BoT-IoT	Includes botnet and IoT attacks; real network simulation.	Limited to specific protocols (HTTP, TCP, UDP); small sample size.	[65]

Kyoto 2006+	Long-term data (3+ years); honeypot-based with diverse features.	Older traffic (up to 2015); may not reflect 2025 threats.	[66,67]
CICIDS2017	Modern attacks; 80 features in flow/packet formats.	Synthetic; potential biases in simulation.	[4,20]
CSE-CIC-IDS2018	Updated with AWS simulation; 80 features, real cyberattacks.	Resource-intensive; focuses on DDoS/volumetric attacks.	[68]
CIC-IDS2019	Fixes prior flaws; 87 features with new DDOS attacks.	Large volume; class imbalance in some categories.	[71,72]
ADFA-LD	Host-based with zero-day/malware; Windows/Linux samples.	Limited to host IDS; not network-flow focused.	[73]
KDD CUP 99	Classic benchmark; 41 features with labeled attacks.	Highly outdated (1998); redundant/irrelevant records.	[74]
CTU-13	Botnet-focused; 153 attack types in Pcap/Netflow.	Synthetic; limited to botnets, not broad threats.	[75]

Given the current datasets used in NIDS, the effectiveness of the adversarial defense and attack mainly depends on the training data volume and diversity since, as known, the larger and more complex datasets are important to generate robust adversarial sample and further improve the generalization on different network systems, as follows. For example, the CICIDS2019 represents the most comprehensive dataset compared to others listed ones, since it incorporates extensive contemporary attack types, e.g., DDoS variants, web-based threats, and advanced persistent threats, which reflect modern cybersecurity landscapes. This can help to make the model optimal in DeepFool's boundary approach through obtaining valuable feature spaces and providing precise decision boundary identification. Therefore, it is excellent for training the classifier.

3.2. Data Preparation

A. Data Preprocessing: It involves the following:

- Handle missing values:
  - Remove rows and/or columns with many missing values.
  - Assign with median and / or mean (for numerical features).
  - Constant value or use mode (for classifying features).
  - Remove irrelevant data: Remove unnecessary columns, e.g., timestamps if not relevant.
- Eliminates duplication: Remove duplicated rows to prevent redundancy.

B. Data Transformation

- Log transformation: Apply to skewed numerical features to make them more normal-like.
- Binary features: Convert certain columns, e.g., flags, attack types, into binary 0/1 variables for classification tasks.

C. Handling Anomalies

- Noise filtering: Remove improbable or corrupt data points, e.g., negative packet sizes.
- Outlier detection: Identify and handle anomalies using z-scores, IQR, or isolation forest methods.

D. Aggregation

- Combine related features: Create summary statistics for network flows, e.g., total packets, average duration.
- Group traffic data by sessions or flows for better contextual understanding.

E. Standardize Data Format

- Ensure all datasets share a consistent:
    - Column naming convention.
    - Labeling scheme, e.g., "Normal" vs. "Anomaly".
- Value format (e.g., "Yes"/"No" vs. 1/0).

F. Encoding Target Variable

- Binary classification: Encode the target as 0 (Normal) and 1 (Attack).
- Multi-class classification: Use integers or one-hot encoding for different attack types.

G. Splitting and Balancing

- Split into train/test/validation sets: Ensure realistic distributions in training and testing splits.
- Balance classes: Use oversampling, e.g., SMOTE, or undersampling to address class imbalance.

H. Data Type Conversion

- Convert all columns to appropriate data types (e.g., float, int, or category) for computational efficiency and modeling compatibility.

I. Dimensionality Reduction

- Incorporate t-SNE or PCA to decrease dimensionality while keeping patterns, especially for datasets with many features.

J. Feature Engineering

- Encode categorical variables: Discrete columns can be transformed leveraging the label and one-hot encoding, or ordinal encoding.
- Normalize numeric features: Use Standardization or the Min-Max technique for numeric values to ensure comparability.
- Extract temporal features: From timestamp columns, derive features like hour, day of the week, or duration.
- Feature selection: Use correlation analysis or feature importance metrics, e.g., mutual information, tree-based models, to retain only the most relevant features.

3.3. Feature Reduction

In order to effectively elevate the resiliency of the IDS, the valuable and important features must be taken from the dataset to decrease the complexity of the model and the data dimensions. It is also called the reduction approach, in which only important features are selected during the training. Generally, there are four main reduction approaches, known as “PCA”, “AE”, “SNDAE”, “RFE”, and SSAE”, as follows:

- **Principal component analysis (PCA):** It is leveraged to obtain valuable features in which the feature’s dimensions are decreased. This will reduce both the model’s computational time and complexity. The preferable features are extracted by this technique to reduce the search range [26]. In cybersecurity, PCA is used to obtain valuable features from the traffic network [39].
- **Recursive feature elimination (Rfe):** This one can be employed to choose preferable features from the entire ones in the given input data, where high ranks of features have only been chosen, while the ones with low ranks are deleted. This approach is used to eliminate redundant features and retrieve important and preferable features. The main aim of incorporating Rfe is to select the most possible sets of significant features [11].
- **The stacked sparse autoencoder (SSAE):** It is one of the deep learning models that is used to select features with high rank behavior and activity data. The classification features have been

presented into SSAE for the first time to automatically extract the deep sparse features. The sparse features with low dimensions have been employed to implement various fundamental classifiers. The SSAE was compared with other feature selection approaches presented by prior studies. The experiments have been conducted using multiclass and binary classifications, and the results ensure the following:

1. Sparse features with high dimensions extracted via SSAE are more effective in identifying intrusion behaviors than prior approaches.
2. Sparse features with high dimensions significantly accelerate the classification of the basic classifiers. To sum up with it has been emphasized that the SSAE offers a new research direction approach to spot intrusion and has been considered as one of the most promising approaches for extracting valuable features [76].

- **Auto-encoder (AE):** AE is an unsupervised feature learning method used to classify both anomaly detection in network and malware. In fact, using deep AE, the original input is transformed into an improved representation through hierarchical feature learning, in which each corresponding level captures into a different degree of complexity. A linear activation function in all units of an AE captures a subspace of the basic parts of the input. Moreover, it is anticipated that incorporating non-linear activation functions with AE can help to detect more valuable features. Given the same number of parameters, AE can achieve better performance compared to a shallow AE [54,76,77].
- **Stacked Non-Symmetrical Deep Autoencoder (SNDAE):** It uses the novel NDAE (Nonlinear Deep Autoencoder) approach in an unsupervised learning for obtaining valuable features. A classification model built from stacked NDAEs has shown to offer great outcomes in feature reduction when the RF algorithm was used with two datasets: NSL-KDD and KDD Cup '99 [54].

It is worth mentioning that both the PCA and Rfe approaches are the two most used and promising scientific ones. The PCA can be employed to decrease the dimensions of data with linear correlation, and its extent is relatively massive; the Rfe is adopted when there is an essential matter that depends on choosing the most relevant features. The Auto-encoder can be employed with complex and non-linear data, such as pictures, sound, and text. The SSAE technique is better than the AE approach in terms of dealing with complicated and complex data. Even though it is expensive and computationally complex, it cannot be leveraged to prevent researchers from using it in their own research areas. The SNDAE is more effective for very large, high-dimensional, and noisy datasets since it can be leveraged to produce deep and non-linear representations that are resistant to the noise. This representation highly refines the classifier's performance.

3.4. Feature Extraction

It is not easy to directly expose intrusions in raw network packets due to the large amount of data in the network. Therefore, to spot these intrusions faster, the valuable features in the raw packet headers should be carefully extracted, and the NIDS model should be trained on these extracted features. Note that the feature extraction technique naturally differentiates adversarial samples in the detector system from those on computer vision. Although packets derived from the extracted features are classified via NIDS, the real functional behavior of network traffic is actually embedded inside the network traffic packets. Thus, changing features via applying adversarial attacks will not produce real network traffic, and reconstructing network packets from adversarial features back into packets is very computational and expensive [78]. Furthermore, unknown relationships among features may occur when feature extraction is applied. This renders incorporating real traffic features into the simulation model challenges since they are affected by the small changes (perturbations). Given the fundamental unit of each feature, the feature extraction process can be divided into flow- or packet-based. A summary of each feature extraction type, with the disadvantages and advantages of feature extraction, is given below:

1. Feature Extraction based on Flow: this technique is the most common used one, where details from network packets in the same connection and flow are extracted. Every extracted feature refers to the main connection between two given devices. The extracted features in this method are mainly obtained from the header of the network packet and can be classified into three steps, as follows:

- Aggregate information can be used to compute the number of specific packet features in the connection, like how many bytes have been sent, the number of flags used, and packets transferred.
- Summary statistics give the entire information in detail regarding the whole connection, including the used protocol/service and the duration of the connection.
- Statistical information is used to evaluate statistics of the packet features in the network connection, e.g., the mean and the standard deviation of interarrival time and packet size of the network. Besides the statistical features, more information and sensitive data could be collected from external tools and software, e.g., system API calls (number of shell prompts), and malware detection engine [66,79]. These characteristics can be used to show the behavior of the internal devices and the network devices in detail. It is worth mentioning that since the system API calls are various among devices, replicating and setting up the external tools and software can be challenging. Therefore, external features cannot be used for general networks, but they are applicable in certain network configurations.

2. Feature Extraction based on Packet: This is another feature extraction technique that can be used to obtain a specific feature at the packet level, rather than at the connection or flow. Usually, features based on packet are statistically aggregated data same to the features flow, but with extra correlation analysis between the inter-packet time and the size of the packet across two devices. The features can be extracted using different time windows to minimize the traffic variations and capture the sequential data of the packets. The packet-based feature extraction techniques can be employed in different detection systems [80,81]. Note that this extraction method is effective for examining the header information of the packet in order to further allow optimal feature extraction.

4. Adversarial Attack against ML-based NIDS Models

4.1. Black Box Attack

4.1.1. Poisoning

Multiple gradient approximation approaches have been presented to accurately estimate the gradient for the attack's performance. In [82], the local search technique has been presented for evaluating the impact of some pixels at the output and approximately calculate the gradient. The ZOO attack can be used to approximate the gradient by checking how output scores change when making small adjustments to the input near x [83]. In a similar way, Ilyas et al. [84] employed a natural evolution approach to obtain the predicted attack value by utilizing a search function [85].

In [86], the authors presented a technique operated in a black-box attack, where two attack strategies have been used—data poisoning and model stealing—to target ML-based NIDS. First, they used an improved version of SMOTE, called A-SMOTE, to create synthetic training data from a small labeled dataset. The created have been employed for training a substitute DNN. Next, they applied a technique called Center Drifting Boundary Pattern (CBP) to generate adversarial examples (AEs) against the substitute model. These AEs were then used to poison the ML-based NIDS. Various datasets: Kyoto 2006+N, SL-KDD, and WSN-DS, with different classifiers (SVM, NB, and LR with linear, sigmoid, and RBF kernels) have been employed to assess the proposal. Experimental findings showed that, for the WSN-DS dataset, their method reduced average model accuracy from 97.71% to 89.61%, demonstrating a significant degradation in detection performance.

In [87], a new AA approach in black-box, namely the Hierarchical Adversarial Attack (HAA), has been introduced. It creates strong attacks to deceive the GNN-based NIDSs in IoT networks with low cost. First, it employs a saliency map to recognize and modify key features with small changes. Then, it selects the most vulnerable nodes for attack using a hierarchical method derived from the Random Walk with Restart (RWR) technique. The authors tested HAA on two GNN models using the UNSW-SOSR2019 dataset [88], Jumping Knowledge Networks (JK-Net) [89], and Graph Convolutional Network (GCN) [90]. They compared HAA with three other methods: Improved Random Walk with Restart (iRWR) [91], Greedily Corrected Random Walk (GCRW) [92], and Resistive Switching Memory (RSM) [93]. The findings illustrated that HAA reduced the accuracy of the two GNN models by over 30%. However, HAA has not been checked well to verify whether it works against adversarial defense techniques. Another technique to create adversarial attacks in real time, utilizing heatmaps from Class Activation Mapping (CAM) and Grad-CAM++, was proposed by Kotak et al. The authors revealed the weaknesses of IoT-NIDS based on ML via leveraging IoT identification models based on payload, like CNNs, Global Average Pooling (GAP), and Fully Connected Neural Network (FCN) [94]. These models were used to process the first seven hundred eighty-four bytes in the TCP payload and change all to a Monochrome image with a size of 28 × 28 with the IoT-trace dataset [95]. The findings demonstrated that the presented defensive model offered the best performance compared to others.

4.1.2. Evasion

In [96], a practical black box attack was proposed to attack different NIDSs in limited queries and decision-based settings. This attack can be used to recognize the benign and anomaly samples to easily bypass the strong NIDS. Query efficient spherical local subspaces with a manifold approach has been employed to create AEs. An algorithm has been implemented on the attacked classifier to reduce the number of queries and reveal the anomaly thresholds. Moreover, the anomaly embedding space has been created using the spherical local subspaces. This technique is effective in exposing anomalies based on threshold decisions, especially once the thresholds between anomalous and benign samples are ambiguous [97]. The proposal has been evaluated employing the Isolation Forests with the CICIDS2018 dataset [98], Deep Support Vector Data Description (DSVDD), Adversarially Learned Anomaly Detection (ALAD) [100], Deep Autoencoding Gaussian Mixture Model (DAGMM), Autoencoder [99], and AnoGAN [102]. The findings showed that this attack accomplished more than 70% successful attack percentage in all classifiers used.

Aiken and Scott-Hayward [103] studied whether adversarial attacks could fool an anomaly-based NIDS in an SDN. They developed a tool called Hydra to generate adversarial evasion threats via altering three main features (size of payloads, two-way traffic flow, and rate of packets) to avoid detection of TCP-SYN DDoS attacks. These adversarial examples were tested on Neptune, an ML-based SDN detection system that uses traffic features and multiple models. The dataset included infected traffic from the DARPA SYN flood dataset and normal traffic from the CICIDS2017 dataset. Different models—FR, SV, LR, and KNN—were evaluated. Results showed that small changes in traffic features could drop Neptune's detection accuracy from 100% to 0% for TCP-SYN DDoS attacks. The outcomes illustrated that LR, SVM, and RF were the significant vulnerable classifiers, while the most robust one was the KNN. A limitation of this work is that Hydra only works for TCP-SYN saturation attacks.

Yan et al. [104] developed DOS-WGAN, a method using Wasserstein GANs with a gradient penalty to create DoS traffic that mimics the normal and evades NIDS detection. They tested it using the KDDCup99 dataset on the CNN classifier, and the outcomes illustrated that the detection rate had been reduced a 47.6%. Separately, Shu et al. [105] proposed Gen-AAL, a GAN-based attack using active learning to fool black-box ML-based NIDS with minimal queries. Their method required only 25 labeled samples to train the GAN and achieved a 98.86% evasion rate when tested on a GBDT-powered NIDS trained/validated on the CICIDS2017 dataset. Guo et al. proposed another black-box attack to generate malicious traffic. First, they trained the model to estimate the threshold in the model that they want to attack. Then, in order to generate hostile network traffic with the structure and tune the model's hyperparameters, the BIM was leveraged. Different models: Residual Network (ResNet), MLP, SVM, CNN, and KNN, with two different datasets: CSE-CIC-IDS2018 and KDD99, have been utilized to evaluate the introduced attack, and it has been pointed out that the attack effectively affected the model performance [106].

TANTRA, modified the traffic network via changing a given network assault's arrival time, has been suggested by Sharon et al. The adversaries employed an LSTM to study and accurately approximate normal patterns in the timing between packets. It then modified the attack traffic using the LSTM to match these normal timing patterns. While TANTRA showed high success in evading detection, adjusting time delays could weaken the attack's effectiveness. However, the authors didn't report whether the manipulated traffic still kept its harmful nature [107].

Zolbayar et al. [108] developed NIDSGAN, which uses predefined constraints and modifies the loss function to create harder-to-detect attacks. Besides fooling the discriminator into mistaking adversarial traffic for benign, the loss function also reduces differences between adversarial and original features. Separately, the authors in [109] suggested IoTGAN to change the IoT device traffic to evade the models. The IoTGAN trains a substitute model in black-box scenarios as the discriminator, while the generator learns to add deceptive perturbations. The attack was tested on five ML classifiers (KNN, RF, SVM, NNs, and DT [110] using the real-world UNSW IoT Trace dataset [95] from 28 IoT devices. Results showed over 90% evasion success against all target models.

In [78], an approach has been presented to automatically modify network data in gray- and black-box scenarios without breaking its functionality. Their approach uses GANs to create adversarial examples (AEs) and uses Particle Swarm Optimization to optimize them for evading detection. The attack was tested on multiple ML/DL-based NIDSs using the Kitsune and CICIDS2017 datasets, achieving over 97% evasion success in half the cases. However, the GAN-based generation process is slow and resource-heavy, making it impractical for real-time attacks, especially in IoT networks. A ZOO attack in black-box settings was developed to test how well an RF-based NIDS could resist attacks. Their ZOO attack produces powerful adversarial examples (AEs) that are completely new and undetectable by the system. Tests showed this attack dropped the model's accuracy down to 48%. To defend against this, they incorporated the GAN model to generate AEs, augmented with the training data, to boost the model's accuracy back up to 76% [5].

Researchers in 2023 created a method named SGAN-IDS that makes adversarial examples (AEs) able to avoid detection by NIDS models. They tested it using the CICIDS2017 dataset and five different machine learning-based IDSs. The system combines GAN technology with self-attention mechanisms to produce attacks that slip past detection systems. Tests showed SGAN-IDS successfully lowered the model's detection rates by 15.93%, ensuring the effectiveness of the presented assault [111]. Researchers tested poisoning attacks on deep learning-based NIDS by injecting malicious data into training sets at different rates (1% to 50%). Results showed that while accuracy only dropped slightly to 0.93, other metrics revealed bigger problems: PPV was 0.082, FPR reached 0.29, and MSE hit 67%. These numbers prove poisoning attacks can seriously harm DL models. The study [55] exposed how vulnerable DL-based NIDS are to these attacks, indicating the real need for strong defenses. The experiments proved that poisoned data can greatly damage performance and is hard to detect.

The study in [62] tested four attack methods (ZOO, GAN, DeepFool, and KDE) with three major datasets: ADFA-LD, CICIDS2018, and CICIDS2019. Researchers evaluated these attacks on a trained ML-based NIDS classifier to determine which attack was strongest. The results clearly showed that DeepFool performed best among all the tested attacks.

4.2. White Box Attack

4.2.1. Poisoning

Fan et al. [112] pointed out problems with current approaches, which utilized gradient-based assaults in order to test Adversarial Training (AdvTrain) defenses [113]. They created a possible unseen assault technique named Non-Gradient Attack (NGA) and a better testing standard called Composite Criterion (CC) that looks at assault success rate and accuracy. Their NGA technique works by searching for adversarial examples outside the decision boundary, then slowly moving them closer to real data while keeping them misclassified. Tests on CIFAR-10 and CIFAR-100 datasets [114] compared AdvTrain's performance against four common gradient attacks (C&W, FGSM, PGD, and BIM). Results showed that previous estimates of DNN-based IoT NIDS security might have been too optimistic. The new NGA+CC method gives a more accurate way to test these systems, both with and without AdvTrain defenses. The authors noted their NGA method currently has slow convergence but plan to improve this in possible future research.

4.2.2. Evasion

Researchers in [115] used Mutual Information (MI) to identify the valuable features in DoS and benign traffic. They crafted attacks by reducing the differences between these key DoS features using l1 norm minimization. The attack has been assessed utilizing DNN and SVM models with the NSL-KDD dataset. The outcomes showed the DNN's DoS detection accuracy fell by 70.7%. Abusnaina et al. [116] studied whether standard adversarial attack methods work for flow-based IDS in SDN networks. They found that these methods aren't suitable because flow features are interconnected (unlike image pixels) and attacks must create realistic network flows. Their solution, FlowMerge, creates believable attack flows by blending original flows with representative "mask" flows from a target class using either averaging or accumulation. When tested against a CNN classifier, standard methods (C&W, ElasticNet, DeepFool, MIM, PGD) achieved 99.84% untargeted attack success, while FlowMerge achieved 100% success for targeted attacks while maintaining realistic flows. The study also showed adversarial training could defend against standard attacks but not against FlowMerge. A limitation of the method's reliance is given on certain assumptions about traffic features.

Hashemi et al. [117] tested a white-box attack using the CIC-IDS2018 dataset on different NIDSs: BiGAN [118], DAGMM, and Kitsune [81]. This method, similar to work in [119], used just three simple packet changes: delaying packets, splitting packets, or adding new packets. The attacks were created by repeatedly trying these changes and checking if they successfully lowered the NIDS's anomaly score below its detection threshold. In [12], AIDAE (Anti-Intrusion Detection AutoEncoder), a classifier that creates fake network features that look normal to IDSs, has been suggested. It learns patterns from real network data and generates convincing fake features without needing access to the target IDS during training. AIDAE uses both an auto-encoder and a GAN, and smartly handles both number-based and category-based data features. The proposal has been assessed utilizing NSL-KDD, CIC-IDS-2017, and UNSW-NB15 datasets with seven detection methods (Adaboost, CNN, LSTM, KNN, DT, RF, and LR), and the outcomes showed that the proposal successfully reduced accuracy significantly, to about 7.11%.

In [34], the genetic algorithm (GA), GAN, and particle swarm optimization (PSO) have been used to produce strong adversarial examples. They tested the proposal on several classifiers, including BAG, DT, QDA, LDA, SVM, MLP, GB, NB, RF, KNN, and LR with NSL-KDDU and NSW-NB15 datasets. The authors compared these techniques to the Monte Carlo simulation that creates random infected data. The average evasion rates have been reported as 96.19% (MC), 99.99% (PSO), 99.95% (GA), and 92.60% (GAN) using the NSL-KDD dataset. For UNSW-NB15, the rates have been reported as 98.06% (PSO), 99.53% (MC), 99.61% (GAN), and 100% (GA). Teuffenbach et al. [120] introduced a method to create adversarial examples (AEs) for NIDS while respecting network traffic feature constraints. Features are grouped, and each group gets a weight based on how easy it is to modify them. This technique uses the C&W threat to optimize perturbations while considering these constraints and weights. Attack difficulty depends on two limits: how many features can be changed (feature budget) and how much they can be altered (perturbation budget). This method ensures realistic adversarial flows by only modifying accessible and independent features (with an average change of less than 0.2). The technique was tested against FGSM and C&W attacks using DNN, DBN, and AE models with two datasets: NSL-KDD and CICIDS2017. The outcomes illustrated that the AE classifier was the most robust against the created adversarial samples but had the minimum accuracy in detecting anomalies compared to DBN and DNN models. Meanwhile, DNN and DBN handled adversarial examples better for DoS attacks than for PortScan or Probe attacks.

Wang et al. [121] pointed out that most studies wrongly use image-based adversarial example (AE) generation methods for NIDS, which don't fit network traffic's unique needs. They developed a Constraint-Iteration Fast Gradient Sign Method (CIFGSM) that accounts for network traffic complexity, feature types, and their relationships. Testing CIFGSM against IFGSM leveraging MLP, DT, and CNN models with the NSL-KDD dataset ensured CIFGSM performed better in accuracy, feature matching, Euclidean distance, and matrix rankings. Under CIFGSM attacks, classifier accuracy dropped to 0.25 (DT), 0.68 (CNN), and 0.73 (MLP). InfoGain Ratio, a feature valuable ranking technique of identifying malicious from benign ones, was proposed by Anthi et al. The perturbation has been produced using the top-ranked features. The AAs have been assessed by SVM, Bayesian Network, DT, and RF, where these AAs have been produced by changing the affected features individually or simultaneously. The experimental findings indicated that the presented technique can create strong AAs for DoS attacks and is effective for systems utilizing supervised ML-based NIDS and create adversarial attacks for DoS attacks [122].

In [123], a DL model has been used to evaluate the IoT-NIDS based on ML. To do so, an integrated CNN and LSTM (LSTM-CNN) classifier for IoT-NIDS has been achieved to assess the classifier's robustness against AEs by modifying the real device conditions, including the CPU load to prevent the attacks, temperature adjusts, and device restarts. The proposal was evaluated leveraging the LwHBench dataset to expose other AEs, such as PGD, Boundary Attack, JSMA, BIM, MIM, FGSM, and C&W [124]. The integrated classifier accomplished an average F1 score of 96% with an 80% TPR for system detection. Even though the classifier remained robust to temperature-based assaults once exposed to various evasion AEs, a few attacks, e.g., BIM, MIM, and FGSM, have bypassed the detection system. Furthermore, both the adversarial training and the model distillation have been incorporated to refine the classifier performance. In fact, combining the adversarial training and the model distillation generated a robust defensive technique.

This study [125] examined how well deep learning (DL) based NIDS can withstand popular adversarial evasion attacks like C&W, JSMA [44], PGD [45], and FGSM [43]. The goal was to trick the NIDS into labeling malicious traffic as normal by feeding it carefully crafted adversarial examples. The attacks noticeably hurt the system's performance, lowering its AUC, f-score, recall, precision, and accuracy. Based on classification reports and confusion matrices, the C&W is generally shown to be a strong attack since its impact on the IDS was similar to FGSM, JSMA, and PGD, but under the C&W attack, the score of the AUC hit about 64, higher than FGSM (59.232) and PGD (58.485) but lower than JSMA (68.037), where the experiments used the CICIDS2017 dataset.

4.3. Gray Box Attack

4.3.1. Poisoning

In [126], the authors introduce a framework to classify and explain various attacks that could target feature selection algorithms. This builds on existing attack models employed to assess the security of unsupervised and supervised learning systems. Using this framework, they then define poisoning attack strategies against common embedded feature selection methods, such as ridge regression, elastic net, and LASSO (least absolute shrinkage and selection operator). The study in [127] created poisoning attacks to attack SVMs. These created threats work by adding carefully designed bad data to the training set, which makes the SVM perform worse. Most machine learning methods assume training data comes from normal, clean sources - but this isn't always true for security applications. The attack method uses gradient ascent (a climbing optimization technique) based on how the SVM finds its best solution. It works with different types of SVMs (even complex non-linear ones) by calculating the attack in the original input space. Tests show this approach consistently finds ways to make the classifier perform much worse by exploiting weak spots in the learning process.

4.3.2. Evasion

In [128], an investigation on how AEs affected the DNN classifier was presented. It has been pointed out that the attacker is capable to generate strong AEs to deceive the model. The experimental results were conducted employing WGAN, C&W, and ZOO attacks. Under the three attacks, the effectiveness of the DNN model on NSL-KDD dataset is drastically reduced, implying that attacks with high detrimental implications might be launched without even knowing the detection model's underlying workings. For ZOO, WGAN, and C&W, the attacks result in 70 percent, 62 percent, and 24 percent F1 score deterioration in accuracy, respectively. The replacement attack model is determined to be inefficient, while the ZOO assault is proven to be the most successful and robust. Note that this approach is computationally costly since it must contact the detection system to upgrade the gradients. This could limit its utility in real-world circumstances. This happens because the NIDS limits the queries to a certain number that an attacker can generate.

Packet-Level Attacks (PLA) work directly on network packets to create practical adversarial examples. Homoliak et al. [119] tested this approach using a gradient boosting (GB) threat model against five basic ML classifiers (Naive Bayes, Decision Trees, SVM, Logistic Regression, and Kernel Density Naive Bayes) on their custom ASNM-NPBO dataset. The attacks randomly applied six types of packet changes to trick the classifiers: delaying packets, losing packets, damaging packets, duplicating packets, rearranging packet order, and breaking up payloads. Each modification was tested to see if it could bypass detection.

4.4. Combination of Poisoning and Evasion

Another scenario of attack explored a certain type of DoS-based AEs to evade DoS IDS based on ANN [129]. The researchers proposed an enhanced boundary approach to relieve malicious DoS attacks by analyzing the features within the DoS traffic. The purpose of this work was to improve the Mahalanobis distance by changing the discrete and continuous features of DoS examples. Two datasets: CICIDS2017 and KDDcup99, with two trained ANN models, have been used, and the experimental results revealed that, by applying the proposed technique, adversarial DoS samples can be generated with a few query number, and this further decreased the predicted output of the true class to about 49 percent.

Apruzzese et al. [32] tested attacks using the CTU-13 dataset on three ML models, RF, MLP, and K-NN, based on NIDS. Normally, these classifiers performed well with recall scores from 0.93 (K-NN) to 0.97 (RF). For the attack, they modified three features: bytes exchanged, connection duration, and total packets by adding random values within their normal ranges. This caused performance to drop sharply, with recall falling to 0.31 (MLP) and 0.34 (RF). The study then tested two defenses: adversarial retraining improved recall to 0.49-0.60, while removing attacked features before training worked even better, boosting recall to 0.76-0.89. In [41], the polymorphic DDoS attacks have been created by incorporating GANs, where polymorphic attacks constantly change their attack patterns (by modifying feature counts and swapping features) to evade DS detection systems. These evolving attacks successfully bypassed detection systems while keeping false alarms rare. The study, conducted on utilizing RF, NB, LR, and DT models with the CICIDS2017 dataset, showed that detection rates plummeted to 5.23% (modified feature counts) and 3.89% (swapped features). While traditional incremental training defenses failed against these attacks, the method proves valuable for generating synthetic attack data to strengthen NIDS training and improve the detection of emerging threats. Table 4 shows the studies in brief on the adversarial attack generation using ML-based NIDS approaches.

Table 4. Summarized studies on adversarial attacks (poisoning and evasion).

	Year	Ref.	Dataset	Setting	Models	Strategy
	2012	[127]	Real data, Artificial data	Gray Box	SVM	Poisoning
	2015	[126]	PDF Malwav	Gray Box	LASSO, Ridge Regression	Poisoning
	2018	[86]	WSN- DS, Kyoto2006 +, NSL-KDD	Black Box	DNN	Poisoning
	2019	[32]	Tu-13	Black Box	K-NN, ML, RF	Evasion, Poisoning
	2019	[129]	CICIDS2017, KDDCup99	Black Box	DNN	Evasion, Poisoning
	2019	[96]	CICIDS2018	Black Box	DAGMM, IF, GAN, BIGM, AE, One-class SVM	Evasion
	2019	[103]	CISIDS2017, DARPA	Black Box	K-NN, RF, SVM, LR	Evasion
	2019	[104]	KDDCup99	Black Box	CNN	Evasion
	2019	[115]	NSL-KDD	White Box	SVM, DNN	Evasion
	2019	[116]	KDD-Cup99, NSL-KDD	White Box	CNN	Evasion
	2019	[119]	ASNM-NPBO	Gray Box	NB, DT, SVM, LR, NB	Evasion
	2020	[34]	NSL-KDD, UNSWNB15	White Box	SVM, DT, NB, K-NN, RF, MLP	Evasion
	2020	[41]	CISIDS2017	Black Box	DT, LR, NB, RF	Evasion, Poisoning
	2020	[105]	CICIDS2017	Black Box	GAN with Activation Learning	Evasion
	2020	[78]	CICIDS2017, Kitsune	Black Box, Gray Box	Kitsune, IF, LR, SVM, DT, MLP	Evasion
	2020	[12]	CICIDS2017, NSL-KDD, UNSW-NB15	White Box	CNN, DT, K-NN, LR, LSTM, RF, Adaboost	Evasion
	2020	[120]	CICIDS2017, NSL-KDD	White Box	DNN, AE, DBW	Evasion
	2020	[128]	NSL-KDD	Gray Box	DNN	Evasion
	2021	[87]	UNSW-SOSR2019	Black Box	GNN	Poisoning
	2021	[106]	CICIDS2018, KDDCup99	Black Box	CNN, KNN, MLP, SVM	Evasion
ATTAKS	2021	[107]	UNSW, IOT Trace	Black Box	Random Forest, Decision Tree, KNN, SVM	Evasion

2021	[114]	CIFAR-100, CIFAR-10	White Box	DNN	Poisoning
2022	[94]	IOT-Trace	Black Box	FCN, GAP, CNN	Poisoning
2022	[108]	CISIDS2017, NSL-KDD	Black Box	KNN, SVM, DNN	Evasion
2022	[5]	CSE-CICIDS2018	Black Box	GAN, RF	Evasion
2023	[111]	NSL-KDD, CICIDS2017	Black Box	SGAN-IDS	Evasion
2024	[62]	CSE.CICIDS2018, ADFA-LDs, CICIDS2019	Black Box	DNN	Evasion
2024	[55]	CSE-CICIDS2017	Black Box	DL	Poisoning
2024	[123]	LWHBench	White Box	CNN and LSTM	Evasion

5. Defending ML-based NIDS Models

5.1. Black Box Attack

5.1.1. Poisoning

Many scholars have introduced various defensive techniques to prevent AEs. For example, the authors in [130] introduced a safety-net system that combines the attack detector with the original classifier, which tests the deeper layers of the classifier to spot AEs. Metzen [131] explored a similar approach. Another defense called Reject on Negative Impact (RONI) [132] works by analyzing each training sample's effect on accuracy and removing those that significantly hurt performance.

The paper in [133] evaluated an outlier detector as a defense against malicious data injection attacks. The idea was to find and eliminate suspicious training data using statistical checks. This method has been checked to see whether it could spot poisoned data made by two strong attack methods: the input instance-key strategy and the Blended Accessory Injection strategy. The results showed the outlier detector failed to catch any poisoned samples from either attack. This means these attack methods can create poisoning data that blends in and avoids detection.

5.1.2. Evasion

In [11], GANs have been incorporated to create AEs that change only non-essential network traffic features without affecting functionality. They also proposed using GAN-generated AEs during training to strengthen defenses. Testing on the KDDCup99 dataset with multiple classifiers (DNN, LE, SVM, KNN, NB, RF, DT, GB) showed initial accuracy ranged from 43.44% (SVM) to 65.38% (GB). After GAN-based training, performance improved significantly, with accuracy reaching 86.64% (LR) and 79.31% (KNN). Wang et al. [134] adapted mutation testing from software engineering to detect adversarial examples in DNNs. Their method works by randomly changing the neural network's structure during operation and checking how often these changes affect the output labels. This "label change rate" helps identify suspicious inputs. The approach successfully detected C&W attacks [58] even without prior knowledge about the attack method.

In (2022), the authors first prepared and cleaned the NSL-KDD dataset by converting all symbols into numerical values. Then, the PCA has been incorporated to extract features and classify the data utilizing SVM, LR, and RF models. Their findings showed SVM achieved the best accuracy (98%), then by RF (85%) and LR (78%), measured using accuracy, precision, and recall metrics [33]. Separately, Faker and Dogdou combined big data analytics with deep learning to improve intrusion detection systems. They tested three classifiers—Random Forest, Gradient Boosting Tree (as an ensemble), and Deep Feed Forward Neural Network (DNN). The DNN performed best, by obtaining the highest accuracy, 99.16%, in binary classification and in the multiclass model was 97.01% [135]. In 2022, researchers developed a defense system combining an FR model with a GAN-powered deep learning approach. They trained this model on a blended dataset containing both real network data and artificial samples produced by ZOO and GAN attack methods. The team also used PCA (Principal Component Analysis) to choose the highest valuable features from the generated adversarial examples (AEs), which helped improve the model during training. Test results showed the system's accuracy improved by 27% compared to previous methods [5].

The researchers developed a DeepFool-based defense method to protect the system against four different threat models: DeepFool, KDE, ZOO, and GAN. They tested this approach utilizing CICIDS2019, CICIDS2018, and ADFA-LD. The experimental findings demonstrated that their defensive solution outperformed other models by detecting more attacks successfully [62].

5.2. White Box Attack

5.2.1. Poisoning

Benzaıd et al. [136] proposed a DDoS self-protection technique, which resists malicious attacks. The proposal employed the software-defined networking and DL to automatically spot and prevent DDoS threats. It showed good performance in improving server response time and reducing system load. The detection model was implemented leveraging a Multilayer Perceptron (MLP) trained on both normal and DDoS traffic from the CICIDS2017 dataset. To defend against attacks, adversarial training was applied using adversarial examples (AEs) created with the FGSM technique.

Benaddi et al. [137] employed Generative Adversarial Networks (GANs) to train NIDS based on Distributional Reinforcement Learning (DRL). This approach helped detect rare network threats and refined the robustness and the performance of NIDS in Industrial Internet of Things (IIoT) environments. Similarly, authors [138] introduced the Decentralized Swift Vigilance (Desvig) approach, which incorporates Conditional GANs (C-GANs) [139] to enable fast-response, high-efficiency security solutions for industrial settings. Benaddi et al. [140] introduced C-GAN (Conditional GAN) to strengthen a hybrid LSTM-CNN-based NIDS for IoT networks. They used this as an external training network to refine the classifier's robustness. The approach was actually adapted from an auxiliary classifier GAN (AC-GAN) framework [141].

5.2.2. Evasion

Stochastic Activation Pruning (SAP) [142] is a post-hoc defense method against adversarial attacks. During the neural network's forward pass, it randomly drops activations in each layer, with smaller activations being more likely to be removed. To balance this, the remaining activations in later layers are scaled up to maintain input strength. By pruning weaker activations, SAP reduces the chance that adversarial changes will build up in deeper layers, improving the model's resistance to adversarial examples.

In [143], a defensive technique was presented to prevent AEs. The DNN classifier has been trained against AEs utilizing the min–max method with the UNSW-NB15 dataset. The max technique has been used to construct AEs, which increase the loss, while the min technique has been used as a protection method to decrease the loss of the inserted infected samples at the training phase. Four different models have been leveraged to create AEs. To refine the resilience of the classifier, the model was trained leveraging both the generated AEs and normal instances for each technique. The used classifiers have been attacked via four groups of AEs during the testing for every strategy. Among all adversarial attack strategies, the "dFGSMS" system, such as trained with AEs produced via dFGSMS, had the least evasion results. Meanwhile, among all other models, the "BGAS" beats all attack strategies. PCA was also used for data dimension reduction purposes and mostly improved the strength of trained models versus AEs. When compared to the initial trial, the evasion rates have been lowered to about three times, and the trained models were equally resilient to all forms of AEs.

Researchers in [4] introduced Reconstruction from Partial Observation (RePO) to make an unsupervised denoising autoencoder better at detecting AEs in NIDS for both packet and flow-based data. They compared RePO with baseline methods like Kitsune [81], BiGAN [118], and DAGMM [99], and used adversarial attacks from [117]. Testing on the CICIDS2017 dataset, RePO enhanced the adversarial detection system by 29% and 45% compared to the baselines. The authors in [144] introduced a method to reveal AAs during inference by analyzing neural network activations. They trained an artificial neural network (ANN) on part of the CICIDS2017 dataset and recorded its activations during testing. These have been then leveraged to train and evaluate different models—Adaboost, ANN, Random Forest (RF), and SVM—to identify adversarial examples (AEs). Their results showed that RF and KNN achieved a high recall of 0.99 against evasion attacks generated utilizing PGD, C&W, BIM, and FGSM methods.

Peng et al. [31] created a way to spot harmful fake data (ASD) using a two-way GAN (BiGAN) to shield NIDS from attacks. Their system has three pieces: (1) an encoder, (2) a discriminator, and (3) a generator. The generator studies normal traffic patterns during training. The ASD then looks for oddities in the data by checking reconstruction mistakes, while the discriminator examines these errors. This lets the ASD block suspicious data before it hits the NIDS. Tests revealed that without ASD, the DNN's performance fell sharply: 60.46% drop (FGSM attacks), 28.23% (PGD), and 46.5% (MI-FGSM). With ASD, performance bounced back—rising 26.46% against PGD and 11.85% against FGSM—though its impact on IM-FGSM remained unclear. Ganesan and Sarac [145] used a feature selection method (RFR) to eliminate features that attackers exploit to fool detection systems. They experimented with smaller sets of features to create a group of ML classifiers: LR, RF, SVM, and DNN, which are harder to trick. The tests used three datasets, KDDCup99, CICIDS, and DARPA, with Hydra creating the attack data. The results proved that this group of models working together could catch attacks that a single model using all features would miss.

The research studied whether attack methods that work on one ML-based NIDS could also fool other differently-designed NIDS models, when the attacker has no internal knowledge (black-box conditions) [146]. To circumvent a DNN model, both FGSM and PGD models were utilized to create AEs. The transferability of these cases was then investigated across prominent ML techniques, LDA, DT, FR, LR, and SVM. The trials emphasized that the DNN classifier was the highest degraded, while the remaining classifiers were less degraded, albeit to varying levels, due to their differentiable unit design. Furthermore, the study found that utilizing a combination of detection systems was resilient to transferable threats compared to an individual classifier. The work ensured that the used Detect and Reject strategies helped to minimize the effects of adversarial attack transferability over ML-based detection classifiers.

Novaes et al. [147] used a GAN to detect DDoS attacks in SDN networks. Since GANs can create adversarial traffic, they also used it for defense through adversarial training. The GAN performed better than MLP, CNN, and LSTM models when tested on the CICDDoS 2019 dataset and simulated SDN traffic. Yumlembam et al. [148] introduced a GAN-based system to strengthen Android malicious detection utilizing Graph Neural Networks (GNN). Their method uses GNN to create graph embeddings from API call graphs, combining 'Permission' and 'Intent' features to boost malware classification. They developed VGAE-MalGAN, which adds fake nodes and edges to API graphs, tricking GNN-based malware detectors while keeping the original malware behavior intact. VGAE-MalGAN has two parts: a Generator (modified Variational Graph Autoencoder) and a substitute detector (GraphSAGE model). Experiments showed that retraining the model with VGAE-MalGAN makes it more resistant to attacks while maintaining high detection accuracy. The study used the CMaldroid and Drebin datasets for evaluation.

To make the model stronger against attacks, three defense methods have been checked: adversarial training, high-confidence, and Gaussian data augmentation (GDA). After applying these defenses, the model's confidence score improved significantly in all four attack scenarios. This study examines different advanced AAs and their defensive technique in NIDS, supported by detailed experiments. The results suggest that DL-based NIDS should not be used for critical, real-time applications unless strong defenses are in place to block adversarial attacks [125]. The study in [149] introduced a defensive approach to mitigate the strong C&W attacks. During the proposal training, the GDA with the adjusted adversarial training has been employed, while during testing, the Feature Squeezing (FS) technique was executed on adversarial samples before sending them to the NIDS to be classified. The method is tested using the newer dataset, CIC-DDoS-2019, and results are measured using classification reports and confusion metrics [149]. 2025 hybrid framework boosts robustness by 20-30% against evasion, enhancing defenses through integrated adversarial training [150].

5.3. Gray Box Attack

5.3.1. Poisoning

In [151], a defense method is proposed against certain attacks. Causative attacks work by tricking the email filter into learning incorrectly when trained on malicious emails, leading to misclassifications. Each attack email worsens the performance of the filter, so when the harmful impact of each email was measured, damaging ones, from the training set, could be eliminated. The detection technique measures an email's impact by comparing filter performance with and without it, where this technique was named as (RONI) Reject On Negative Impact. To test this, a small 50-email validation set (V) and 20-email training set (T) have been repeatedly talked five times to obtain correct accuracy. The T and T∪Q (where Q is the test email) and measure Q's impact were trained by obtaining the average change in misclassifications on V. If Q's effect is harmful, it is removed. RONI was tested with 120 normal spam emails and dictionary attacks (using Aspell and Usenet dictionaries). Results show that RONI can effectively prevent dictionary threats, in which I can almost detect all threat emails without wrongly flagging harmless ones. Note that threat emails drop the accuracy to about 6.8 in true negatives (correctly classified safe emails), while normal spam caused at most 4.4 in true negatives, rendering a threshold more efficient. However, RONI fails against focused attacks because they target future emails, leaving almost no trace in the training data.

5.3.2. Evasion

Apruzzese et al. in [152] introduced AppCon, which is a combinational technique integrated strong defenses to protect NIDS against adversarial evasion threats with high performance even when no attacks are present. The AppCon reduced the number of malicious instances that are needed by an opponent to bypass the detection system. The combinations of the infected feature flow have been slightly modified to create robust AEs [153]. The AppCon has been assessed leveraging finely built NIDS classifiers, involving the CTU-13 dataset with different models. The results demonstrated that the proposal decreased the effectiveness the evasion threats by more than 50% without reducing the classifier performance.

Siganos et al. (2023) created an AI-based IDS for IoT systems, making it easier to understand using SHapley Additive exPlanations (SHAP) with both ML and DL models. This helped explain how the IDS works, removing the "black-box" problem. The system performed well in detection and met the growing need for Explainable AI in complex systems. For testing, they used two balanced datasets, including IEC 60870-5-104, and compared ten different ML models. Among these models, RF was the most accurate, achieving an F1-score of 66% [154]. In 2023, researchers developed an AI-powered NIDS to fix data imbalance problems. The system generates realistic artificial data and monitors training balance to address this issue. When tested with auto-encoder, DNN, and CNN detection models, it performed better than existing methods. Experiments used both real-world and IoT datasets, showing major improvements - hitting 93.2% and 87% accuracies on NSL-KDD and UNSW-NB15 datasets, respectively. The system also effectively detected network attacks in IoT data [155].

In 2024, researchers created an attention-GAN method to better detect cyber threats. This approach generates powerful attack samples that help train detection models to catch more threats. By combining GAN with attention mechanisms, the system improves the detection of sophisticated attacks. Tests using both CICIDS2017 and KDD Cup datasets showed impressive results - reaching 97% for recall, precision, and F1-scores, and 99.69% accuracy. The method also includes a new defense strategy against future threats [156].

5.4. Combination of Poisoning and Evasion

Qureshi et al. [157] proposed a threat detection technique named RNN-ADV, which employed Artificial Bee Colony (ABC) with Random Neural Network method to spot attack attempts. They checked how well it works to prohibit JSMA threats and compared it with the DNNs trained on the NSL-KDD dataset. When under attack, RNN-ADV performed better - scoring 44.65% for DoS attacks and 52.60% F1 for benign traffic, while the DNN only managed 35.69% and 25.89% for these same cases. The results showed that RNN-ADV functioned excellently in classifying adversarial examples with higher precision and accuracy than DNN. Table 5 illustrates the research studies in brief on defensive ML-based NIDS.

Table 5. Summarized the defensive studied against adversarial attacks.

	Year	Ref.	Dataset	Setting	Models	Strategy
	2009	[151]	Emails, Dictionaries	Gray Box	Spam Boyes	Poisoning
Defenses against adversarial examples and attacks	2010	[132]	Healthcare dataset, Malware dataset, image dataset	Black Box	Regression Model, Classification Model, Generative Model	Poisoning
	2017	[133]	YouTube dataset, Aligned face	Black Box	Deep ID, VGG-face	Poisoning
	2019	[11]	KDDCup99	Black Box	SVM, DNN, LE, KNN, NB, RF, DT	Evasion
	2019	[134]	CIFAR10, MNIST	Black Box	DNN	Evasion
	2019	[142]	UNSW-NB15	White Box	DNN	Evasion
	2020	[4]	CICDS2017	White Box	DAGMM, BIGAN, Kitsune, DAE	Evasion
	2020	[31]	NSL-KDD	White Box	DNN	Evasion
	2020	[157]	NSL-KDD	White Box	DNN, ABC, and RNN-ADV	Evasion, poisoning
	2020	[136]	CICIDS2017	White Box	MLP	Poisoning
	2020	[144]	CICDS2017	White Box	Adaboost, DNN, EF, SVM, K-NN	Evasion
	2020	[153]	CTU-13	Gray Box	Adaboost, DT, MLP, RF, WND	Evasion

2021	[145]	CICDS2017, DARPA, KDDCup99	White Box	RF, LR, SVM, DNN	Evasion
2021	[146]	NSL-KDD	White Box	SVM, RF, LR, DT, LDA, and DNN.	Evasion
2021	[147]	CICIDS2019	White Box	LSTM, MLP, GAN, CNN	Evasion
2022	[5]	CSE-CICIDS2018	Black Box	RF, GAN	Evasion
2022	[135]	NSL-KDD	Black Box	DNN, SVM	Evasion
2022	[137]	Orchestration System dataset / The Distributed Smart Space.	White Box	DRL	Poisoning
2022	[148]	CMaLdroid, Drebin	White Box	GNN	Evasion
2023	[154]	IEC60879-5-104, CIC-IOT Dataset. 2022.	Gray Box	DT, RF, Adaboost, DNN, LR	Evasion
2023	[154]	NSL-KDD, UNSW-NB15	Gray Box	CNN, DNN, GAN, Autoencoder	Evasion
2024	[62]	CSE.CICIDS2018, ADFA-LDs, CICIDS2019	Black Box	DNN	Evasion
2024	[149]	CICIDOS2019	White Box	Feature Squeezing	Evasion
2024	[156]	CICIDS2017, KDD-Cup	Gray Box	Attention, GAN	Evasion
2025	[10]	Various (taxonomy-focused)	White/Black/ Gray	ML models (general taxonomy)	Evasion, Poisoning
2025	[14]	CICIDS2017, NSL-KDD	Black Box	DNN, RF	Evasion
2025	[150]	CICIDS2019	Gray Box	Ensemble (RF + DNN)	Evasion

6. Discussion

In this section, we suggest and give insight into some prospective concepts for further future research studies. NIDS operates in hostile environments, and therefore, it is subject to many threats of adversarial attacks. To mitigate this dilemma, designing solid systems and resistance to these attacks is necessary. The results of experimental studies indicated a high rate of smuggling attacks against golden-free systems. We analyzed and summarized the ML-based NIDS research for the past five years, from 2020 to 2025, as shown in Figure 3.

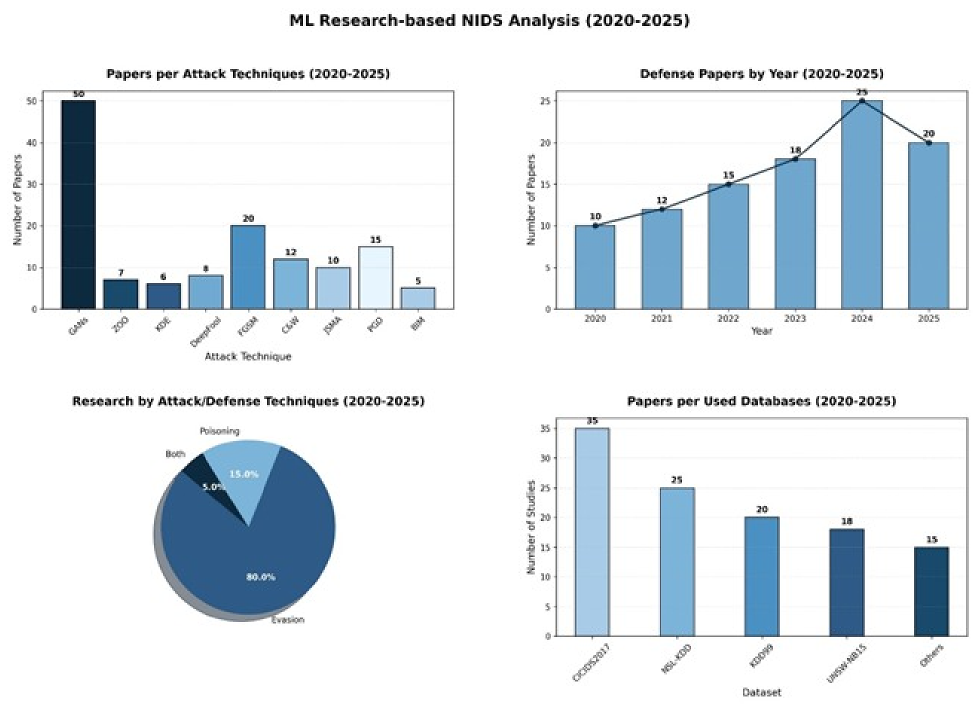

Figure 3. Analyzing the NIDS based on ML research for the past five years from 2020 to 2025.

Adversarial attacks can largely compromise and deteriorate the classifiers of NIDS. This belongs to the transferability of AEs over several architectures, where an opponent can create a substitution model and then use these hostile examples to later attack other targeted systems.

In general, many survey studies are conducted; most of the attempts to mitigate the attack try to provide ways to improve and develop the overall performance of network systems as a defense mechanism. However, it is interesting that a few of these studies provided defensive solutions and evaluated them through several factors, such as time, memory, complexity, cost, power, and their suitability for the target field and ignoring available methods such as Robust optimization, data sanitization, and game theory.

From the previous tables given in this work, several considerations can be expressed, the first of which is that many studies used the widely deprecated dataset KDD99, as in [11,158,159], that has a lot of flaws and does not represent the current attacks on real-world applications. Consequently, emerging and large realistic data are requested to mitigate sophisticated attacks, such as CICIDS2017, CSECICIDS2018, and LITENT2020, and also moderate configuration networks according to the growing size of their users. Using these recent datasets enables scholars to have results that are right and consistent with modern threats against technology in modern networks. Some researchers leverage more than one dataset in their experiments, as in [12,41,160], and this is an important point for the development of sophisticated systems. Therefore, the evaluation of these experiments using large datasets against adversarial attacks should show that the proposed technique is robust and resilient. Furthermore, these findings permit researchers to better comprehend the impact of their attacks in current detection approaches by taking into consideration up-to-date datasets.

We also realize that only a few proposed techniques involve poisoning efforts as in [41,129,161], whereas the majority of other techniques concentrate on evasion assaults, as in [162–164], during inference. These are important findings since poisoning assaults are hard to carry out in real conditions and circumstances, since the attackers are not able to control the training data. In addition, many research works have concentrated on specific attack techniques, including FGSM [163], which obtained the highest percentage in their use, because of its ease, efficiency, and spread, while there are other methods that are not being applied, such as the ZOO method (Black box attack), especially in the field of NIDS. It has been noted that the FGSM method (white box attack) might not be suitable for the network field since it needs to change every feature and provide hostile examples. This means exaggerating the knowledge of the enemy and its full control to alter all the features of the system in such a fine-grained approach. Also, the focus of most studies was on the adversary knowledge factor, called the white box attack, in which the opponent is strong in attacking the target system. On the contrary, a few studies have focused on the weak adversary side within the black box control, such as in [128,162,165], in which adversarial threats have no knowledge about the attacked system, and an attacker can enter the hostile data and monitor the outputs, aiming to understand the behavior of the system. However, it is essential to take into account the number of attempts that a malicious agent needs in order to reach the stage of triggering the detection mechanism. Therefore, to evaluate the systems adequately, there must be diversity in terms of the factors of the attacker model to know its mission, the acceptability, and its suitability in a wide variety of scenarios in the NIDS domain. This conclusion demonstrates the potential of new research avenues for evaluating more complicated yet realistic antagonistic samples.

A noteworthy problem is that many researchers recommended solutions, such as ensemble learning [159,160] and feature reduction [166], that have significant flaws, rendering them inefficient for preventing NIDSs from adversarial assaults. In the non-adversarial context, feature removal affects the detection of the model's performance, while ensemble learning lacks interoperability and needs extensive calculations. Such a method could also not be resistant to adversarial samples when it relates to the transferability. As a result, creating a highly robust architecture for machine learning systems that can withstand adversarial attacks remains a research challenge. Although most of the research concentrate on conventional networks, there are fewer studies that investigate adversarial assaults in IoT and wireless networks.

Many studies have focused on the aspect of images, and a few of them are interested in the aspect of networks domain, and this may be because the features in the field of images don not restricts, unlike the features of networks that contain different types of data, large sizes, errors, and many forms, such as digital, continuous and binary, correlated and interdependent, where these restrictions are not fully addressed, leading to incorrect or mismatched results for hostile traffic.

Furthermore, not all studies have considered the possibility of initiating these assaults in real-world circumstances. A typical research project can start with any threat model and then examine the attack's impacts, with no or inadequate consideration of the scenario's plausibility [29]. Others believe that an adversary can conduct an infinite number of tries against the NIDSs without being detected [129]. Many criteria, such as required processing power, the number of changeable features, and the magnitude of injected instances, limit the choice of a suitable adversarial generation approach [13]. Determining the impact of AAs on ML model is essential and aims to create a more robust and secure system since the cybersecurity systems should always include actual concerns and adversaries. As a result, there is a loophole between academic and actual-world contexts that must be bridged in the interests of mutual benefit. We would like to highlight three important keys that can help to open the door for future work researchers, as follows:

- There is no defensive technique, including NIDS or hybrid ML with NIDS, that has been presented yet that can provide a very high attack detection rate against strong attacks, such as GAN, DeepFool, and KDE. In [44], extensive experiments have been conducted, and it has been pointed out that no defensive approaches were able to prevent sophisticated attacks generated via incorporating different datasets with different percentage rates of attacks.
- Although GANs have been shown to be the most promising and powerful attacks, the recent study in [44] confirmed that other attack models, e.g., DeepFools, can be stronger than GANs if they have been trained carefully.
- The implementation of adversarial attacks has been extensively leveraged and explored in deep detail in both image and speech processing areas. However, their impacts on NIDS remain to be explored. Creating adversarial examples in the real-time domain from network traffic in the

physical layer has not been proposed or presented yet. This one is especially important to open the door for future interesting research areas.

- Many ML-based NIDS can detect attacks very accurately, but this usually makes them slower and more resource-intensive to run. On the other hand, simpler detection systems can work faster and cheaper by removing less critical features. While this approach makes the system more efficient, it does cause a small drop in accuracy [167].
- Current defensive techniques typically address evasion or poisoning attacks separately, leaving models subject to combined threats. Our analysis shows that very few techniques can effectively mitigate both attacks at once. To close this gap, researchers should develop strong unified defenses to prevent both attack types, and this, in turn, makes the detection systems safer against real-world adversarial approaches.
- Unified defenses for evasion+poisoning, per the 2025 NIST taxonomy, which defines terminology for attacks across ML lifecycles [10].
- Bridge academic-real-world gaps with physical-layer attacks, informed by 2025 reviews on NIDS-specific adversarial impacts [14].

7. Conclusions

In this survey, we have investigated the ML techniques and their importance in NIDS, as well as a brief explanation of NIDSs and their correlation with ML, because NIDS are among the most crucial cybersecurity problems in the world's advanced technologies. The most important point is how attackers exploit the ML technique's weakness to launch attacks. Despite their superior efficiency and performance, ML-based NIDSs are subjected to small perturbations that are injected into legal traffic to trick the detection systems, resulting in disastrous repercussions for network security. AML remains a threat; 2025 advancements like enhanced ensemble frameworks for IDS robustness offer promise. Because this aspect of ML-NIDS has gotten so little attention, it needs to be addressed right away as NIDS needs more robust defense techniques against black-box adversarial attacks. This study provides an in-depth review on adversarial examples on NIDS based on ML. In addition, it lays out possible interesting future work.

Conflicts of Interest: No interest conflicts are defined by the authors.

References

1. S. Ahmad et al., "Novel approach using deep learning for intrusion detection and classification of the network traffic", IEEE CIVEMSA, 2020.
2. Alasad et al., "Resilient and secure hardware devices using ASL," ACM JETC, vol. 17, no. 2, 2021.
3. Q. Alasad, J.-S. Yuan, and P. Subramanyan," Strong logic obfuscation with low overhead against IC reverse engineering attacks," ACM TODAES, vol. 25, no. 4, 2020.
4. M. J. Hashemi and E. Keller "Enhancing robustness against adversarial examples in network intrusion detection systems," IEEE NFV-SDN, 2020.
5. S. Alahmed et al.," Mitigation of black-box attacks on intrusion detection systems-based ML, Computers," vol. 11, no. 7, 2022.
6. N. Aboueata et al., "Supervised machine learning techniques for efficient network intrusion detection," IEEE ICCCN, 2019.
7. A. Kumari and A. K. Mehta, "A hybrid intrusion detection system based on decision tree and support vector machine,"IEEE ICCCA, 2020.
8. G. Sah and S. Banerjee, Feature reduction and classification techniques for intrusion detection system, IEEE ICCSP, 2020.
9. B. M. Serinelli, A. Collen, and N. A. Nijdam, "Training guidance with kdd cup 1999 and nsl-kdd data sets of anidinr: Anomaly-based network intrusion detection system," Procedia Computer Science, vol. 175, pp. 560-565, 2020.

10. A. Vassilev et al., "Adversarial machine learning: A taxonomy and terminology of attacks and mitigations," NIST Trustworthy and Responsible AI Report, NIST AI 100-2e2025.
11. M. Usama et al., "Generative adversarial networks for launching and thwarting adversarial attacks on network intrusion detection systems," IEEE IWCMC, 2019.
12. J. Chen et al., "Fooling intrusion detection systems using adversarially autoencoder," Digit. Commun. Netw., vol. 7, no. 3, 2021.
13. H. A. Alatwi and A. Aldweesh, "Adversarial black-box attacks against network intrusion detection systems: A survey," in 2021 IEEE World AI IoT Congress (AIIoT), 2021: IEEE, pp. 0034-0040.
14. M. M. Hasan et al., "Adversarial attacks on deep learning-based network intrusion detection systems: A taxonomy and review," SSRN 5096420, 2025.
15. A. Alqahtani and H. AlShaher, "Anomaly-Based Intrusion Detection Systems Using Machine Learning," Journal of Cybersecurity & Information Management, vol. 14, no. 1, 2024.
16. H. A. Alatwi and C. Morisset, "Adversarial machine learning in network intrusion detection domain: A systematic review," arXiv preprint arXiv:2112.03315, 2021.
17. O. H. Abdulganiyu, T. A. Tchakoucht, and Y. K. Saheed, RETRACTED ARTICLE: Towards an efficient model for network intrusion detection system (IDS): systematic literature review, Wireless Netw., vol. 30, no. 1, 2024.
18. Yogesh and L. M. Goyal, RETRACTED ARTICLE: Deep learning based network intrusion detection system: a systematic literature review and future scopes, Int. J. Inf. Secur., vol. 23, no. 6, 2024.
19. A. Rawal, D. Rawat, and B. M. Sadler, "Recent advances in adversarial machine learning: status, challenges and perspectives, "AI & ML for Multi-Domain Ops Apps III, vol. 11746, 2021.
20. Z. Ahmad et al.," Network intrusion detection system: A systematic study of machine learning and deep learning approaches," Trans. Emerg. Telecommun. Techn., vol. 32, no. 1, 2021.
21. M. Ozkan-Okay et al.," A comprehensive systematic literature review on intrusion detection systems," IEEE Access, vol. 9, 2021.
22. I. Rosenberg et al., "Adversarial machine learning attacks and defense methods in the cyber security domain," ACM CSUR, vol. 54, no. 5, 2021.
23. H. Jmila and M. I. Khedher, "Adversarial machine learning for network intrusion detection: A comparative study," Computer Networks, vol. 214, p. 109073, 2022.
24. H. Khazane et al., "A holistic review of machine learning adversarial attacks in IoT networks," Future Internet, vol. 16, no. 1, 2024.
25. A. Alotaibi and M. A. Rassam, "Adversarial machine learning attacks against intrusion detection systems: A survey on strategies and defense," Future Internet, vol. 15, no. 2, p. 62, 2023.
26. W. Lim et al., "Future of generative adversarial networks (GAN) for anomaly detection in network security: A review," Comput. Secur., vol. 139, 2024.
27. Y. Pacheco and W. Sun, "Adversarial Machine Learning: A Comparative Study on Contemporary Intrusion Detection Datasets," in ICISSP, 2021, pp. 160-171.
28. K. He, D. D. Kim, and M. R. Asghar, "Adversarial machine learning for network intrusion detection systems: A comprehensive survey," IEEE Communications Surveys & Tutorials, vol. 25, no. 1, pp. 538-566, 2023.
29. O. Ibitoye, R. Abou-Khamis, M. e. Shehaby, A. Matrawy, and M. O. Shafiq, "The Threat of Adversarial Attacks on Machine Learning in Network Security--A Survey," arXiv preprint arXiv:1911.02621, 2019.
30. Q. Liu et al.," A survey on security threats and defensive techniques of machine learning: A data driven view," IEEE Access, vol. 6, 2018.
31. Y. Peng et al.," Detecting adversarial examples for network intrusion detection system with GAN," IEEE ICSESS, 2020.
32. G. Apruzzese et al., "Addressing adversarial attacks against security systems based on machine learning, "IEEE CyCon, vol. 900, 2019.
33. G. Apruzzese et al.," Modeling realistic adversarial attacks against network intrusion detection systems," DTRAP, vol. 3, no. 3, 2022.

34. E. Alhajjar, P. Maxwell, and N. Bastian, "Adversarial machine learning in network intrusion detection systems," Expert Systems with Applications, vol. 186, p. 115782, 2021.
35. M. A. Ayub et al., "Model evasion attack on intrusion detection systems using adversarial machine learning," IEEE CISS, 2020.
36. S. H. Silva and P. Najafirad, "Opportunities and challenges in deep learning adversarial robustness: A survey," arXiv preprint arXiv:2007.00753, 2020.
37. G. C. Marano et al., "Generative adversarial networks review in earthquake-related engineering fields," Bull. Earthq. Eng., vol. 22, no. 7, 2024.
38. S. Bourou et al.," A review of tabular data synthesis using GANs on an IDS dataset", Information, vol. 12, no. 9, 2021.
39. R. Soleymanzadeh and R. Kashef," Efficient intrusion detection using multi-player generative adversarial networks (GANs): an ensemble-based deep learning architecture," Neural Comput. Appl., vol. 35, no. 17, 2023.
40. I. K. Dutta et al., "Generative adversarial networks in security: A survey," IEEE UEMCON, 2020.
41. R. Chauhan and S. S. Heydari, "Polymorphic adversarial DDoS attack on IDS using GAN, "ISNCC, 2020.
42. P.-Y. Chen et al., "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," ACM AI & Security Workshop, 2017.
43. D. Golovin et al., "Gradientless descent: High-dimensional zeroth-order optimization, "arXiv:1911.06317, 2019.
44. S. Kumar, S. Gupta, and A. B. Buduru, "BB-Patch: BlackBox Adversarial Patch-Attack using Zeroth-Order Optimization," arXiv preprint arXiv:2405.06049, 2024.
45. X. Tian et al., "Dynamic geothermal resource assessment: Integrating reservoir simulation and Gaussian Kernel Density Estimation under geological uncertainties," Geothermics, vol. 120, 2024.
46. E. Aghaei and G. Serpen, "Host-based anomaly detection using Eigentraces feature extraction and one-class classification on system call trace data," arXiv preprint arXiv:1911.11284, 2019.
47. G. Pillonetto et al., "Deep networks for system identification: a survey," Automatica, vol. 171, 2025.
48. M. Ahsan et al.," Support vector data description with kernel density estimation (SVDD-KDE) control chart for network intrusion monitoring," Sci. Rep., vol. 13, no. 1, 2023.
49. Y.-C. Chen, "A tutorial on kernel density estimation and recent advances," Biostatistics & Epidemiology, vol. 1, no. 1, pp. 161-187, 2017.
50. S. Węglarczyk, "Kernel density estimation and its application," in ITM web of conferences, 2018, vol. 23: EDP Sciences, p. 00037.
51. D. V. Petrovsky et al., "PSSNet—An accurate super-secondary structure for protein segmentation," Int. J. Mol. Sci., vol. 23, no. 23, 2022.
52. S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2574-2582.
53. N. Fatehi, Q. Alasad, and M. Alawad, "Towards adversarial attacks for clinical document classification," Electronics, vol. 12, no. 1, p. 129, 2022.
54. Shone, N., et al., A deep learning approach to network intrusion detection. IEEE transactions on emerging topics in computational intelligence, 2018. 2(1): p. 41-50.
55. S. Alahmed et al.," Impacting robustness in deep learning-based NIDS through poisoning attacks," Algorithms, vol. 17, no. 4, 2024.
56. D. Jakubovitz and R. Giryes," Improving DNN robustness to adversarial attacks using Jacobian regularization,' ECCV, 2018.
57. I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," arXiv preprint arXiv:1412.6572, 2014.
58. N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in 2017 IEEE Symposium on security and privacy (sp), 2017: IEEE, pp. 39-57.
59. N. Papernot et al.," The limitations of deep learning in adversarial settings, "IEEE EuroS&P, 2016.
60. A. Madry et al., 'Towards deep learning models resistant to adversarial attacks," arXiv:1706.06083, 2017.

61. A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in Artificial intelligence safety and security: Chapman and Hall/CRC, 2018, pp. 99-112.
62. M. Ahmed et al.," Re-Evaluating Deep Learning Attacks and Defenses in Cybersecurity Systems, "Big Data Cogn. Comput., vol. 8, no. 12, 2024.
63. S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," Journal of Computational Science, vol. 25, pp. 152-160, 2018.
64. N. Moustafa and J. Slay," UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," MilCIS, 2015.
65. S. Alosaimi and S. M. Almutairi, "An intrusion detection system using BoT-IoT," Applied Sciences, vol. 13, no. 9, p. 5427, 2023.
66. J. Song et al., 'Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation," Workshop on Building Analysis Datasets, 2011.
67. P. Manisha and S. Gujar, "Generative Adversarial Networks (GANs): What it can generate and What it cannot?" arXiv preprint arXiv:1804.00140, 2018.
68. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," ICISSp, vol. 1, no. 2018, pp. 108-116, 2018.
69. L. Liu, P. Wang, J. Lin, and L. Liu, "Intrusion detection of imbalanced network traffic based on machine learning and deep learning," IEEE Access, vol. 9, pp. 7550-7563, 2020.
70. I. F. Kilincer, F. Ertam, and A. Sengur, "Machine learning methods for cyber security intrusion detection: Datasets and comparative study," Computer Networks, vol. 188, p. 107840, 2021.
71. I. Sharafaldin et al.," Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy, "IEEE ICCST, 2019
72. S. Rizvi et al.," Application of artificial intelligence to network forensics: Survey, challenges and future directions," IEEE Access,2022.
73. G. Singh and N. Khare, "A survey of intrusion detection from the perspective of intrusion datasets and machine learning techniques," Int. J. Comput. Appl., vol. 44, no. 7, 2022.
74. V. Rampure and A. Tiwari, "A rough set based feature selection on KDD CUP 99 data set," Int. J. Database Theory Appl., 2015.
75. A. Sharma and H. Babbar, "Detecting cyber threats in real-time: A supervised learning perspective on the CTU-13 dataset," IEEE INCET, 2024.
76. B. Yan and G. Han, "Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system," IEEE Access, vol. 6, pp. 41238-41248, 2018.
77. M. Yousefi-Azar et al., "Autoencoder-based feature learning for cyber security applications," IEEE IJCNN, 2017.
78. D. Han et al., "Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors," IEEE JSAC, vol. 39, no. 8, 2021.
79. M. Tavallaee et al., "A detailed analysis of the KDD CUP 99 data set," IEEE CISDA, 2009.
80. V. L. Thing, IEEE 802.11" network anomaly detection and attack classification: A deep learning approach," IEEE WCNC, 2017.
81. Y. Mirsky et al., "Kitsune: an ensemble of autoencoders for online network intrusion detection," arXiv:1802.09089, 2018.
82. N. Narodytska and S. P. Kasiviswanathan, "Simple Black-Box Adversarial Attacks on Deep Neural Networks," in CVPR workshops, 2017, vol. 2, no. 2.
83. S. Ghadimi and G. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," SIAM journal on optimization, vol. 23, no. 4, pp. 2341-2368, 2013.
84. A. Ilyas et al., "Black-box adversarial attacks with limited queries and information," ICML, 2018, Springer, pp. 261-273.
85. D. Wierstra et al.," Natural evolution strategies," arXiv:1106.4487, 2011.
86. P. Li et al., "Poisoning machine learning based wireless IDSs via stealing learning model," WASA, 2018.

87. X. Zhou et al., "Hierarchical adversarial attacks against graph-neural-network-based IoT network intrusion detection system," IEEE Internet of Things Journal, vol. 9, no. 12, pp. 9310-9319, 2021.
88. A. Hamza et al., "Detecting volumetric attacks on lot devices via sdn-based monitoring of mud activity," ACM Symposium on SDN Research, 2019, pp. 36-48.
89. K. Xu et al., "Representation learning on graphs with jumping knowledge networks," ICML, 2018: pmlr, pp. 5453-5462..
90. T. Kipf, "Semi-Supervised Classification with Graph Convolutional Networks," arXiv preprint arXiv:1609.02907, 2016.
91. X. Zhou et al.,"Academic influence aware and multidimensional network analysis for research collaboration navigation based on scholarly big data," IEEE TETC, 2018.
92. J. Ma, S. Ding, and Q. Mei, "Towards more practical adversarial attacks on graph neural networks," neurIPS, vol. 33, pp. 4756-4766, 2020.
93. Z. Sun et al., "In-memory PageRank accelerator with a cross-point array of resistive memories," IEEE TED, 2020.
94. J. Kotak and Y. Elovici, "Adversarial attacks against IoT identification systems,"IEEE Internet of Things Journal, vol. 10, no. 9, pp. 7868-7883, 2022.
95. A. Sivanathan et al., "Classifying IoT devices in smart environments using network traffic characteristics," IEEE TMC, 2018.
96. J. Tian, "Adversarial vulnerability of deep neural network-based gait event detection: A comparative study using accelerometer-based data," BSPC, vol. 73, p. 103429, 2022.
97. A. Kuppa et al.,"Black box attacks on deep anomaly detectors," in Proceedings of the 14th international conference on availability, reliability and security, 2019, pp. 1-10.
98. F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," IEEE ICDM, 2008.
99. B. Zong et al., "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," ICLR, 2018.
100. H. Zenati et al., "Adversarially learned anomaly detection," IEEE ICDM, 2018.
101. B. Schölkopf et al., "Support vector method for novelty detection," neurIPS, 1999.
102. T. Schlegl et al., "f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks [J]," Medical image analysis, vol. 54, pp. 30-44, 2019.
103. J. Aiken and S. Scott-Hayward, "Investigating adversarial attacks against network intrusion detection systems in sdns,"IEEE NFV-SDN, 2019.
104. Q. Yan et al., "Automatically synthesizing DoS attack traces using generative adversarial networks," IJMLC, 2019.
105. D. Shu et al., "Generative adversarial attacks against intrusion detection systems using active learning," in Proceedings of the 2nd ACM workshop on wireless security and machine learning, 2020.
106. S. Guo et al., "A Black-Box Attack Method against Machine-Learning-Based Anomaly Network Flow Detection Models," Security and Communication Networks, vol. 2021, no. 1, p. 5578335, 2021.
107. Y. Sharon et al, "Tantra: Timing-based adversarial network traffic reshaping attack," IEEE TIFS, 2022.
108. B.-E. Zolbayar et al., "Generating practical adversarial network traffic flows using NIDSGAN," arXiv preprint arXiv:2203.06694, 2022.
109. T. Hou,"IoTGAN: GAN powered camouflage against machine learning based IoT device identification," IEEE DySPAN, 2021.
110. J. Bao, B. Hamdaoui, and W.-K. Wong, "Iot device type identification using hybrid deep learning approach for increased iot security," IEEE IWCMC, 2020.
111. S. Aldhaheri and A. Alhuzali, "SGAN-IDS: Self-attention-based generative adversarial network against intrusion detection systems," Sensors, vol. 23, no. 18, p. 7796, 2023.
112. M. Fan et al., "Toward Evaluating the Reliability of Deep-Neural-Network-Based IoT Devices," IEEE Internet of Things Journal, vol. 9, no. 18, pp. 17002-17013, 2021.
113. E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," arXiv preprint arXiv:2001.03994, 2020.

114. A. Krizhevsky, V. Nair, and G. Hinton, "The CIFAR-10 and CIFAR-100 datasets, 2014," URL https://www. cs. toronto. edu/~ kriz/cifar. html.
115. M. Usama, J. Qadir, A. Al-Fuqaha, and M. Hamdi, "The adversarial machine learning conundrum: can the insecurity of ML become the achilles' heel of cognitive networks?" IEEE Network, vol. 34, no. 1, pp. 196-203, 2019.
116. A. Abusnaina, A. Khormali, D. Nyang, M. Yuksel, and A. Mohaisen, "Examining the robustness of learning-based ddos detection in software defined networks," IEEE DSC, 2019.
117. M. J. Hashemi, G. Cusack, and E. Keller, "Towards evaluation of nidss in adversarial setting," in Proceedings of the 3rd ACM CoNEXT Workshop on Big DATA, Machine Learning and Artificial Intelligence for Data Communication Networks, 2019, pp. 14-21.
118. H. Zenati et al., "Efficient GAN-based anomaly detection," arXiv preprint arXiv:1802.06222, 2018.
119. I. Homoliak et al., "Improving network intrusion detection classifiers by non-payload-based exploit-independent obfuscations: An adversarial approach," arXiv preprint arXiv:1805.02684, 2018.
120. M. Teuffenbach, E. Piatkowska, and P. Smith, "Subverting network intrusion detection: Crafting adversarial examples accounting for domain-specific constraints," in International Cross-Domain Conference for Machine Learning and Knowledge Extraction, 2020: Springer, pp. 301-320.
121. Y. Wang et al., "A c-ifgsm based adversarial approach for deep learning based intrusion detection,"VECoS, 2020: Springer, pp. 207-221.
122. E. Anthi, "Hardening machine learning denial of service (DoS) defences against adversarial attacks in IoT smart home networks," computers & security, vol. 108, p. 102352, 2021.
123. P. M. S. Sánchez et al., "Adversarial attacks and defenses on ML-and hardware-based IoT device fingerprinting and identification," Future Generation Computer Systems, vol. 152, pp. 30-42, 2024.
124. P. M. S. Sanchez et al., "LwHBench: A low-level hardware component benchmark and dataset for Single Board Computers," arXiv preprint arXiv:2204.08516, 2022.
125. K. Roshan, A. Zafar, and S. B. U. Haque, "Untargeted white-box adversarial attack with heuristic defence methods in real-time deep learning based network intrusion detection system," Computer Communications, vol. 218, pp. 97-113, 2024.
126. H. Xiao et al., "Is feature selection secure against training data poisoning?" in international conference on machine learning, 2015: PMLR, pp. 1689-1698.
127. B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," arXiv preprint arXiv:1206.6389, 2012.
128. K. Yang et al.,"Adversarial examples against the deep learning based network intrusion detection systems," IEEE MILCOM, 2018.
129. X. Peng, W. Huang, and Z. Shi, "Adversarial attack against dos intrusion detection: An improved boundary-based method," IEEE ICTAI, 2019.
130. J. Lu, T. Issaranon, and D. Forsyth, "Safetynet: Detecting and rejecting adversarial examples robustly," IEEE ICCV, 2017.
131. J. H. Metzen et al., "On detecting adversarial perturbations," arXiv preprint arXiv:1702.04267, 2017.
132. M. Barreno et al., "The security of machine learning," Machine learning, vol. 81, no. 2, pp. 121-148, 2010.
133. X. Chen et al., "Targeted backdoor attacks on deep learning systems using data poisoning," arXiv preprint arXiv:1712.05526, 2017.
134. J. Wang et al., "Adversarial sample detection for deep neural network through model mutation testing," IEEE ICSE, 2019.
135. A. Raghuvanshi et al., "Intrusion detection using machine learning for risk mitigation in IoT-enabled smart irrigation in smart farming," Journal of Food Quality, vol. 2022, no. 1, p. 3955514, 2022.
136. C. Benzaïd, M. Boukhalfa, and T. Taleb, "Robust self-protection against application-layer (D) DoS attacks in SDN environment," IEEE WCNC, 2020.
137. H. Benaddi et al., "Anomaly detection in industrial IoT using distributional reinforcement learning and generative adversarial networks," Sensors, vol. 22, no. 21, p. 8085, 2022.
138. G. Li et al., "DeSVig: Decentralized swift vigilance against adversarial attacks in industrial artificial intelligence systems," IEEE Transactions on Industrial Informatics, vol. 16, no. 5, pp. 3267-3277, 2019.

139. M. Mirza and S. Osindero, "Conditional generative adversarial nets," arXiv preprint arXiv:1411.1784, 2014.
140. H. Benaddi et al., "Adversarial attacks against iot networks using conditional gan based learning," IEEE GLOBECOM 2022, pp. 2788-2793.
141. A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in International conference on machine learning, 2017: PMLR, pp. 2642-2651.
142. G. S. Dhillon et al., "Stochastic activation pruning for robust adversarial defense," arXiv preprint arXiv:1803.01442, 2018.
143. R. Abou Khamis, M. O. Shafiq, and A. Matrawy, "Investigating resistance of deep learning-based ids against adversaries using min-max optimization," IEEE ICC, 2020.
144. M. Pawlicki, M. Choraś, and R. Kozik, "Defending network intrusion detection systems against adversarial evasion attacks," Future Generation Computer Systems, vol. 110, pp. 148-154, 2020.
145. A. Ganesan and K. Sarac, "Mitigating evasion attacks on machine learning based nids systems in sdn," IEEE NetSoft, 2021.
146. I. Debicha et al., "Detect & reject for transferability of black-box adversarial attacks against network intrusion detection systems," ACeS , 2021: Springer, pp. 329-339.
147. M. P. Novaes et al., "Adversarial Deep Learning approach detection and defense against DDoS attacks in SDN environments," Future Generation Computer Systems, vol. 125, pp. 156-167, 2021.
148. R. Yumlembam et al., "Iot-based android malware detection using graph neural network with adversarial defense," IEEE Internet of Things Journal, vol. 10, no. 10, pp. 8432-8444, 2022.
149. M. K. Roshan and A. Zafar, "Boosting robustness of network intrusion detection systems: A novel two phase defense strategy against untargeted white-box optimization adversarial attack," Expert Systems with Applications, vol. 249, p. 123567, 2024.
150. Z. Awad, M. Zakaria, and R. Hassan, "An enhanced ensemble defense framework for boosting adversarial robustness of intrusion detection systems," Scientific Reports, vol. 15, no. 1, p. 14177, 2025.
151. B. Nelson et al., "Misleading learners: Co-opting your spam filter," in Machine learning in cyber trust: Security, privacy, and reliability: Springer, 2009, pp. 17-51.
152. G. Apruzzese et al.," AppCon: Mitigating evasion attacks to ML cyber detectors," Symmetry, vol. 12, no. 4, p. 653, 2020.
153. G. Apruzzese and M. Colajanni, "Evading botnet detectors based on flows and random forest with adversarial samples," IEEE NCA, 2018.
154. M. Siganos et al., "Explainable ai-based intrusion detection in the internet of things," in Proceedings of the 18th international conference on availability, reliability and security, 2023, pp. 1-10.
155. C. Park et al., "An enhanced AI-based network intrusion detection system using generative adversarial networks," IEEE Internet of Things Journal, vol. 10, no. 3, pp. 2330-2345, 2022.
156. M. A. Sen, "Attention-GAN for anomaly detection: A cutting-edge approach to cybersecurity threat management," arXiv preprint arXiv:2402.15945, 2024.
157. H. Larijani et al., "An adversarial attack detection paradigm with swarm optimization," IEEE IJCNN, 2020.
158. S. Waskle, L. Parashar, and U. Singh, "Intrusion detection system using PCA with random forest approach," IEEE ICESC, 2020.
159. A. H. Mirza, "Computer network intrusion detection using various classifiers and ensemble learning," IEEE SIU, 2018.
160. Q. R. S. Fitni and K. Ramli, "Implementation of ensemble learning and feature selection for performance improvements in anomaly-based intrusion detection systems," IEEE IAICT, 2020.
161. P. Li et al., "Chronic poisoning against machine learning based IDSs using edge pattern detection," IEEE ICC, 2018.
162. M. Usama et al.,"Black-box adversarial machine learning attack on network traffic classification," IEEE IWCMC, 2019.
163. A. Warzyński and G. Kołaczek, "Intrusion detection systems vulnerability on adversarial examples," IEEE INISTA, 2018.
164. S. Zhao et al., "attackgan: Adversarial attack against black-box ids using generative adversarial networks," Procedia Computer Science, vol. 187, pp. 128-133, 2021.

165. Z. Lin, Y. Shi, and Z. Xue, "Idsgan: Generative adversarial networks for attack generation against intrusion detection," KDDM, 2022: Springer, pp. 79-91.
166. A. Punitha, S. Vinodha, R. Karthika, and R. Deepika, "A feature reduction intrusion detection system using genetic algorithm," IEEE ICSCAN, 2019.
167. Q. Alasad, M. M. Hammood, and S. Alahmed, "Performance and Complexity Tradeoffs of Feature Selection on Intrusion Detection System-Based Neural Network Classification with High-Dimensional Dataset," ICETIS, 2022: Springer, pp. 533-542.