future internet

MDPI

Article

Securing Network Traffic Classification Models against Adversarial Examples Using Derived Variables

James Msughter Adeke [1,2], Guangjie Liu [1,2,*], Junjie Zhao [1,2], Nannan Wu [3] and Hafsat Muhammad Bashir [1]

[1] School of Electronics & Information Engineering, Nanjing University of Information Science & Technology, Nanjing 210044, China; adekejames@nuist.edu.cn (J.M.A.); ginogogo@nuist.edu.cn (J.Z.); hafsatmbashir@nuist.edu.cn (H.M.B.)
[2] Key Laboratory of Intelligent Support Technology for Complex Environments, Ministry of Education, Nanjing 210044, China
[3] School of Computer & Software, Nanjing University of Information Science & Technology, Nanjing 210044, China; wu_nannan@nuist.edu.cn
* Correspondence: gjieliu@nuist.edu.cn; Tel.: +86-138-1304-9821

Abstract: Machine learning (ML) models are essential to securing communication networks. However, these models are vulnerable to adversarial examples (AEs), in which malicious inputs are modified by adversaries to produce the desired output. Adversarial training is an effective defense method against such attacks but relies on access to a substantial number of AEs, a prerequisite that entails significant computational resources and the inherent limitation of poor performance on clean data. To address these problems, this study proposes a novel approach to improve the robustness of ML-based network traffic classification models by integrating derived variables (DVars) into training. Unlike adversarial training, our approach focuses on enhancing training using DVars, introducing randomness into the input data. DVars are generated from the baseline dataset and significantly improve the resilience of the model to AEs. To evaluate the effectiveness of DVars, experiments were conducted using the CSE-CIC-IDS2018 dataset and three state-of-the-art ML-based models: decision tree (DT), random forest (RF), and k-neighbors (KNN). The results show that DVars can improve the accuracy of KNN under attack from 0.45% to 0.84% for low-intensity attacks and from 0.32% to 0.66% for high-intensity attacks. Furthermore, both DT and RF achieve a significant increase in accuracy when subjected to attack of different intensity. Moreover, DVars are computationally efficient, scalable, and do not require access to AEs.

Keywords: machine learning; adversarial attack; network traffic classification; derived variables; robustness

Citation: Adeke, J.M.; Liu, G.; Zhao, J.; Wu, N.; Bashir, H.M. Securing Network Traffic Classification Models against Adversarial Examples Using Derived Variables. *Future Internet* **2023**, *15*, 405. https://doi.org/10.3390/fi15120405

Academic Editor: Franco Davoli

Received: 13 November 2023
Revised: 4 December 2023
Accepted: 14 December 2023
Published: 16 December 2023

1. Introduction

Machine learning (ML)-based models are still widely used in cybersecurity, including network traffic classification and intrusion detection, because of their maturity and ease of implementation [1]. Moreover, they remain mainstream in real-time network traffic analysis, particularly in high-throughput networks [2]. However, despite the widespread use, these models are vulnerable to adversarial attacks (an attack is a method of crafting an adversarial example, AE for short), including poisoning, extraction, evasion, and inference, which can compromise their efficiency and cause significant network damage [3–6]. An adversarial attack is a type of cyberattack in which an attacker intentionally modifies the input data to an ML model to produce the desired result [7]. The discovery of this vulnerability can be traced back to the field of computer vision, where researchers first demonstrated that deep neural networks (DNNs), which were thought to be robust and accurate, could easily be deceived by maliciously crafted inputs [8]. They demonstrated that a small and well-computed perturbation of a few input features (i.e., the pixel values of an image) can deceive an image recognition system, causing the misclassification of objects

or scenes with a high degree of confidence (Figure 1). However, adversarial attacks are not limited to computer vision: this is also an evolving concern in other fields of ML, including cybersecurity [9].

ML-based network traffic classification systems are particularly vulnerable to these attacks because they are often used in security-critical applications [10–12]. The infiltration of AEs into network traffic data during the training or testing phase can compromise the efficiency of ML-based network security systems (Figure 2), potentially causing significant network damage. Attackers can achieve this by using several methods, including altering packet rates and payload sizes, modifying packet headers, and using evasion techniques to avoid detection by network security systems [13–15]. The growing concern surrounding adversarial attacks has led to an increased interest in adversarial defense [16,17]. Adversarial training, a technique that involves the retraining of ML models with AEs, is widely recognized as the state-of-the-art defense because of its efficacy [18]. However, it is not without significant limitations. These limitations encompass the need for access to a substantial number of AEs, a prerequisite that entails significant computational resources, and the inherent limitation of poor performance on clean data (non-adversarial data). Models trained with adversarial training may exhibit suboptimal performance when presented with non-adversarial data. In response to these limitations, alternative defenses have garnered attention, including anomaly detection [19] and ensemble methods [20], each of which is characterized by a distinctive set of strengths and weaknesses. Numerous other defensive approaches have been primarily validated in the domain of computer vision, where adversarial attacks were initially discovered. Furthermore, these defensive strategies have been predominantly applied in deep learning models, with limited exploration in shallow models, which are particularly relevant in real-time network traffic analysis. The significance of this gap becomes evident when considering the relevance of shallow machine learning models in real-time network traffic analysis. Consequently, a pressing imperative emerges for the advancement and implementation of defense mechanisms specifically tailored to secure shallow ML models in such critical applications.

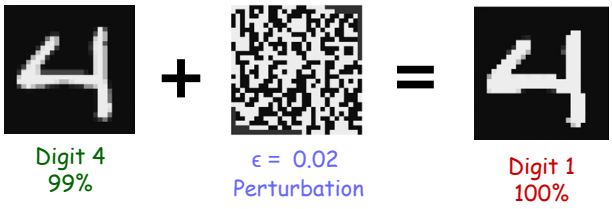


Figure 1. Adversarial attack in computer vision [21].

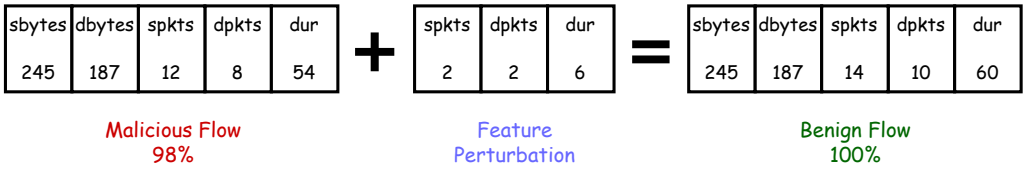


Figure 2. Adversarial attack on network traffic classification systems [22].

This study is particularly relevant in the dynamic field of network security, where the evolving threat of adversarial attacks is of paramount concern. In contrast to conventional retraining with AEs, the introduction of derived variables (DVars) effectively bridges the recognized gap in the existing literature and provides valuable insights to improve the resilience of ML-based network traffic classifiers. DVars, an automated approach designed to enhance the robustness of ML-based network traffic classification systems, introduces controlled randomness into the data, thus enhancing its diversity. The core of this research focuses on the improvement of three state-of-the-art shallow ML-based network traffic classifiers: decision tree (DT), random forest (RF), and k-neighbors (KNN) in response to adversarial attacks, with a specific focus on countering the Jacobian-based Saliency

Map Attack (JSMA). While the primary intent of the JSMA was originally designed to be attacking image classification systems, its perturbation crafting procedures could be used to preserve the underlying logic of a network traffic flow [22]. The JSMA only perturbs the most appropriate features in the decision boundary without affecting the remaining features, which can preserve the correlations among most features of a network flow; hence, this underscores the strategic motivation behind its adoption. Our findings show the profound impact of adversarial attacks on these models and underscore the remarkable efficacy of DVars in restoring their performance in the presence of black-box attacks.

The main contributions of the work presented in this paper are as follows:

- A method that improves the robustness of ML-based network traffic classification models against adversarial attack is proposed by integrating DVars into training. The DVars follows the logic of adding randomness to the input data. In particular, our proposed approach preserves the underlying logic and maliciousness of the network flow.
- Evaluation of the proposed method on the CSE-CIC-IDS-2018 dataset, with a specific focus on improving the accuracy of network traffic classification models when subjected to AEs. According to experimentation and analysis, our approach shows considerable improvements in the performance of the models.
- Investigation of the impact of AEs on ML-based network traffic classification models. Using experiments and analysis, we explore the effect of AEs on the performance and robustness of the investigated models.

The remainder of this paper is organized as follows: Section 2 provides a detailed review of related studies on ML-based network traffic classification, common AE generation methods, and defenses. Section 3 presents the impact of AEs on ML-based network traffic classification models, and Section 4 presents our novel defensive method based on DVars. Section 5 presents the dataset and evaluation metrics used in assessing the performance of DVars. Section 6 provides the results and analysis, and Section 7 concludes the work and discusses future directions.

2. Related Work

In this section, we provide a detailed review of related works on ML-based network traffic classification, common AE generation, and defense methods against AEs, with focus on cybersecurity systems.

2.1. ML-Based Network Traffic Classification

ML models such as DT, RF, and KNN have been explored for network traffic classification due to their simplicity, interpretability, and efficiency. Several studies have investigated the use of ML models for network traffic classification. For example, Mohanty et al. [23] proposed a robust stacking ensemble model to combine the predictions of RF, KNN, and DT for darknet traffic classification, achieving accuracy rates of 98.89% and 97.88% for darknet traffic identification and characterization, respectively. Zaki et al. [24] proposed a hybrid feature selection algorithm based on the filter and wrapper method. Their method was evaluated using DT, KNN, naive Bayes (NB), and support vector machine (SVM), with average accuracy of 98.90%. Cao et al. [25] developed an improved network traffic classification model based on SVM, achieving higher classification accuracy, 97.20%. Bhatia et al. [26] proposed a DT-based multi-level P2P traffic classification technique based on packet and flow characteristics, achieving a combined accuracy rate of 98.30%. Dey et al. [27] compared the performance of several ML models, including DT, RF, and artificial neural networks (ANNs), to classify network traffic and found that DT, RF, and ANNs achieved the best accuracy.

Despite this tremendous success, ML models are vulnerable to adversarial attacks, which can manipulate input data to mislead classifiers and cause misclassification. Adversarial attacks on ML models have become a significant concern in the cybersecurity community because they can lead to serious consequences in real-world scenarios, such as

malicious traffic bypassing intrusion detection systems. Various studies have investigated the susceptibility of ML models to adversarial attacks in the context of network traffic classification. Rust-Nguyen et al. [28] demonstrated that RF is vulnerable to adversarial attacks on darknet traffic classification. They proposed a defense mechanism based on an encoding scheme to transform class features using probability analysis to effectively deal with such adversarial attacks. Other studies have explored different adversarial attack techniques and defense mechanisms for ML models in network traffic classification. For example, Lin et al. [29] proposed an adversarial example generation method based on the Wasserstein distance to evade DT classifiers for intrusion detection. Alhajjar et al. [30] proposed an adversarial example generation method using evolutionary computation, i.e., particle swarm optimization (PSO) and genetic algorithms (GAs), along with a generative adversarial network (GAN) to fool eleven different ML-based network traffic classifiers, including DT, RF, and KNN. In the defense race, Asadi et al. [31] proposed a method to detect botnets using the PSO algorithm based on the voting system. Moreover, some studies have investigated the impact of adversarial attacks on the interpretability and explainability of ML network traffic classification models. For example, Capuano et al. [32], in their survey, demonstrated that adversarial attacks can cause DT to produce counterintuitive and misleading explanations for network traffic classification. They proposed a defense mechanism based on model distillation and gradient regularization to improve the interpretability and robustness of the decision trees. In general, ML algorithms have shown promising results in the classification of network traffic. However, their vulnerability to adversarial attacks is a significant challenge that must be addressed. Further research is required to develop more robust and secure ML models for network traffic classification.

2.2. Common AE Generation Methods

AEs are specifically designed inputs that can cause ML models to classify the output incorrectly. In the context of network traffic classification, AEs can be used to evade detection or generate false positives, leading to the inefficient use of resources and potential security breaches. An AE can be generated as shown in Equation (1).

$$\begin{aligned} & x' = x + \epsilon \\ & \text{subject to} \quad f(x') \neq f(x) \end{aligned} \tag{1}$$

where x is the original input to the model, $f(x)$ is the output of the model for that input, x' is the perturbed input, and ϵ is the small perturbation added to x to generate x'. Perturbation ϵ is chosen to be sufficiently small to be imperceptible to humans but sufficiently large to cause the model to make an incorrect prediction.

AEs can be generated using various techniques, such as gradient-based methods, evolutionary algorithms, and black-box attacks. Gradient-based methods, in which the gradient of the loss function with respect to the input data is used to iteratively modify the input until the desired misclassification is achieved, are the most commonly used techniques for generating AEs. For example, in [33], a gradient-based method was proposed to generate AEs that can evade detection using a well-trained network traffic classification model. Evolutionary algorithms such as GAs and PSO can also be used to generate AEs. These algorithms search for the optimal modification of the input that maximizes the adversarial objective function. For example, in [34], a GA was used to generate AEs that could evade detection by an SVM-based network traffic classification system. A black-box attack is a more challenging type of attack where the attacker has limited access to the target model and cannot directly compute the gradients. These attacks require the use of transferability and model inversion techniques to generate AEs. For example, Usama et al. [35] proposed a black-box attack technique to generate AEs. They exploited the statistical properties of the features to calculate the mutual information between each feature and true label. Their proposed scheme evaded detection even when the attacker had no knowledge of the target model.

Several gradient-based methods have been proposed for generating AEs, including Fast Gradient Sign Method (FGSM), Jacobian-based Saliency Map Attack (JSMA), Project Gradient Descent (PGD), Basic Iterative Method (BIM), etc. These methods aim to determine the optimal perturbation of the input data that can cause the maximum misclassification error of the model while remaining imperceptible to humans. In this study, the designed defense method is evaluated using the JSMA. The reason for this choice is that this attack manipulates the features that have the greatest effect on output classification [36]; therefore, it represents a real-life scenario among others. However, for clarity, we further explain in detail the underlying logic of the most influential gradient-based methods for generating AEs (also designated as adversarial attacks).

2.2.1. Fast Gradient Sign Method (FGSM) Attack

The FGSM attack, proposed by Goodfellow et al. [37], is an optimal method of max norm constraint to craft perturbations of the original input. This attack fools the ML/DNN model to misclassify the input generated by increasing the direction of the gradient based on the gradient descent principle. The FGSM can be formulated as shown in Equation (2).

$$x' = x + \epsilon * sign(\bigtriangledown x J(\theta, x, y)) \tag{2}$$

This equation computes the perturbed input, x', which is the original input (x) supplemented with a small perturbation (ϵ) that is proportional to the sign of the gradient of the model's loss function ($J(\theta, x, y)$) with respect to the input features (x), where θ is the model parameter and y is the true label of the data point. The sign function ensures that the perturbation is in the direction that maximizes the loss function, whereas the epsilon value controls the intensity of the perturbation. The FGSM method is computationally efficient and can be used to quickly generate AEs. However, it may not always be effective in generating strong AEs and can be easily defended from methods such as adversarial training.

2.2.2. Project Gradient Descent (PGD) Attack

Madry et al. [38] proposed the PGD attack to improve the robustness of neural network models with adversarial training. This is an advanced iterative attack method used to evaluate the robustness of ML models against adversarial attacks. The PGD attack involves iteratively perturbing the input data in small steps in the direction of the gradient of the loss function with respect to the input while projecting the perturbed data back onto a restricted set of allowed values. The projection step ensures that the perturbed data remain within a bounded region of the input space, which is typically defined by the maximum permissible perturbation or range of the allowed input values. The PGD attack can be customized by varying the number of iterations, the step size, and the projection function. A higher number of iterations and smaller step sizes can result in stronger attacks but may also require more computation time. The projection function can also affect the effectiveness of an attack, with different projection functions leading to different levels of success in generating AEs. The PGD attack is formulated as shown in Equation (3).

$$x_{t+1} = \Pi_{x+s}\left(x^t + \alpha sign(\bigtriangledown x L(\Theta, x, y))\right) \tag{3}$$

where $\Pi(.)$ is the projection operator, t denotes the current step, α represents the step size, and ϵ is the magnitude of the perturbation. This attack is considered to be one of the strongest and most effective attacks in adversarial ML.

2.2.3. Jacobian-Based Saliency Map Attack (JSMA)

The JSMA was proposed by Papernot et al. [39] to find the optimal perturbation to create an AE by taking forward derivatives of the classifier model. It is a targeted attack that aims to modify the input data in a manner that causes the model to misclassify the input as a specific target class. The JSMA attack works by computing the saliency map of the input

data, which indicates the importance of each input feature for the model's decision. The saliency map is computed using the Jacobian matrix, which describes the rate of change of the model outputs with respect to the input features. The saliency map is then used to identify the most important input features that could be modified to cause the model to misclassify the input as the target class. The JSMA is a white-box attack, which means that it requires access to the internal parameters of the ML model being attacked. It is typically performed by iteratively modifying the input features that have the highest saliency values to maximize the likelihood that the model misclassifies the input as the target class. The modified input is then checked to determine if it has the desired misclassification, and the process is repeated until a successful AE is found.

Given an input data point x, an ML model with parameters θ, a target class t, and a perturbation budget ϵ, the JSMA seeks to find a perturbed data point x' that maximizes the probability of the model predicting the target class (t), subject to the constraint that the perturbations are bounded by ϵ:

$$\begin{aligned} &argmax'_x P(y = t|x') \\ &\text{subject to} \quad ||x' - x||_p \leq \epsilon \end{aligned} \tag{4}$$

where $P(y = t|x')$ is the probability that the model predicts the target class (t) for perturbed data points x' and $||.||_p$ denotes the L_p norm that measures the distance between x and x'.

To compute the perturbed data point x', the JSMA uses the Jacobian matrix ($J(\theta, x)$), which describes the rate of change in the output of the model with respect to the input features. The saliency score of each input feature (i) is calculated as shown in Equation (5).

$$S_i(x) = max_{j \neq t}[(J(\theta, x)_j, i * (f_t(x') - f_j(x')))] \tag{5}$$

where $J(\theta, x)_j$, i is the $(j, i)th$ element of the Jacobian matrix, $f_t(x')$ is the output of the model for the target class (t), and $f_j(x')$ is the output of the model for class j.

The saliency scores are then used to identify the k most salient input features for modification. The modification to each input feature i is given by Equation (6).

$$x'_i = x_i + \epsilon_i \tag{6}$$

where ϵ_i is the smallest perturbation that increases the saliency score of feature i while maintaining the perturbation within the budget (ϵ). The JSMA iteratively modifies the k most salient input features until the model predicts the target class (t) for the perturbed data point (x'). The attack can terminate after a specified number of iterations or when the model predicts the target class with a specified probability threshold. The JSMA is effective in generating AEs with high success rates, but it requires access to the model's parameters and outputs, which may not always be available in real-world scenarios.

2.3. Common Defense Methods against AEs

Several defense methods have been proposed to mitigate the impact of AEs on ML-based network traffic classifiers. These defense methods can be broadly categorized into two types [40]: reactive and proactive. On the one hand, reactive defense methods aim to detect and reject AEs during classification. These methods can be based on various preprocessing techniques, such as feature squeezing and input normalization, to narrow the search space for an adversary and postprocessing techniques, such as mechanisms that deal with model uncertainty and require predictions with high confidence scores [41,42]. Although these defense methods are effective, they often suffer from high false-positive rates and require significant computational resources. On the other hand, proactive defense methods aim to improve the robustness of ML models against AEs by modifying the training process or model architecture. These methods are based on techniques such as adversarial training [43], regularization [44] to better calibrate the learning process, and defensive distillation to create smaller models that are less sensitive to data variations [45].

Adversarial training is a technique in which a model is trained on both clean data and AE to improve its robustness against AEs. For example, in [46,47], the authors proposed adversarial training as a defense method against adversarial attacks on ML-based network security systems that achieved significant improvements in terms of robustness against AEs. However, adversarial training is susceptible to transferability attacks, where the AEs generated for one model can also fool other models trained using similar techniques. Model-based defense aims to improve the robustness of a classification model by modifying its architecture or training process. Examples of model-based defense include defensive distillation, ensemble methods, and regularization techniques. Input preprocessing techniques aim to render the ML model more resilient to AEs by introducing random noise or perturbations into the input data. Despite the effectiveness of these defense methods, recent studies have shown that they are not foolproof and can be circumvented by more sophisticated AEs. Therefore, the development of more robust and reliable defense methods against AEs remains an active area of research in the field of cybersecurity.

In this paper, we propose a new defense method to enhance the robustness of network traffic classification models against AEs. The proposed method involves the derivation of additional variables from the baseline dataset. These derived variables are used together with the baseline dataset to train a more resilient model that is robust to AE. Specifically, we first identify features in the dataset that are likely to be targeted by the adversary and then use these features to derive distinct variables. By incorporating these derived variables into training, a more robust classifier can be trained. The proposed method offers several advantages over existing defense methods. First, it does not require the generation of AEs during training, making it less vulnerable to transferability attacks. Second, it does not rely on AE detection, thus reducing the risk of false positives. Third, it is computationally efficient and can be easily integrated into existing network traffic classification models. In general, the proposed method is a promising solution to improve the robustness of network traffic classification models against adversarial attacks.

3. AE Impact on Network Traffic Classifiers

We begin by developing well-trained ML-based models for network traffic classification to study the impact of adversarial attacks on these models. For development, we employed the Python programming language, as well as the popular ML libraries Keras [48] and scikitlearn [49]. Additionally, Adversarial Robustness Toolbox (ART) [50] is used to guide the AE generation process.

3.1. The Target Models

As previously stated, our goal is to secure ML-based network traffic classification models against adversarial attacks. Figure 3 provides an overview of the procedures involved in ML-based network traffic classification, also employed by the target models. We investigated three state-of-the-art ML-based network traffic classification models: DT, RF, and KNN. Throughout this study, these models will be referred to as target models. First, we used scikit-learn to build a multi-class classifier for each of these models. We preprocessed, scaled, and split the re-sampled dataset into 70:30 training and test samples, respectively. The accuracy, precision, recall, and F1-score of the classifiers were used to evaluate their performance.

3.2. Generating AEs for Non-Gradient-Based Models

Due to their simplicity, interpretability, and efficiency, non-gradient-based models such as the target models employed in this study are often used for classifying network traffic. However, these models cannot be directly used to craft AEs using gradient-based attacks, as they do not have a continuous and differentiable loss function. To overcome this limitation, we developed a neural network model called AdverNet trained on the baseline dataset using the JSMA technique to generate AEs that can be transferred to non-gradient-based models to evaluate their robustness against adversarial attacks. To use AdverNet to

attack non-gradient-based models, we first generate AEs for AdverNet and then exploit the transferability property of AEs to attack the target models. This transferability property allows the AEs generated for one model to be effective in fooling other models even if they are trained on different datasets or using different architectures [51]. Because the target models and AdverNet are known to be similar and trained with portions of the same dataset, the approach is intended to be illustrative of black-box attacks. Nevertheless, because we inevitably have some knowledge of the underlying models, this may well be called a gray-box attack. AdverNet was trained using the same procedures and architecture as in McCarthy et al. [33], except that 50 training epochs were used to train AdverNet, whereas 300 epochs were used in their study, with an early stoppage time of 100 epochs.

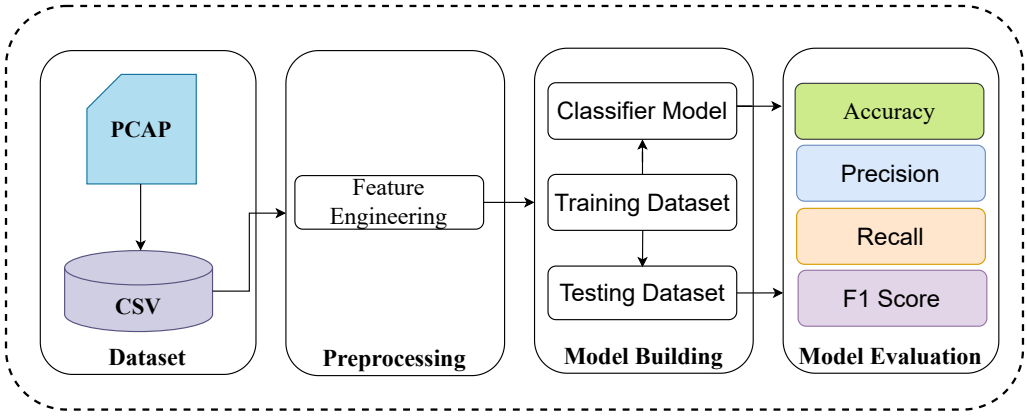


Figure 3. An overview of ML-based network traffic classification.

3.3. Domain Constraints in AE Generation

The feasibility of adversarial attacks differs depending on the domain, as it is highly restricted by several constraints, which can be divided into two distinct groups [52]: syntactic constraints and semantic constraints. All the constraints related to syntax are often referred to as syntactic constraints. Merzouk et al. [53] stated three syntactic constraints that an AE must satisfy: out-of-range values, non-binary values, and multi-category membership. The values considered out of range are those that are higher than the theoretical maximum value that cannot be exceeded. Nonbinary values are entries that invalidate a feature's binary nature, whereas multi-category membership values cannot be one-hot encoded.

However, semantic links represent the relationships that distinct features may have with each other. Teuffenbach et al. [54] suggested an obvious method for NIDS by splitting the features into roughly three distinct groups with various semantic relationships. The first group comprises features that can be directly altered by the adversary (e.g., number of forward packets, size of the forward packets, and flow duration). The second group of features is dependent on the first feature and is updated with respect to the latest feature (e.g., number of packets/second, average forward packet size). The last group of features comprises elements that the adversary cannot change (e.g., IP address and protocol number).

3.4. Using AdverNet to Attack Target Models

As an adversary, the objective is to misclassify the output of target models. Adversarial attacks are generally classified into three categories: white box, black box, and gray box. White-box attacks involve complete knowledge of the target model and direct access to its parameters and architecture. Using this information, attackers can craft sophisticated attacks by leveraging gradient-descent search algorithms. Gray-box attacks assume partial knowledge of the target model. Although not as powerful as white-box attacks, gray-box attacks can still be effective by exploiting certain vulnerabilities or weaknesses in the model. An attacker may have access to limited information, such as the model's architecture, output probabilities, or a subset of its parameters. This partial knowledge can be used to

devise evasion strategies and generate AEs to deceive a model. Black-box attacks present the most challenging scenario, as the attacker has zero knowledge of the target model. In such cases, attackers often use techniques such as transferability, whereby they train substitute models or gather information from similar models to create surrogate models. These surrogate models act as proxies for the target model and are used to generate AEs. By exploiting the transferability of the AEs, an attacker fools the target model without direct knowledge of its internals. In our approach, we generate AEs for AdverNet using the JSMA technique. These AEs are then tested against the target models to assess their vulnerabilities and the potential impact of the attack.

4. DVars for Adversarial Attack Defense

The proposed method for defending against adversarial attacks in network traffic classification models involves a two-step process, namely, feature selection and variable derivation. This section provides a detailed explanation of the methodology employed in this paper. Figure 4 shows the schematic overview of the proposed system.

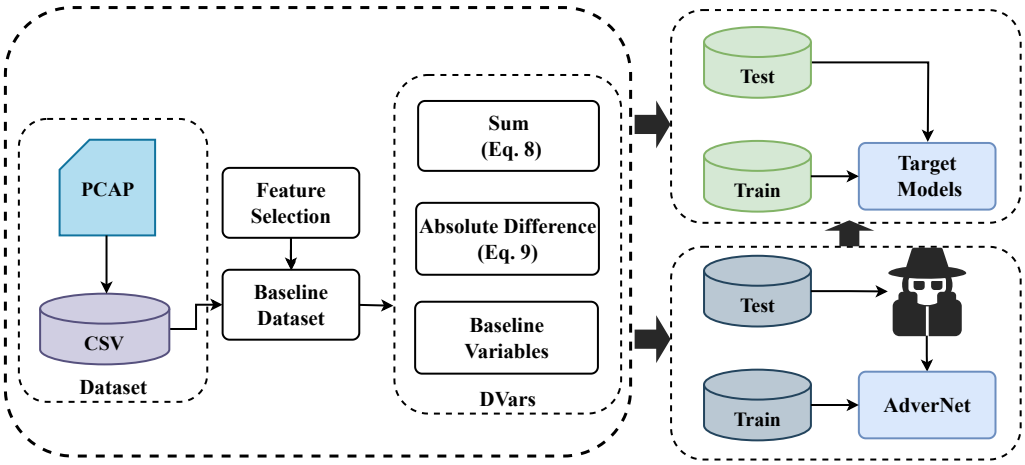


Figure 4. Schematic overview of the proposed system.

4.1. Feature Selection

Feature selection plays a crucial role in identifying the most relevant input feature subset that positively influences the classification performance of ML models [55]. During our experiments, we focused on the analysis of four specific features of network flow: source bytes (sbytes), destination bytes (dbytes), source packets (spkts), and destination packets (dpkts), as explored by Jiang et al. [56]. These features represent the counts of bytes and packets transmitted between the source and destination of a network flow. We selected these four features because of their susceptibility to modification by attackers through small content changes. Moreover, changes in these features do not affect the underlying logic or maliciousness of the network flow. These four features are intrinsic to network flows and are often used to distinguish malicious from benign flows.

4.2. DVar Algorithms

Following the logic of introducing randomness into the input data without compromising the underlying logic and maliciousness of the network flow, we propose two DVar algorithms. These algorithms are designed to compute representative features considering the sum of adjacent variables and the absolute differences between adjacent variables. The former aims to capture the holistic characteristics of a network flow, which allows for the extraction of valuable insights from the combined features of adjacent variables, whereas the latter aims to discern subtle patterns and variations within network flows. Table 1 shows the representation of DVars using both the sum and the absolute difference in the flow characteristics.

Table 1. A representation of DVars using both the sum and absolute difference in adjacent flow features; *C* indicates the condition.

Sbytes	Dbytes	Spkts	Dpkts	C
$sbytes_1$	$dbytes_1$	$spkts_1$	$dpkts_1$	—
...	...	...	...	—
$sbytes_{n+1}$	$dbytes_{n+1}$	$spkts_{n+1}$	$dpkts_{n+1}$	Equation (7)
$\|spkts_{n-1}\|$	$\|dbytes_{n-1}\|$	$\|spkts_{n-1}\|$	$\|dpkts_{n-1}\|$	Equation (8)
$sbytes_n$	$dbytes_n$	$spkts_n$	$dpkts_n$	—

4.2.1. The Sum of Adjacent Values

This process involves taking the sum of the adjacent values of each set of features in the baseline data. To ensure that the DVars remain within the same feature space as the baseline and preserve the validity of the network flow, a condition is set to retain the maximum value if any resulting value exceeds the maximum value in the baseline. This process is repeated for each adjacent feature value until the final feature value of the selected features is obtained. The new set of representative features obtained through this process are then combined with the baseline dataset. Representative feature Y_{ij} can be calculated using Equation (7).

$$Y_{ij} = \begin{cases} X_{ij} + X_{i(j+1)}, & \text{if } Y_{ij} > \max(X_{i,:}) \\ \max(X_{i,:}), & \text{otherwise} \end{cases} \tag{7}$$

where X is the baseline dataset, X_{ij} represents the value of the j-th feature in the i-th sample, Y is the matrix containing the representative features, Y_{ij} represents the value of the representative feature for the j-th feature in the i-th sample, and $\max(X_{i,:})$ represents the maximum value of the features in the i-th sample.

The process of deriving variables using the sum of adjacent feature values is shown in Algorithm 1.

Algorithm 1: Derivation of representative features using the sum of adjacent values.

Data: Baseline dataset X
Result: Matrix of representative features Y
for *each sample i in X* **do**
  for $j = 1$ **to** *number of features* **do**
    if $j + 1 \leq$ *number of features* **then**
      $Y_{ij} = X_{ij} + X_{i(j+1)}$;
      if $Y_{ij} > \max(X_{i,:})$ **then**
        $Y_{ij} = \max(X_{i,:})$;
      end
    end
    else
      $Y_{ij} = \max(X_{i,:})$;
    end
  end
end

4.2.2. The Absolute Difference in Adjacent Values

In contrast to the sum of adjacent values, this process computes the absolute difference in adjacent values in each set of features in the baseline dataset. To keep the DVars in the same feature space with the baseline, if any resulting value is below the minimum value of the baseline data, a condition is set to retain the minimum value. This process is

repeated for each adjacent feature value until the final feature value of the selected features is obtained. Representative feature Y_{ij} can be calculated using Equation (8).

$$Y_{ij} = \begin{cases} |X_{ij} - X_{i(j+1)}|, & \text{if } |Y_{ij}| < \min(|X_{i,:}|) \\ \min(|X_{i,:}|), & \text{otherwise} \end{cases} \tag{8}$$

where X represents the baseline dataset; X_{ij} denotes the value of the j-th feature in the i-th sample of the baseline dataset; Y is the matrix containing the representative features; Y_{ij} represents the value of the representative feature for the j-th feature in the i-th sample; $|X_{i,:}|$ denotes the absolute values of all elements in the i-th row of the baseline dataset; $\min(|X_{i,:}|)$ represents the minimum absolute difference between all values in the i-th row of the baseline dataset; and $|Y_{ij}|$ represents the absolute value of Y_{ij}, which is the calculated absolute difference. The process of deriving variables using the absolute difference between the values of the adjacent flow features is shown in Algorithm 2.

Algorithm 2: Derivation of representative features using the absolute difference in adjacent values.

Data: Baseline dataset X
Result: Matrix of representative features Y
for *each sample i in X* **do**
  for $j = 1$ **to** *number of features* **do**
    if $j + 1 \leq$ *number of features* **then**
      $Y_{ij} = |X_{ij} - X_{i(j+1)}|$;
      if $|Y_{ij}| < \min(|X_{i,:}|)$ **then**
        $Y_{ij} = \min(|X_{i,:}|)$;
      end
    end
    else
      $Y_{ij} = |X_{ij}|$;
    end
  end
end

After acquiring a new set of representative features through the process described in the equations (Equations (7) and (8)), they are merged with the baseline dataset. Subsequently, this combined dataset is used to train the ML models for network traffic classification. The primary objective of this training is to enhance the ability of the models to withstand adversarial attacks while also improving their overall accuracy.

5. Dataset and Performance Metrics

5.1. Dataset

The performance of the proposed method was evaluated using the Canadian Institute of Cybersecurity (CSE-CIC-IDS-2018) dataset [57], recognized as a standard benchmark dataset in the field of network security. This dataset has three classes, namely, a benign class and two malicious classes, which comprise two families of Denial of Service (DoS) attacks: GoldenEye and Slowloris. Due to the high class imbalance present in the dataset, we re-sampled this dataset using a random down-sampling technique for the majority class to achieve a more balanced dataset. Moreover, the dataset was cleansed to eliminate missing entries. The resulting dataset consisted of 50,400 class counts. Further details of this dataset are provided in Table 2.

Table 2. The breakdown of the CSE-CIC-IDS2018 dataset before and after preprocessing.

	Benign	**GoldenEye**	**Slowloris**
Before preprocessing (raw dataset)			
Class counts	996,077	41,508	10,990
Proportion of total	94.99%	3.96%	1.05%
After preprocessing (re-sampled dataset)			
Class counts	21,200	18,400	10,800
Proportion of total	42.06%	36.51%	21.43%

5.2. Performance Metrics

To measure the performance of our method, we employed several commonly used performance metrics, including accuracy (Acc), precision (Prec), recall (Rec), and F1-score (F1). These metrics provide insights into the effectiveness of our defense technique in correctly classifying network flows:

- Accuracy: This is a common metric used to evaluate the performance of ML models. It measures the proportion of correctly classified samples relative to the total number of samples.

$$\text{Accuracy} : \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{9}$$

- Precision: This metric measures the proportion of true-positive predictions to the total number of positive predictions.

$$\text{Precision} : \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{10}$$

- Recall: This metric measures the proportion of true-positive predictions to the total number of positive samples in the dataset.

$$\text{Recall} : \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{11}$$

- F1-score: This metric is the harmonic mean of precision and recall, providing a single score that balances both metrics. It ranges from 0 to 1, with higher values indicating better performance.

$$\text{F1-score} : 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

True positives (TPs) and true negatives (TNs) represent the number of instances that were correctly classified as positive and negative, respectively, whereas false positives (FPs) and false negatives (FNs) represent the number of instances that were incorrectly classified as positive and negative, respectively.

For adversarial ML, additional evaluation metrics may be required to measure the robustness of the model against adversarial attacks. A commonly used metric for evaluating the robustness of ML models is adversarial accuracy:

- Adversarial Accuracy: Adversarial accuracy measures the percentage of correct predictions made by the model on the AEs. This metric provides a measure of the model's ability to correctly classify AEs.

$$\text{Adv Acc} = \frac{\text{TP}_{\text{adv}} + \text{TN}_{\text{adv}}}{\text{TP}_{\text{adv}} + \text{FP}_{\text{adv}} + \text{TN}_{\text{adv}} + \text{FN}_{\text{adv}}} \tag{13}$$

A higher value of adversarial accuracy indicates that the model is more resilient to AEs.

6. Results and Analysis

In this section, we present the results of our experiments and discuss their implications. We evaluated the performance of our proposed method in defending against adversarial attacks in the context of network traffic classification. Furthermore, an assessment in comparison to similar techniques applied to shallow models was performed. These experiments were carried out with a dataset consisting of various network flows.

6.1. Experimental Setup

Before presenting the results, we provide details of the experimental setup. We used three state-of-the-art network traffic classification models as the baseline. These models were trained on a labeled dataset that contained both benign and two families of DoS malicious network flows. We split the dataset into training and test sets, in a 70:30 ratio. The entire test set was then used to generate AEs. This approach enables us to assess the resilience of the proposed method against adversarial perturbations. Table 3 outlines the hyper-parameters settings for the studied models and attack method.

Table 3. Hyper-parameters of the studied models and attack method.

Model	Parameters
AdvNet	Layer 1 = 128, Layer 2 = 64, Layer 3 = 3, activation = Relu, optimizer = adam, output_layer_activation = softmax, epochs = 50
DT	Criterion = gini, max_depth = 12
RF	Number of estimators = 170, random_state = 4, min_sample_split = 5
KNN	Number of neighbors = 10, distance metric = Euclidean
JSMA	$\theta = 1.0$, $\gamma = 0.1$, clip min = 0.0, clip max = 1.0

6.2. Comparative Analysis

The experimental results are presented below to provide a concise summary of the findings. The performance of various ML models in the absence of AEs is summarized in Table 4. The evaluation metrics of accuracy (Acc), precision (Prec), recall (Rec), and F1–score (F1) were used to assess model performance. AdverNet, which is a feed-forward neural network specifically employed to craft AEs, exhibited accuracy of 98%, while the DT, RF, and KNN models showed exceptional performance, achieving accuracy scores of 100% and consistent metrics at 99%. Additionally, the training curves of AdverNet on the baseline dataset are depicted in Figure 5.

Table 4. Baseline results of the target models in accuracy, precision, recall, or F1-score.

	Acc (%)	Prec (%)	Rec (%)	F1 (%)
AdverNet	0.98	0.98	0.98	0.98
DT	1.00	1.00	1.00	1.00
RF	0.99	0.99	0.99	0.99
KNN	0.99	0.99	0.99	0.99

Furthermore, as detailed in Table 5, we present a comparative analysis of baseline performance using the CSE-CIC-IDS2018 dataset. Notably, our approach consistently achieved exceptional levels of accuracy and other performance metrics, surpassing established baseline models. This table underscores the pronounced superiority of DVars on clean examples, thus reaffirming their substantial influence in the field of network traffic classification research. Additionally, a comparative study examining both the attack success rate and defense effectiveness on the leading models considered within the same dataset is presented in Tables 6 and 7, respectively. The proposed defense outperformed the

referenced studies, except for the removal of altered features (RAF), which was examined with a self-validated attack.

Table 5. Comparison of baseline performance of the leading models using the CSE-CIC-IDS2018 dataset.

Work	Model	Acc (%)	Prec (%)	Rec (%)	F1 (%)
Apruzzes et al. [6]	KNN	-	0.99	0.99	0.99
Pujari et al. [58]	RF	0.92	-	0.91	0.94
Pujari et al. [59]	RF	0.91	-	0.91	0.94
Shu et al. [60]	N/A	0.94	-	-	-
Ours	KNN	0.99	0.99	0.99	0.99

Table 6. Comparison of attack success rate of the leading models using the CSE-CIC-IDS2018 dataset.

Work	Attack	Acc (%)	Rec (%)	F1 (%)
Apruzzes et al. [6]	Self	-	0.48	-
Pujari et al. [58]	JSMA	0.84	0.57	0.59
Pujari et al. [59]	C&W	0.81	0.81	0.83
Shu et al. [60]	JSMA	0.94	-	-
Ours	JSMA	0.45	0.48	0.44

Table 7. Comparison of defense effectiveness of the leading models using the CSE-CIC-IDS2018 dataset.

Work	Attack	Acc (%)	Rec (%)	F1 (%)
Apruzzes et al. [6]	RAF	-	0.90	0.82
Pujari et al. [59]	GAN	0.82	0.83	0.84
Shu et al. [60]	A2	0.64	-	-
	A3	0.51	-	-
	A4	0.63	-	-
	A5	0.78	-	-
Ours	DVars	0.84	0.84	0.83

The results in Table 8 underscore the effectiveness of our defense strategy against adversarial attacks with different perturbation intensity. In the absence of perturbations, all models (DT, RF, KNN) performed exceptionally well, reflecting their robustness. As the perturbation intensity increased, the model performance gradually diminished. Despite this, KNN consistently demonstrated a higher degree of defense effectiveness, outperforming DT and RF, particularly at low and medium perturbation levels. This highlights the effectiveness of our defense mechanism in enhancing model resilience to adversarial attacks, with KNN emerging as the most reliable choice for such scenarios.

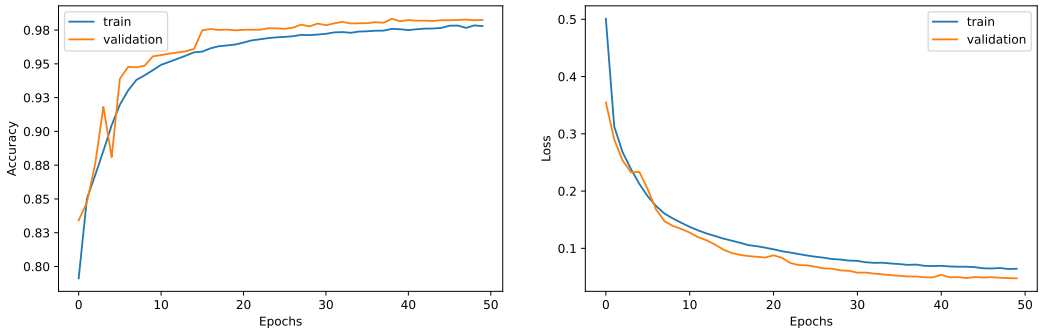
The results obtained when no AEs were present in the dataset are shown in (Figure 6). The figure shows Acc, Prec, Rec, and F1 with respect to each class in the dataset for different classification models, including DT, RF, KNN, and AdverNet. As can be seen in this figure, all the models performed well in classifying the network flows, achieving the best accuracy scores across all classes present in the dataset.

Table 8. Performance of the proposed defense against JSMA with different perturbation intensity.

Intensity (ϵ)	Model	Acc (%)	Prec (%)	Rec (%)	F1 (%)
Baseline	DT	1.00	1.00	1.00	1.00
	RF	0.99	0.99	0.99	0.99
	KNN	0.99	0.99	0.99	0.99

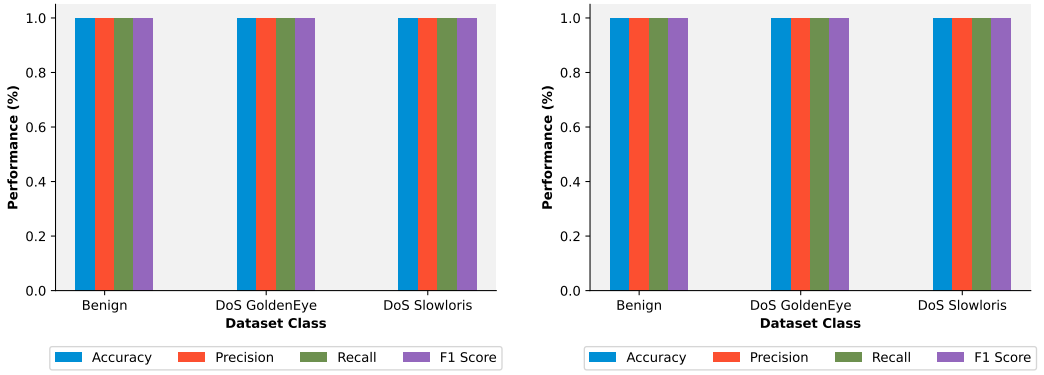
Table 8. *Cont.*

Intensity (ϵ)	Model	Acc (%)	Prec (%)	Rec (%)	F1 (%)
Low	DT	0.72	0.81	0.72	0.65
	RF	0.72	0.81	0.72	0.65
	KNN	0.84	0.87	0.84	0.83
Medium	DT	0.67	0.50	0.67	0.56
	RF	0.67	0.50	0.67	0.56
	KNN	0.72	0.79	0.72	0.64
High	DT	0.67	0.50	0.67	0.56
	RF	0.67	0.50	0.67	0.56
	KNN	0.66	0.67	0.66	0.55

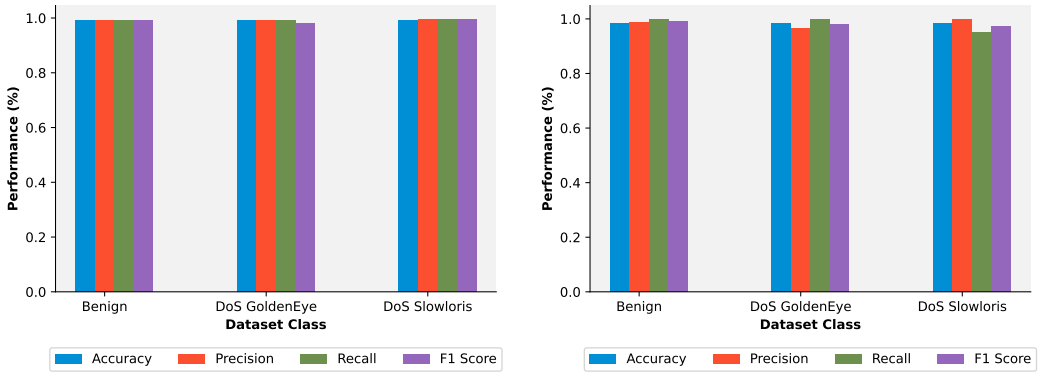


(**a**) Training accuracy vs. epochs (**b**) Training loss vs. epochs

Figure 5. Performance evaluation of AdverNet trained on the baseline dataset.



(**a**) Decision Tree (**b**) Random Forest



(**c**) KNN (**d**) AdverNet

Figure 6. The baseline results showing accuracy, precision, recall, and F1-score with respect to each class of the dataset.

Furthermore, Figure 7 presents the results of when the models were subjected to adversarial attacks with different perturbation intensity. The figure highlights the decrease in accuracy for each model to below 50%. As observed in the confusion matrix, the performance of these models declined even with small, subtle changes in the dataset, leading to a high class misclassification rate. A large proportion of DoS attacks were classified as benign (Figure 8). However, our proposed defense method still provided reasonable protection, maintaining relatively high accuracy, 84%, and achieving satisfactory precision, 87%, using our best-performing model, KNN.

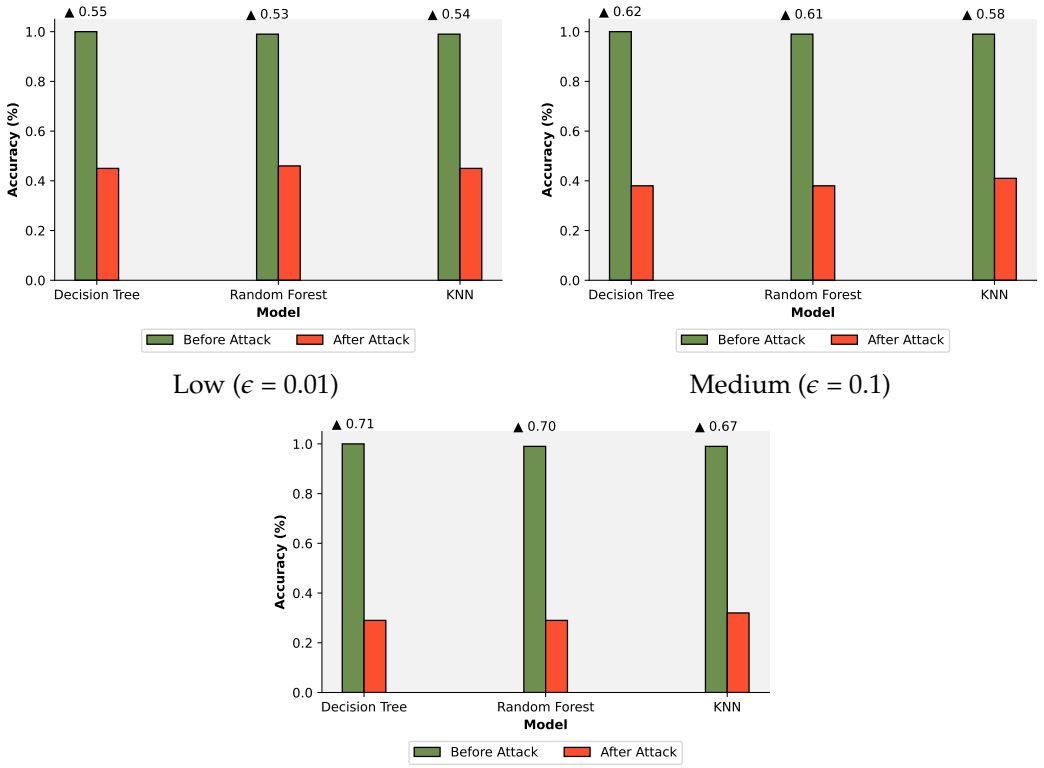
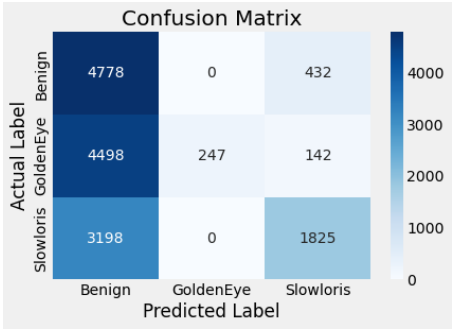


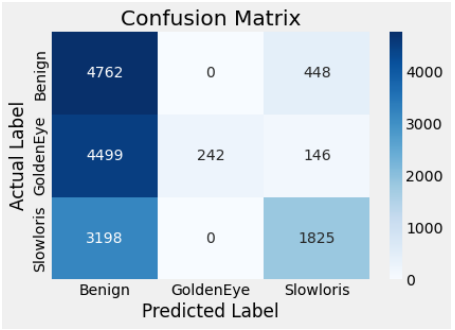
Figure 7. The results with the AE effect showing accuracy with respect to the perturbation intensity.

Based on the results presented in Table 8 and Figures 9 and 10, it is evident that the proposed defense method enhances the robustness of network traffic classification models against adversarial attacks. In the absence of attacks, the models achieved high accuracy, precision, recall, and F1-score, indicating their effectiveness in classifying the network traffic flows. However, when subjected to adversarial attacks of increasing intensity, the performance of these models gradually declined. Nevertheless, our defense method still provided significant protection, consistently maintaining accuracy of above 70% and 65% for low- and high-intensity attacks, respectively. Notably, KNN achieved the highest accuracy rates, 84%, and 72%, for low- and medium-intensity adversarial attacks, respectively, outperforming the DT and RF models. These results highlight the efficacy of the proposed approach in mitigating the impact of adversarial attacks on network traffic classification models.

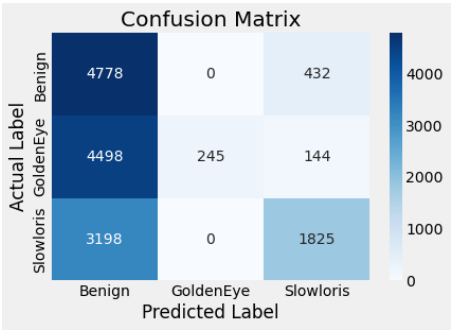
Through the integration of feature selection and variable derivation techniques, our defense method introduces additional randomness into input data. This increased randomness poses a challenge for attackers attempting to craft AEs that can completely deceive classification models. As a result, our defense method maintains reasonable levels of accuracy, precision, recall, and F1-score, even in adversarial scenarios.



(**a**) Decision Tree

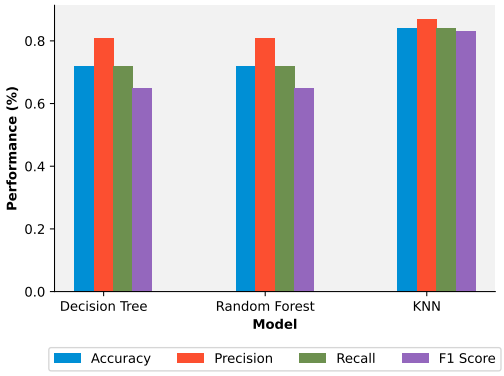


(**b**) Random Forest

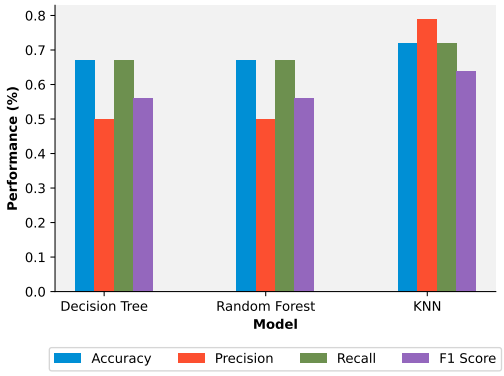


(**c**) KNN

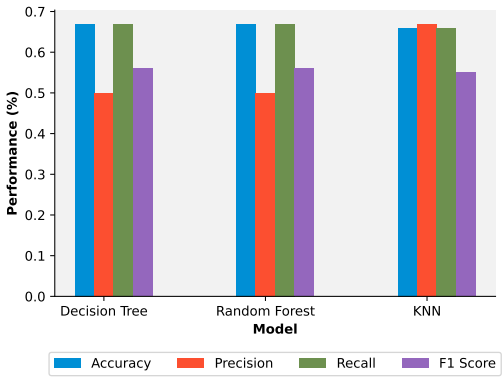
Figure 8. The confusion matrix showing results with AEs present with respect to low perturbation intensity (ϵ = 0.01).



Low (ϵ = 0.01)

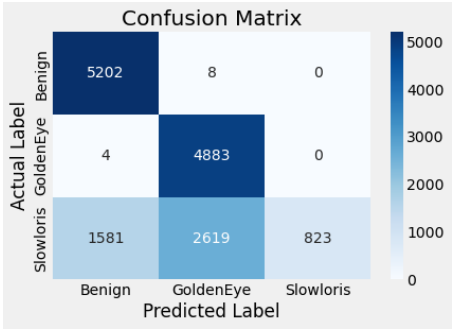


Medium (ϵ = 0.1)

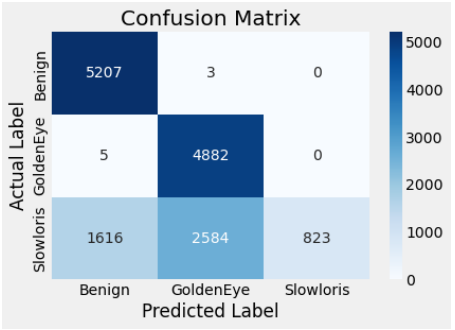


High (ϵ = 0.2)

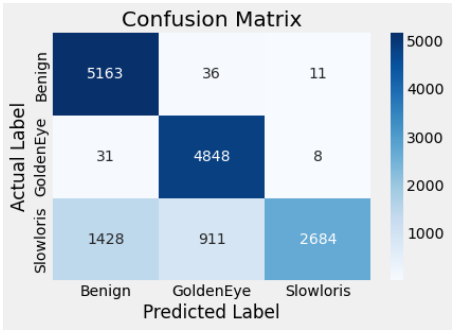
Figure 9. The results with defense effect, showing accuracy, precision, recall, and F1-score with respect to perturbation intensity.



(a) Decision Tree



(**b**) Random Forest



(**c**) KNN

Figure 10. The confusion matrix showing robustness of DVars against AEs with respect to low perturbation intensity (ϵ = 0.01).

6.3. Scalability

The proposed methodology not only enhances the resilience of ML-based network traffic classification models but also highlights significant scalability. Through the integration of DVars into training, our approach achieves a reduction in computational overhead, imposes minimal resource demands, and demonstrates adaptability across various classifiers. Its seamless compatibility with existing systems, coupled with a straightforward integration process, makes it a lightweight, efficient, and easily scalable solution that is particularly useful for deployment in real-world scenarios. Moreover, the adaptability of DVars to changing network dynamics and their ability to maintain robust performance underscores their scalability, positioning them as a versatile and practical solution for securing communication networks of varying scales against adversarial attacks. Additionally, it is imperative to highlight that even for clean data, the baseline classifiers exhibit commendable performance. This observation emphasizes the reliability of our approach, affirming that the introduction of DVars does not compromise the efficacy of the baseline classifiers during routine operational scenarios.

7. Conclusions and Future Research

This study examines the robustness of existing ML-based network traffic classification models against adversarial attacks. A novel framework called DVars is proposed to enhance the resilience of these models. DVars, which introduce randomness into the input data by generating distinct variables from the baseline dataset, play a crucial role in strengthening the robustness of models against adversarial attacks. Notably, the key distinction between the proposed approach and traditional adversarial training lies in the emphasis on utilizing DVars rather than AEs. The evaluation conducted herein reveals that the integration of DVars in the training of ML models considerably improves their robustness when faced with adversarial attacks. Moreover, the practicality and scalability of DVars are underscored by their computational efficiency and independence from AEs.

This research not only highlights the potential of DVars as an effective defense mechanism against adversarial attacks but also provides a viable avenue for further research.

Further research is warranted to assess the applicability of DVars to deep learning-based network traffic classification models.

Author Contributions: Conceptualization, J.M.A. and G.L.; methodology, J.M.A.; software, J.M.A. and J.Z.; validation, J.M.A., G.L. and J.Z.; formal analysis, H.M.B. and N.W; investigation, J.M.A.; resources, G.L.; data curation, N.W.; writing—original draft preparation, J.M.A. and J.Z.; writing—review and editing, J.M.A.; visualization, H.M.B.; supervision, G.L.; project administration, G.L.; funding acquisition, G.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Key R & D Program of China (2021QY0700); in part by National Natural Science Foundation of China (U21B2003); and in part by the Startup Foundation for Introducing Talent of the Nanjing University of Information Science & Technology, China (2023r014).

Data Availability Statement: The dataset used in the experiments is the CSE-CIC-IDS-2018 dataset, which is cited in the article.

Acknowledgments: All authors would like to thank the editors and reviewers for their advice.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Jmila, H.; Khedher, M.I. Adversarial machine learning for network intrusion detection: A comparative study. *Comput. Networks* **2022**, *214*, 109073. [CrossRef]
2. Fu, C.; Li, Q.; Shen, M.; Xu, K. Frequency domain feature based robust malicious traffic detection. *IEEE/ACM Trans. Netw.* **2022**, *31*, 452–467. [CrossRef]
3. Wang, C.; Chen, J.; Yang, Y.; Ma, X.; Liu, J. Poisoning attacks and countermeasures in intelligent networks: Status quo and prospects. *Digit. Commun. Networks* **2022**, *8*, 225–234. [CrossRef]
4. Pawlicki, M.; Choraś, M.; Kozik, R. Defending network intrusion detection systems against adversarial evasion attacks. *Future Gener. Comput. Syst.* **2020**, *110*, 148–154. [CrossRef]
5. Chan, P.P.; Zheng, J.; Liu, H.; Tsang, E.C.; Yeung, D.S. Robustness analysis of classical and fuzzy decision trees under adversarial evasion attack. *Appl. Soft Comput.* **2021**, *107*, 107311. [CrossRef]
6. Apruzzese, G.; Colajanni, M.; Marchetti, M. Evaluating the effectiveness of adversarial attacks against botnet detectors. In Proceedings of the 2019 IEEE 18th International Symposium on Network Computing and Applications (NCA), Cambridge, MA USA, 26–28 September 2019; IEEE: Piscataway Township, NJ, USA, 2019; pp. 1–8.
7. Biggio, B.; Corona, I.; Maiorca, D.; Nelson, B.; Šrndić, N.; Laskov, P.; Giacinto, G.; Roli, F. Evasion attacks against machine learning at test time. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Prague, Czech Republic, 22–26 September 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 387–402.
8. Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; Fergus, R. Intriguing properties of neural networks. *arXiv* **2013**, arXiv:1312.6199.
9. Ibitoye, O.; Shafiq, O.; Matrawy, A. Analyzing adversarial attacks against deep learning for intrusion detection in IoT networks. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Big Island, HI, USA, 9–13 December 2019; IEEE: Piscataway Township, NJ, USA, 2019; pp. 1–6.
10. Martins, N.; Cruz, J.M.; Cruz, T.; Abreu, P.H. Adversarial machine learning applied to intrusion and malware scenarios: A systematic review. *IEEE Access* **2020**, *8*, 35403–35419. [CrossRef]
11. Apruzzese, G.; Andreolini, M.; Colajanni, M.; Marchetti, M. Hardening random forest cyber detectors against adversarial attacks. *IEEE Trans. Emerg. Top. Comput. Intell.* **2020**, *4*, 427–439. [CrossRef]
12. Apruzzese, G.; Andreolini, M.; Ferretti, L.; Marchetti, M.; Colajanni, M. Modeling realistic adversarial attacks against network intrusion detection systems. *Digit. Threat. Res. Pract. (DTRAP)* **2022**, *3*, 1–19. [CrossRef]
13. Aiken, J.; Scott-Hayward, S. Investigating adversarial attacks against network intrusion detection systems in sdns. In Proceedings of the 2019 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Dallas, TX, USA, 12–14 November 2019; IEEE: Piscataway Township, NJ, USA, 2019; pp. 1–7.
14. Han, D.; Wang, Z.; Zhong, Y.; Chen, W.; Yang, J.; Lu, S.; Shi, X.; Yin, X. Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 2632–2647. [CrossRef]
15. Wang, J.; Qixu, L.; Di, W.; Dong, Y.; Cui, X. Crafting adversarial example to bypass flow-&ML-based botnet detector via RL. In Proceedings of the 24th International Symposium on Research in Attacks, Intrusions and Defenses, San Sebastian, Spain, 6–8 October 2021; pp. 193–204.
16. Zhang, H.; Wang, J. Defense against adversarial attacks using feature scattering-based adversarial training. *Adv. Neural Inf. Process. Syst.* **2019**, *32, 1831–1841.*

17. Carlini, N.; Athalye, A.; Papernot, N.; Brendel, W.; Rauber, J.; Tsipras, D.; Goodfellow, I.; Madry, A.; Kurakin, A. On evaluating adversarial robustness. *arXiv* **2019**, arXiv:1902.06705.
18. Wong, E.; Rice, L.; Kolter, J.Z. Fast is better than free: Revisiting adversarial training. *arXiv* **2020**, arXiv:2001.03994.
19. Feinman, R.; Curtin, R.R.; Shintre, S.; Gardner, A.B. Detecting adversarial samples from artifacts. *arXiv* **2017**, arXiv:1703.00410.
20. Wang, J.; Pan, J.; AlQerm, I.; Liu, Y. Def-ids: An ensemble defense mechanism against adversarial attacks for deep learning-based network intrusion detection. In Proceedings of the 2021 International Conference on Computer Communications and Networks (ICCCN), Athens, Greece, 19–22 July 2021; IEEE: Piscataway Township, NJ, USA, 2021; pp. 1–9.
21. Edwards, D.; Rawat, D.B. Study of adversarial machine learning with infrared examples for surveillance applications. *Electronics* **2020**, *9*, 1284. [CrossRef]
22. Vitorino, J.; Praça, I.; Maia, E. SoK: Realistic adversarial attacks and defenses for intelligent network intrusion detection. *Comput. Secur.* **2023**, *134*, 103433. [CrossRef]
23. Mohanty, H.; Roudsari, A.H.; Lashkari, A.H. Robust stacking ensemble model for darknet traffic classification under adversarial settings. *Comput. Secur.* **2022**, *120*, 102830. [CrossRef]
24. Zaki, F.A.M.; Chin, T.S. FWFS: Selecting robust features towards reliable and stable traffic classifier in SDN. *IEEE Access* **2019**, *7*, 166011–166020. [CrossRef]
25. Cao, J.; Wang, D.; Qu, Z.; Sun, H.; Li, B.; Chen, C.L. An improved network traffic classification model based on a support vector machine. *Symmetry* **2020**, *12*, 301. [CrossRef]
26. Bhatia, M.; Sharma, V.; Singh, P.; Masud, M. Multi-level P2P traffic classification using heuristic and statistical-based techniques: a hybrid approach. *Symmetry* **2020**, *12*, 2117. [CrossRef]
27. Dey, S.; Ye, Q.; Sampalli, S. A machine learning based intrusion detection scheme for data fusion in mobile clouds involving heterogeneous client networks. *Inf. Fusion* **2019**, *49*, 205–215. [CrossRef]
28. Rust-Nguyen, N.; Sharma, S.; Stamp, M. Darknet Traffic Classification and Adversarial Attacks Using Machine Learning. *Comput. Secur.* **2023**, 103098. [CrossRef]
29. Lin, Z.; Shi, Y.; Xue, Z. Idsgan: Generative adversarial networks for attack generation against intrusion detection. In Proceedings of the Advances in Knowledge Discovery and Data Mining: 26th Pacific-Asia Conference, PAKDD 2022, Chengdu, China, 16–19 May 2022; Proceedings, Part III; Springer: Berlin/Heidelberg, Germany, 2022; pp. 79–91.
30. Alhajjar, E.; Maxwell, P.; Bastian, N. Adversarial machine learning in network intrusion detection systems. *Expert Syst. Appl.* **2021**, *186*, 115782. [CrossRef]
31. Asadi, M.; Jamali, M.A.J.; Parsa, S.; Majidnezhad, V. Detecting botnet by using particle swarm optimization algorithm based on voting system. *Future Gener. Comput. Syst.* **2020**, *107*, 95–111. [CrossRef]
32. Capuano, N.; Fenza, G.; Loia, V.; Stanzione, C. Explainable Artificial Intelligence in CyberSecurity: A Survey. *IEEE Access* **2022**, *10*, 93575–93600. [CrossRef]
33. McCarthy, A.; Ghadafi, E.; Andriotis, P.; Legg, P. Defending against adversarial machine learning attacks using hierarchical learning: A case study on network traffic attack classification. *J. Inf. Secur. Appl.* **2023**, *72*, 103398. [CrossRef]
34. Qian, Y.; Lu, H.; Ji, S.; Zhou, W.; Wu, S.; Yun, B.; Tao, X.; Lei, J. Adversarial example generation based on particle swarm optimization. *J. Electron. Inf. Technol.* **2019**, *41*, 1658–1665.
35. Usama, M.; Qayyum, A.; Qadir, J.; Al-Fuqaha, A. Black-box adversarial machine learning attack on network traffic classification. In Proceedings of the 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 24–28 June 2019; IEEE: Piscataway Township, NJ, USA, 2019; pp. 84–89.
36. Xu, H.; Ma, Y.; Liu, H.C.; Deb, D.; Liu, H.; Tang, J.L.; Jain, A.K. Adversarial attacks and defenses in images, graphs and text: A review. *Int. J. Autom. Comput.* **2020**, *17*, 151–178. [CrossRef]
37. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
38. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv* **2017**, arXiv:1706.06083.
39. Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z.B.; Swami, A. The limitations of deep learning in adversarial settings. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Delft, The Netherlands, 3–7 July 2016; IEEE: Piscataway Township, NJ, USA, 2016; pp. 372–387.
40. Yuan, X.; He, P.; Zhu, Q.; Li, X. Adversarial examples: Attacks and defenses for deep learning. *IEEE Trans. Neural Networks Learn. Syst.* **2019**, *30*, 2805–2824. [CrossRef] [PubMed]
41. Chakraborty, A.; Alam, M.; Dey, V.; Chattopadhyay, A.; Mukhopadhyay, D. A survey on adversarial attacks and defences. *CAAI Trans. Intell. Technol.* **2021**, *6*, 25–45. [CrossRef]
42. Qiu, S.; Liu, Q.; Zhou, S.; Wu, C. Review of artificial intelligence adversarial attack and defense technologies. *Appl. Sci.* **2019**, *9*, 909. [CrossRef]
43. Zhang, L.; Qi, G.J. Wcp: Worst-case perturbations for semi-supervised deep learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 3912–3921.
44. Bai, T.; Luo, J.; Zhao, J.; Wen, B.; Wang, Q. Recent advances in adversarial training for adversarial robustness. *arXiv* **2021**, arXiv:2102.01356.
45. Zhang, J.; Li, C. Adversarial examples: Opportunities and challenges. *IEEE Trans. Neural Networks Learn. Syst.* **2019**, *31*, 2578–2593. [CrossRef] [PubMed]

46. Anthi, E.; Williams, L.; Javed, A.; Burnap, P. Hardening machine learning denial of service (DoS) defences against adversarial attacks in IoT smart home networks. *Comput. Secur.* **2021**, *108*, 102352. [CrossRef]
47. Abou Khamis, R.; Matrawy, A. Evaluation of adversarial training on different types of neural networks in deep learning-based idss. In Proceedings of the 2020 International Symposium on Networks, Computers and Communications (ISNCC), Montreal, QC, Canada, 20–22 October 2020; IEEE: Piscataway Township, NJ, USA, 2020; pp. 1–6.
48. Chollet, F. Keras. 2015. Available online: https://github.com/fchollet/keras (accessed on 15 September 2023).
49. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
50. Nicolae, M.I.; Sinn, M.; Tran, M.N.; Buesser, B.; Rawat, A.; Wistuba, M.; Zantedeschi, V.; Baracaldo, N.; Chen, B.; Ludwig, H.; et al. Adversarial Robustness Toolbox v1. 0.0. *arXiv* **2018**, arXiv:1807.01069.
51. Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z.B.; Swami, A. Practical black-box attacks against machine learning. In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, Abu Dhabi, United Arab Emirates, 2–6 April 2017; pp. 506–519.
52. Debicha, I.; Cochez, B.; Kenaza, T.; Debatty, T.; Dricot, J.M.; Mees, W. Adv-Bot: Realistic Adversarial Botnet Attacks against Network Intrusion Detection Systems. *Comput. Secur.* **2023**, *129*, 103176. [CrossRef]
53. Merzouk, M.A.; Cuppens, F.; Boulahia-Cuppens, N.; Yaich, R. Investigating the practicality of adversarial evasion attacks on network intrusion detection. *Ann. Telecommun.* **2022**, *77*, 763–775. [CrossRef]
54. Teuffenbach, M.; Piatkowska, E.; Smith, P. Subverting network intrusion detection: Crafting adversarial examples accounting for domain-specific constraints. In Proceedings of the Machine Learning and Knowledge Extraction: 4th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2020, Dublin, Ireland, 25–28 August 2020; Proceedings 4; Springer: Berlin/Heidelberg, Germany, 2020; pp. 301–320.
55. Zhou, Y.; Cheng, G.; Jiang, S.; Dai, M. Building an efficient intrusion detection system based on feature selection and ensemble classifier. *Comput. Networks* **2020**, *174*, 107247. [CrossRef]
56. Jiang, H.; Lin, J.; Kang, H. FGMD: A robust detector against adversarial attacks in the IoT network. *Future Gener. Comput. Syst.* **2022**, *132*, 194–210. [CrossRef]
57. Canadian Institute for Cybersecurity. CSE-CIC-IDS2018 on AWS. 2018. Available online: https://www.unb.ca/cic/datasets/ids-2018.html (accessed on 15 September 2023).
58. Pujari, M.; Pacheco, Y.; Cherukuri, B.; Sun, W. A Comparative Study on the Impact of Adversarial Machine Learning Attacks on Contemporary Intrusion Detection Datasets. *SN Comput. Sci.* **2022**, *3*, 412. [CrossRef]
59. Pujari, M.; Cherukuri, B.P.; Javaid, A.Y.; Sun, W. An approach to improve the robustness of machine learning based intrusion detection system models against the carlini-wagner attack. In Proceedings of the 2022 IEEE International Conference on Cyber Security and Resilience (CSR), Virtual Conference, 27–29 July 2022; IEEE: Piscataway Township, NJ, USA, 2022; pp. 62–67.
60. Shu, R.; Xia, T.; Williams, L.; Menzies, T. Omni: Automated ensemble with unexpected models against adversarial evasion attack. *Empir. Softw. Eng.* **2022**, *27*, 1–32. [CrossRef]