electronics

Article

Enhancing Adversarial Robustness in Network Intrusion Detection: A Novel Adversarially Trained Neural Network Approach

Vahid Heydari [1,*] **and Kofi Nyarko** [2]

[1] Computer Science Department, Morgan State University, Baltimore, MD 21251, USA
[2] Electrical and Computer Engineering Department, Morgan State University, Baltimore, MD 21251, USA; kofi.nyarko@morgan.edu
[*] Correspondence: vahid.heydari@morgan.edu

Abstract

Machine learning (ML) has greatly improved intrusion detection in enterprise networks. However, ML models remain vulnerable to adversarial attacks, where small input changes cause misclassification. This study evaluates the robustness of a *Random Forest (RF)*, a standard *neural network (NN)*, and a Transformer-based *Network Intrusion Detection System (NIDS)*. It also introduces *ADV_NN*, an adversarially trained neural network designed to improve resilience. Model performance is tested using the UNSW-NB15 dataset under both clean and adversarial conditions. The attack types include *Projected Gradient Descent (PGD)*, *Fast Gradient Sign Method (FGSM)*, and *Black-Box* transfer attacks. The proposed *ADV_NN* achieves 86.04% accuracy on clean data. It maintains over 80% accuracy under *PGD* and *FGSM* attacks, and exceeds 85% under *Black-Box* attacks at $\epsilon = 0.15$. In contrast, the *RF*, *NN*, and Transformer-based models suffer significant degradation under adversarial perturbations. These results highlight the need to incorporate adversarial defenses into ML-based NIDS for secure deployment in real-world environments.

Keywords: adversarial attacks; machine learning; network intrusion detection systems (NIDSs); cybersecurity; model robustness

Academic Editor: Aryya Gangopadhyay

Received: 16 June 2025
Revised: 6 August 2025
Accepted: 12 August 2025
Published: 15 August 2025

Citation: Heydari, V.; Nyarko, K. Enhancing Adversarial Robustness in Network Intrusion Detection: A Novel Adversarially Trained Neural Network Approach. *Electronics* **2025**, *14*, 3249. https://doi.org/10.3390/electronics14163249

1. Introduction

The rapid expansion of digital networks and the growing interconnectivity of enterprise systems have led to an increase in both the frequency and complexity of cyberattacks. Robust network security is therefore essential to protect enterprise systems that manage sensitive data. *Network Intrusion Detection Systems (NIDSs)* play a critical role in this context. They continuously monitor network traffic for malicious activity, using pattern recognition and anomaly detection techniques to identify potential threats [1,2].

In recent years, *machine learning (ML)* has emerged as a transformative approach for building adaptive and efficient NIDS capable of detecting both known and novel attack patterns [3,4]. Deep learning, in particular, has demonstrated strong performance in anomaly detection due to its powerful feature extraction capabilities [5]. Transformer-based models offer global attention mechanisms and have been explored to capture complex dependencies in network data. Despite their success in controlled settings, ML-based NIDSs remain vulnerable to *adversarial attacks*. These are carefully crafted perturbations that deceive models into misclassifying malicious activity as benign. These attacks pose a serious threat to enterprise cybersecurity.

A key limitation of conventional NIDS evaluation practices is their reliance on performance metrics derived exclusively from clean, unperturbed data. High accuracy on such datasets may create a false sense of security regarding a model's robustness in real-world settings. Recent studies have shown that even small adversarial perturbations can drastically impair the performance of ML-based NIDSs [6]. This concern is especially critical in Industry 4.0 environments, where cyber–physical systems rely on ML to safeguard critical infrastructure [7]. Therefore, evaluations should go beyond traditional metrics like accuracy and AUC, and also assess robustness under adversarial conditions.

Adversarial attacks are highly relevant to cybersecurity because they exploit model vulnerabilities using techniques that mimic realistic attack strategies. For example, an attacker may subtly alter network traffic features to evade detection while preserving malicious functionality. Even minimal changes to traffic flow attributes can achieve high evasion rates [6].

To study the effects of adversarial samples on NIDS, this work uses the UNSW-NB15 dataset—a benchmark designed to reflect modern attack scenarios and diverse traffic conditions. Created using the *IXIA PerfectStorm* platform, it includes both benign and malicious traffic. Compared to legacy datasets such as KDDCUP99, UNSW-NB15 provides a more realistic and current representation of cybersecurity threats [8–12].

In our previous study [13], we evaluated the adversarial robustness of two ML models: a *Random Forest (RF)* classifier and a *neural network (NN)*. Both were trained on the UNSW-NB15 dataset, and we used *Projected Gradient Descent (PGD)* to generate adversarial samples at various perturbation levels. The results showed that although both models performed well on clean data, even small perturbations ($\epsilon = 0.01$) caused a significant drop in detection accuracy. The *NN* was especially vulnerable compared to the *RF*.

To mitigate these vulnerabilities, we propose *ADV_NN*, a novel adversarially trained neural network. It incorporates *Curriculum Adversarial Training (CAT)* and additional regularization strategies. While adversarial training is a known defense, its application to NIDSs presents unique challenges. This work introduces a training framework that combines curriculum-based adversarial training with a composite loss function tailored to NIDSs. The loss minimizes classification error on both clean and adversarial inputs, while improving model stability through *gradient alignment loss* and *feature smoothing loss*. These regularization terms help address the specific vulnerabilities of NIDSs to subtle input perturbations.

We also compare *ADV_NN* against a Transformer-based NIDS. Transformer models are gaining attention for their ability to model global feature relationships, but their robustness under attack is not well understood. Following Madry et al. [14], we evaluate all models using multiple attack types: *PGD*, *Fast Gradient Sign Method (FGSM)*, and *Black-Box* transfer attacks. This allows us to assess their generalizability and resilience.

1.1. Contributions

This paper builds on our prior research [13] by introducing *ADV_NN*, an adversarially trained neural network tailored for ML-based NIDSs. The key contributions are as follows:

- **Enhanced Adversarial Training:** We incorporate *gradient alignment* and *feature smoothing* losses into the training process to reduce discrepancies between clean and adversarial samples at both the feature and gradient levels.
- **Curriculum Adversarial Training:** We gradually increase the attack strength (ϵ) over training epochs, allowing the model to adapt progressively to stronger perturbations.
- **Comparative Evaluation with Traditional and Transformer Models:** We benchmark *ADV_NN* against *RF*, standard *NN*, and a Transformer-based NIDS. Results show that *ADV_NN* achieves superior robustness across *PGD*, *FGSM*, and *Black-Box* attacks.

- **Comprehensive Empirical Analysis:** Experiments on the UNSW-NB15 dataset demonstrate that *ADV_NN* maintains high accuracy under various adversarial attacks. In contrast, the baseline models exhibit significant performance degradation, especially at higher perturbation levels.

1.2. Paper Organization

The remainder of the paper is organized as follows: Section 2 reviews prior work on adversarial attacks and defenses in ML-based NIDS. Section 3 describes the proposed methodology, including dataset selection, model design, and the adversarial training approach. Section 4 outlines the hyperparameter tuning process and provides justification for the selected configuration. Section 5 presents a theoretical analysis of the loss components and training strategy. Section 6 reports empirical results under both clean and adversarial conditions. Section 7 discusses key findings, limitations, and future research directions. Finally, Section 8 concludes the paper.

2. Related Work

Machine learning techniques have become central to modern NIDSs due to their ability to analyze network traffic and classify patterns indicative of cyber threats. However, the susceptibility of ML-based NIDSs to adversarial attacks has emerged as a critical concern. This section reviews ML techniques, adversarial attack strategies, defense mechanisms, and evaluation challenges, providing context for the proposed *ADV_NN* framework.

2.1. Machine Learning Techniques for NIDS

Common ML models used in NIDSs include *Decision Trees*, *Random Forests (RF)*, *Support Vector Machines (SVMs)*, *neural networks (NNs)*, and more recently, *Transformer-based architectures*. These models extract features such as protocol type, connection duration, and packet statistics to detect malicious activity. Transformers, in particular, offer a way to capture global dependencies among features using self-attention mechanisms.

Decision Trees: *Decision Trees* are widely adopted due to their interpretability and efficiency in rule-based classification. They differentiate between benign and malicious traffic using hierarchical feature splits [15,16]. However, they are prone to overfitting in high-dimensional spaces, which can limit their ability to generalize to new attack types. Recent work by [2] demonstrates their use in lightweight NIDSs but also highlights vulnerabilities under adversarial conditions.

Random Forests: *RF* models aggregate multiple *Decision Trees* trained on randomly selected feature subsets. This ensemble approach improves generalization and reduces overfitting. *RF* is widely used in NIDSs due to its stability and accuracy [15–17]. Nonetheless, it remains vulnerable to adversarial perturbations that subtly alter input features.

Support Vector Machines: *SVMs* classify network data by identifying optimal hyperplanes in high-dimensional feature spaces [18,19]. While they are effective in detecting nuanced attack patterns, their high computational cost limits scalability, particularly in real-time or large-scale environments.

Neural Networks: *NNs*, especially deep architectures, excel at learning complex feature representations. They have demonstrated strong performance in detecting advanced cyber threats [20,21]. However, their non-linear decision boundaries make them highly susceptible to adversarial manipulations. Graph-based extensions have been proposed to improve structural learning and robustness [5].

Transformer Models: Transformer-based architectures have recently been explored for NIDSs due to their self-attention mechanisms, which can capture global dependencies among input features. These models show strong performance on clean data and can model complex temporal and structural relationships. However, their robustness to adversarial attacks remains an open challenge. This study evaluates them alongside traditional models.

While traditional ML models perform well on clean datasets, their robustness degrades significantly under adversarial conditions. Feature selection techniques such as *Principal Component Analysis (PCA)* and *Correlation Analysis* may reduce redundancy and improve interpretability, but they offer limited protection against adversarial manipulation. For example, recent work introduced a perturbability score (PS) metric to identify features vulnerable to evasion attacks. Selecting robust, low-PS features can help reduce the attack surface. However, PS-guided selection alone does not fully mitigate adversarial threats [22]. Robust training methodologies remain essential to address these limitations.

2.2. Adversarial Attacks on Machine Learning Models for NIDS

Adversarial attacks threaten ML-based NIDS by subtly manipulating input features to evade detection. These perturbations can cause classifiers to mislabel malicious traffic as benign, undermining system security. This study evaluates robustness against three major attack types: *FGSM*, *PGD*, and *Black-Box* transfer attacks.

Fast Gradient Sign Method: *FGSM* is a fast, one-step white-box attack. It perturbs inputs in the direction of the gradient of the loss function. While efficient, *FGSM* is generally less effective than multi-step attacks like *PGD*. However, it is useful as a baseline for measuring model sensitivity to quick perturbations.

Projected Gradient Descent: *PGD* extends *FGSM* by applying multiple iterative updates to craft stronger adversarial examples [14]. Each update is bounded within an ℓ_∞ ball of radius ϵ. *PGD* is widely used to benchmark robustness under worst-case white-box conditions. Studies [6,23] have shown that *PGD* can preserve malicious intent while modifying only non-functional features, making it both effective and realistic.

Black-Box Transfer Attack: *Black-box* attacks represent scenarios where adversaries lack access to model parameters or gradients. In this study, adversarial examples are crafted using *FGSM* on a surrogate neural network and transferred to other models. This approach reflects realistic scenarios where attackers train substitute models to launch transferable attacks [24].

Justification for Multi-Attack Evaluation: While *PGD* remains the most rigorous white-box benchmark due to its iterative nature, real-world adversaries may prefer simpler strategies. Including *FGSM* and *Black-Box* attacks provides a more complete evaluation of model robustness across a spectrum of threat scenarios, from fully informed to limited-knowledge attackers.

Impact on ML-Based NIDS: All three attack types significantly degrade the performance of standard neural networks, even at small perturbation levels (e.g., $\epsilon = 0.01$). Random Forests show moderate resistance but struggle under stronger attacks. The Transformer-based NIDS performs well on clean data but degrades sharply under adversarial conditions. In contrast, the proposed *ADV_NN* maintains high accuracy across all attacks. These results highlight the importance of robust training strategies that generalize across diverse adversarial scenarios.

2.3. Defense Mechanisms Against Adversarial Attacks

Various strategies have been proposed to defend ML-based NIDS against adversarial threats. Among them, adversarial training is one of the most effective. It improves robustness by incorporating perturbed inputs during training, allowing the model to learn to resist manipulation.

Adversarial Training: Holla et al. [24] enhance robustness in cloud-based NIDS by combining adversarial training with feature selection. Their method shows increased resistance to evasion attacks on the CICIDS2017 dataset. Kumar et al. [25] propose a deep autoencoder-based detector (NIDS-DA) that identifies adversarial samples altered in non-functional features. Their model generalizes well across datasets, including UNSW-NB15.

Ensemble Approaches: Apruzzese et al. [26] introduce a bagging-based ensemble strategy that integrates multiple classifiers to reduce the success rate of adversarial evasion. Their method increases robustness against strong attacks such as Carlini & Wagner (C&W) [27].

These approaches align with the goals of our proposed *ADV_NN* framework, which combines gradient alignment and feature smoothing to enhance robustness across multiple adversarial scenarios.

2.4. Datasets and Evaluation Challenges

Commonly used datasets for NIDS research include NSL-KDD, UNSW-NB15, and CICIDS2017. However, Maseer et al. [28] note that these datasets often lack diversity in attack types, limiting their usefulness for evaluating generalization to new threats. Apruzzese et al. [29] also highlight the lack of standardized benchmarks for adversarial robustness, which makes it difficult to compare results across studies.

This work uses the UNSW-NB15 dataset and adopts a multi-metric evaluation strategy to address these challenges.

2.5. Comparison Between Madry et al.'s Method and ADV_NN

Adversarial Training Strategy: Madry et al. [14] formulate adversarial training as a min–max optimization problem. The inner loop generates *PGD*-based adversarial examples, while the outer loop updates model parameters to minimize classification loss. Our proposed *ADV_NN* framework extends this approach by incorporating additional regularization terms that enhance model stability:

- **Gradient Alignment Loss:** Minimizes the difference between gradients of clean and adversarial inputs to reduce model sensitivity to perturbations.
- **Feature Smoothing Loss:** Promotes consistent internal feature representations across clean and adversarial samples.
- **Curriculum Training:** Gradually increases perturbation strength (ϵ) throughout training, improving generalization to stronger attacks.

Loss Function Differences: Whereas Madry et al. optimize only the adversarial cross-entropy loss, *ADV_NN* employs a composite loss that combines classification, gradient alignment, and feature smoothing objectives. This design enhances adversarial robustness without compromising clean-data performance:

$$L_{\text{total}} = L_{\text{CE}}(\mathbf{x}_{\text{clean}}, y) + L_{\text{CE}}(\mathbf{x}_{\text{adv}}, y) + \lambda_{\text{align}} L_{\text{align}} + \lambda_{\text{smooth}} L_{\text{smooth}} \tag{1}$$

Evaluation Metrics: While Madry et al.'s framework focuses on generic attack-agnostic benchmarks, our evaluation targets three concrete adversarial threat types: *PGD*, *FGSM*, and transfer-based *Black-Box* attacks. Results show that *ADV_NN* consistently outperforms standard neural networks in the NIDS domain. These findings support recent studies advocating regularization-based defenses [29].

3. Methodology

3.1. Dataset Description

This study uses the UNSW-NB15 dataset, a widely adopted benchmark for evaluating NIDS. The dataset simulates realistic enterprise network traffic, including both benign and malicious activities. It contains over 100 GB of flow-level data collected across two sessions.

The dataset includes 49 features that describe protocol types, connection durations, TCP flags, packet sizes, and other flow-level attributes. These features span multiple layers of the network stack and support the detection of both low-level and application-layer attacks. Each sample is labeled as either normal or one of nine attack categories:

- **Fuzzers:** Random input generators used to find software vulnerabilities.
- **Backdoors:** Mechanisms that enable unauthorized remote access.
- **Denial of Service (DoS):** Attacks that exhaust resources to disrupt availability.
- **Reconnaissance:** Scanning activity used to collect information about a target.
- **Shellcode:** Payloads that execute arbitrary code after exploitation.
- **Worms:** Self-replicating malware that spreads across networks.
- **Exploits, Analysis, and Generic Attacks:** Categories covering various threat behaviors.

Compared to older datasets like KDDCUP99 and NSL-KDD, UNSW-NB15 offers a more realistic and up-to-date representation of network threats. It is well suited for evaluating adversarial robustness in intrusion detection models.

3.2. Data Preprocessing

The following preprocessing steps were applied before model training and evaluation:

- **Encoding Categorical Features:** Categorical attributes (e.g., *protocol*, *service*, *state*, and *attack category*) were transformed into numerical values using label encoding.
- **Handling Missing and Infinite Values:** Missing values were imputed using the feature-wise mean. Infinite values were clipped to remain within valid numerical ranges.
- **Feature Standardization:** All numerical features were normalized using *z-score normalization* to ensure zero mean and unit variance. This improves convergence for neural network training.
- **Train–Test Split:** A stratified 70/30 train–test split was performed, maintaining class balance across normal and attack categories.

From the original 49 features in the UNSW-NB15 dataset, the `id` and `attack_cat` columns were removed, as they are not suitable learning features. After label encoding and target variable separation, the final input space for all models consists of 42 features.

3.3. Model Architecture: Curriculum-Adversarial Neural Network (ADV_NN)

To improve adversarial robustness, we implemented a fully connected feedforward neural network (FFNN) and trained it using curriculum-based adversarial training. While adversarial training has been explored in other domains, its application to NIDS presents unique challenges. These include the subtle nature of network traffic perturbations and the need to preserve malicious intent in adversarial samples.

Our training framework combines curriculum-based adversarial training with a composite loss function tailored for intrusion detection. This function minimizes classification error on both clean and adversarial samples while improving model stability through two regularization terms: gradient alignment and feature smoothing. These components directly address vulnerabilities introduced by adversarial perturbations.

The final model, denoted as *ADV_NN*, is trained by gradually increasing attack strength and optimizing both classification and robustness objectives.

The architecture, shown in Figure 1, includes the following:

- **Input Layer:** Accepts d-dimensional feature vectors from the dataset.
- **Hidden Layers:** Two fully connected layers with 128 and 64 neurons, each using *ReLU* activation.

- **Output Layer:** A dense layer that outputs logits for binary classification. Softmax is applied to obtain class probabilities.

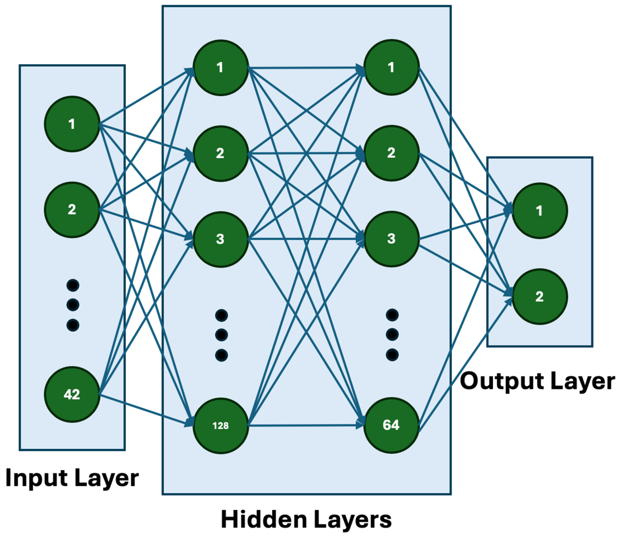


Figure 1. Architecture of the proposed *ADV_NN* model with two hidden layers.

Robustness is further enhanced through the composite loss function, described in Section 3.7.

3.4. Transformer-Based NIDS Architecture

To evaluate recent architectures, we also implemented a Transformer-based NIDS. Transformers are known for their ability to capture global dependencies across input features, which is valuable for identifying complex patterns in network traffic.

The model includes an input projection layer, positional encoding, and a multi-layer Transformer encoder, followed by a dense output layer for binary classification. It is trained using cross-entropy loss with class balancing to address label imbalance.

- **Input Projection:** A fully connected layer maps 42-dimensional input vectors into a higher-dimensional embedding space.
- **Transformer Encoder:** Comprises three encoder layers, each with four attention heads and a feedforward dimension of 256.
- **Output Layer:** A dense layer outputs class logits, followed by softmax activation.

To stabilize training, we applied early stopping and adaptive learning rate scheduling based on validation loss. The same preprocessing and stratified splits used for other models were applied to ensure a fair comparison.

Adversarial evaluations used the same PGD, FGSM, and Black-Box attacks applied to *ADV_NN* and other baseline models.

3.5. Adversarial Sample Generation

Adversarial samples were generated using the *PGD* method, as defined in [14]. *PGD* iteratively perturbs inputs to maximize model loss, while constraining changes within a bounded region.

PGD Attack Formulation

The *PGD* attack generates adversarial inputs through the following iterative process:

$$\mathbf{x}_{\text{adv}}^{(t+1)} = \Pi_{\mathbf{x},\epsilon}\left(\mathbf{x}_{\text{adv}}^{(t)} + \alpha \cdot \text{sign}\left(\nabla_{\mathbf{x}_{\text{adv}}^{(t)}} L(\theta, \mathbf{x}_{\text{adv}}^{(t)}, y)\right)\right) \quad (2)$$

where the formulas are as follows:

- $\mathbf{x}_{\text{adv}}^{(0)} = \mathbf{x}$ (the original input);
- α is the step size;
- $L(\theta, \cdot, y)$ is the loss function;
- $\text{sign}(\nabla L)$ represents the sign of the gradient;
- $\Pi_{\mathbf{x},\epsilon}$ projects the result back into the ℓ_∞ ball of radius ϵ centered at $\mathbf{x}$.

3.6. Curriculum-Based Adversarial Training

To gradually improve model robustness, we employed curriculum-based adversarial training. Rather than using a fixed perturbation budget throughout training, the attack strength ϵ was increased linearly over the training epochs.

The perturbation at epoch e is defined as follows:

$$\epsilon_e = \epsilon_{\max} \cdot \left(\frac{e}{E}\right) \tag{3}$$

where E is the total number of epochs and $\epsilon_{\max}$ is the maximum perturbation. Figure 2 shows the progression of ϵ from 0.01 to 0.2 over 20 epochs.

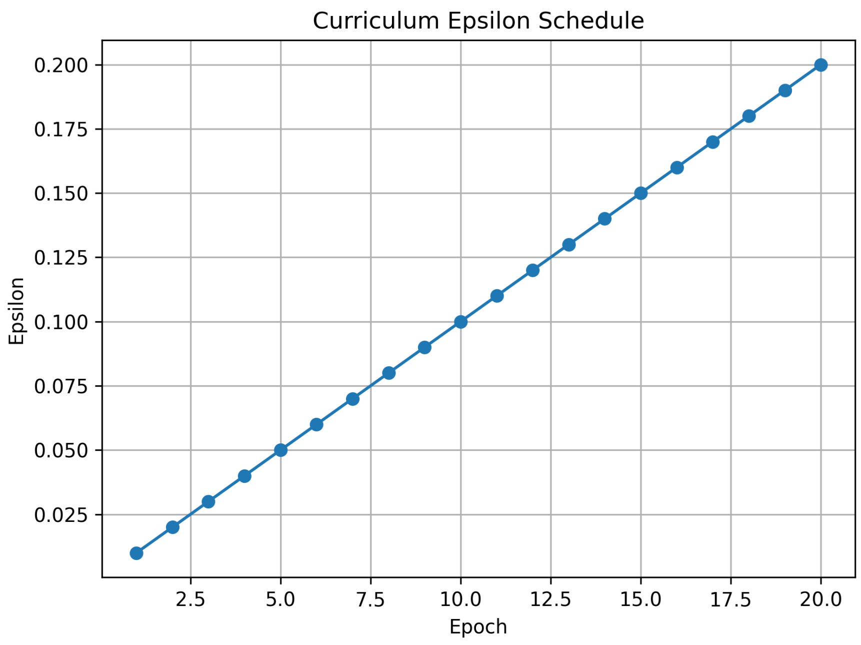


Figure 2. Curriculum-based epsilon schedule from 0.01 to 0.2 across 20 epochs.

For evaluation, fixed ϵ values of {0.01, 0.05, 0.1, 0.15} were used.

3.7. Loss Functions for Robust Training

The total loss used to train *ADV_NN* combines cross-entropy with two regularization terms:

- **Cross-Entropy Loss (L_{CE}):** Applied to both clean and adversarial inputs to enforce correct classification.
- **Feature Smoothing Loss (L_{smooth}):** Encourages similar feature representations in the final hidden layer:

$$L_{\text{smooth}} = \mathbb{E}\left[\|f(\mathbf{x}_{\text{clean}}) - f(\mathbf{x}_{\text{adv}})\|_2^2\right] \tag{4}$$

- **Gradient Alignment Loss (L_{align}):** Aligns the gradients of clean and adversarial inputs:

$$L_{\text{align}} = \mathbb{E}\left[\left\|\nabla_{\mathbf{x}_{\text{clean}}} L_{\text{CE}} - \nabla_{\mathbf{x}_{\text{adv}}} L_{\text{CE}}\right\|_2^2\right] \tag{5}$$

The final loss function is defined as follows:

$$L_{\text{total}} = L_{\text{CE}}(\mathbf{x}_{\text{clean}}, y) + L_{\text{CE}}(\mathbf{x}_{\text{adv}}, y) + \lambda_{\text{align}} L_{\text{align}} + \lambda_{\text{smooth}} L_{\text{smooth}} \tag{6}$$

Here, $\lambda_{\text{align}} = 1.0$ and $\lambda_{\text{smooth}} = 0.5$ were selected empirically.

3.8. Evaluation Strategy

Model performance was assessed using the following criteria:

- **Clean Accuracy:** Accuracy on unperturbed test samples.
- **Adversarial Accuracy:** Accuracy under adversarial attacks at varying ϵ values.
- **Baseline Comparison:** Performance relative to a standard neural network, a random forest classifier, and a Transformer-based NIDS.

As discussed in Section 6, *ADV_NN* consistently demonstrates greater robustness under adversarial conditions while maintaining high accuracy on clean data.

4. Hyperparameter Tuning and Justification

To ensure reproducibility and establish a solid foundation for the *ADV_NN* model, this section details the hyperparameters used in our experiments. These parameters were tuned to balance clean accuracy and adversarial robustness against *PGD*, *FGSM*, and *Black-Box* attacks. Evaluation results are presented in Section 6.

4.1. Hyperparameter Configuration

Key hyperparameters for both the standard neural network and *ADV_NN* are as follows:

- **Model Architecture:** A feedforward neural network with two hidden layers (128 and 64 units), *ReLU* activations, and a binary output layer.
- **Optimizer:** Adam optimizer with a learning rate of 0.001, chosen for its stability and efficiency [30].
- **Training Epochs:** All models were trained for 20 epochs using batch sizes between 32 and 64.
- **Loss Functions:**
  - *Standard NN:* Cross-entropy loss.
  - *ADV_NN:* A composite loss combining:
    - Cross-entropy for clean inputs ($L_{\text{CE, clean}}$);
    - Cross-entropy for adversarial inputs ($L_{\text{CE, adv}}$);
    - Gradient alignment loss (L_{align});
    - Feature smoothing loss (L_{smooth}).

$$L_{\text{total}} = L_{\text{CE, clean}} + L_{\text{CE, adv}} + \lambda_{\text{align}} L_{\text{align}} + \lambda_{\text{smooth}} L_{\text{smooth}} \tag{7}$$

  Hyperparameter values were explored for $\lambda_{\text{align}} \in \{0.0, 0.5, 1.0, 2.0\}$ and $\lambda_{\text{smooth}} \in \{0.0, 0.5, 1.0\}$.
- **Curriculum Training:** Perturbation strength ϵ was linearly increased over epochs using the schedule $\epsilon_t = 0.20 \cdot (t/20)$, where t is the current epoch.
- **PGD Attack:** Step size $\alpha = 0.01$, 10 iterations, evaluated at $\epsilon \in \{0.01, 0.05, 0.1, 0.15\}$.
- **FGSM Attack:** Evaluated using the same ϵ values as for *PGD*.

- **Black-Box Attack:** Simulated via *FGSM*-based transfer attacks using scaled perturbations ($0.5 \cdot \epsilon$).
- **Data Preprocessing:**
  - Categorical features (*proto*, *service*, *state*) encoded using category codes.
  - Numerical features standardized using `StandardScaler`.
  - Dataset split: 70% for training and 30% for testing, stratified by class using a fixed seed (42).
- **Software Environment:** All experiments were conducted using PyTorch 2.6.0 and Python 3.9.

4.2. TransformerNIDS Hyperparameter Configuration

To ensure a fair comparison, the same preprocessing pipeline and data splits were applied to the Transformer-based NIDS. The following hyperparameters were selected based on empirical tuning and validation performance:

- **Input Projection Size:** 128
- **Transformer Encoder Layers:** 3
- **Attention Heads:** 4
- **Feedforward Network Dimension:** 256
- **Dropout Rate:** 0.2
- **Optimizer:** Adam with a learning rate of 0.001
- **Scheduler:** ReduceLROnPlateau with factor 0.5 and patience 3
- **Batch Size:** 1000
- **Training Epochs:** Up to 20, with early stopping (patience = 5)

These parameters provided a good balance between expressiveness and generalization. All adversarial evaluations (PGD, FGSM, and Black-Box) used the same attack configurations described above.

4.3. Tuning Process and Justification

The *ADV_NN* model was fine-tuned using a grid search over λ_{align} and λ_{smooth}. These hyperparameters control the strength of gradient alignment and feature smoothing, respectively. The standard neural network used a baseline configuration with $\lambda_{\text{align}} = 0.0$ and $\lambda_{\text{smooth}} = 0.0$.

Table 1 summarizes the clean accuracy, *PGD* adversarial accuracy at $\epsilon = 0.15$, and AUC for selected configurations. The key findings are as follows:

- **Baseline:** Achieved 85.35% clean accuracy. However, adversarial accuracy dropped to 82.25%, confirming vulnerability to *PGD*.
- **Gradient Alignment:** Setting $\lambda_{\text{align}} = 1.0$ improved robustness without loss of clean accuracy. Higher values yielded diminishing returns and slight performance drops.
- **Feature Smoothing:** A moderate value of $\lambda_{\text{smooth}} = 0.5$ improved both clean and adversarial accuracy. Larger values provided minimal gains and reduced precision.
- **Best Configuration:** $\lambda_{\text{align}} = 1.0$ and $\lambda_{\text{smooth}} = 0.5$ achieved the best balance between robustness and generalization, with an AUC of 0.8474 under *PGD*.

These results demonstrate the effectiveness of regularization and curriculum-based strategies in building adversarially robust NIDS.

Table 1. Effect of λ_{align} and λ_{smooth} on Clean and Adversarial Accuracy (PGD, $\epsilon = 0.15$).

λ_{align}	λ_{smooth}	Clean Accuracy	PGD Accuracy	AUC
0.0	0.0	85.35%	82.25%	0.8402
1.0	0.5	86.55%	84.07%	0.8474
1.0	1.0	86.13%	83.91%	0.8459
2.0	0.5	85.81%	83.02%	0.8447

5. Theoretical Analysis

This section formalizes the adversarial robustness objective of the proposed *ADV_NN* model for NIDSs. It provides theoretical motivation for the loss components and the curriculum training strategy used to improve resilience against adversarial attacks such as *PGD* [14], complementing the empirical results in Section 6.

5.1. Problem Formulation

Let $f_\theta : \mathcal{X} \to \mathcal{Y}$ denote a neural network classifier for NIDS. Here, $\mathcal{X} \subseteq \mathbb{R}^d$ is the space of preprocessed inputs (e.g., normalized features from UNSW-NB15), and $\mathcal{Y} = \{0, 1\}$ represents binary labels (benign or malicious). The model is parameterized by θ.

Given an input $\mathbf{x} \in \mathcal{X}$ and its true label $y \in \mathcal{Y}$, an adversarial example $\mathbf{x}_{\text{adv}} = \mathbf{x} + \delta$ satisfies $f_\theta(\mathbf{x}_{\text{adv}}) \neq y$ under the constraint $||\delta||_\infty \leq \epsilon$.

The adversarial training objective minimizes the expected loss under worst-case perturbations:

$$\min_\theta \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \left[\max_{||\delta||_\infty \leq \epsilon} L_{\text{CE}}(f_\theta(\mathbf{x} + \delta), y) \right], \tag{8}$$

where L_{CE} is the cross-entropy loss and $\mathcal{D}$ denotes the data distribution.

The *ADV_NN* framework extends this by introducing regularization terms to promote stability in both the feature space and gradient behavior:

$$L_{\text{total}} = L_{\text{CE}}(\mathbf{x}_{\text{clean}}, y) + L_{\text{CE}}(\mathbf{x}_{\text{adv}}, y) + \lambda_{\text{align}} L_{\text{align}} + \lambda_{\text{smooth}} L_{\text{smooth}}, \tag{9}$$

with $\lambda_{\text{align}} = 1.0$ and $\lambda_{\text{smooth}} = 0.5$ as regularization weights.

5.2. Gradient Alignment Loss

The gradient alignment loss encourages similar gradient behavior for clean and adversarial inputs:

$$L_{\text{align}} = \frac{1}{N} \sum_{i=1}^{N} \left\| \nabla_{\mathbf{x}_i} L_{\text{CE}}(\mathbf{x}_i, y_i) - \nabla_{\mathbf{x}_i^{\text{adv}}} L_{\text{CE}}(\mathbf{x}_i^{\text{adv}}, y_i) \right\|_2^2, \tag{10}$$

where N is the batch size. Minimizing L_{align} reduces sensitivity to perturbations by aligning gradient directions. This leads to smoother loss surfaces and more robust decision boundaries [23].

5.3. Feature Smoothing Loss

The feature smoothing loss reduces divergence in internal feature representations:

$$L_{\text{smooth}} = \frac{1}{N} \sum_{i=1}^{N} \left\| h(\mathbf{x}_i) - h(\mathbf{x}_i^{\text{adv}}) \right\|_2^2, \tag{11}$$

where $h(\cdot)$ denotes the output of the final hidden layer. This promotes stable embeddings for perturbed inputs, enhancing generalization under attack [29].

5.4. Curriculum Training

The *ADV_NN* model employs curriculum training by linearly increasing the perturbation budget ϵ over $T = 20$ epochs:

$$\epsilon_t = \epsilon_{\max} \cdot \frac{t}{T}, \quad \epsilon_{\max} = 0.2, \tag{12}$$

where t is the current epoch. This incremental schedule allows the model to first adapt to weak perturbations before confronting stronger ones. It improves generalization by reducing overfitting to fixed-strength attacks and better exploring the ℓ_∞ neighborhood around each input.

5.5. Theoretical Justification

The robustness improvements can be viewed through the lens of Lipschitz continuity. The gradient alignment loss constrains the gradient norm:

$$\|\nabla_{\mathbf{x}} L_{\text{CE}}(\mathbf{x}, y)\|_2 \leq K, \tag{13}$$

for some constant K, promoting smooth decision boundaries.

Likewise, feature smoothing ensures Lipschitz continuity in the feature space:

$$\|h(\mathbf{x}) - h(\mathbf{x} + \delta)\|_2 \leq L \cdot ||\delta||_2, \tag{14}$$

where $h(\cdot)$ denotes the final hidden layer output and L is a constant. These constraints reduce the model's sensitivity to input perturbations.

Although formal convergence guarantees are difficult due to non-convexity, empirical training using Adam (learning rate = 0.001) consistently yielded clean accuracies between 85 and 90% as well as adversarial accuracies between 75 and 85% across $\epsilon \in [0.01, 0.15]$ (see Section 6). These results suggest that combining L_{align} and L_{smooth} improves robustness and generalization beyond conventional adversarial training [14].

6. Results

This section evaluates the adversarially trained neural network (*ADV_NN*), the *RF* model, a standard *NN* baseline, and a Transformer-based NIDS. Performance is assessed on clean data and under three adversarial attacks: *PGD*, *FGSM*, and *Black-Box* transfer attacks. Four metrics are reported: accuracy, precision, recall, and area under the receiver operating characteristic curve (AUC).

6.1. Evaluation Metrics

- **Accuracy:** Proportion of correctly classified samples.
- **Precision:** Ratio of true positives to all predicted positives.
- **Recall:** Ratio of true positives to all actual positives.
- **AUC:** Area under the ROC curve, indicating discriminative ability.

6.2. Clean Data Performance

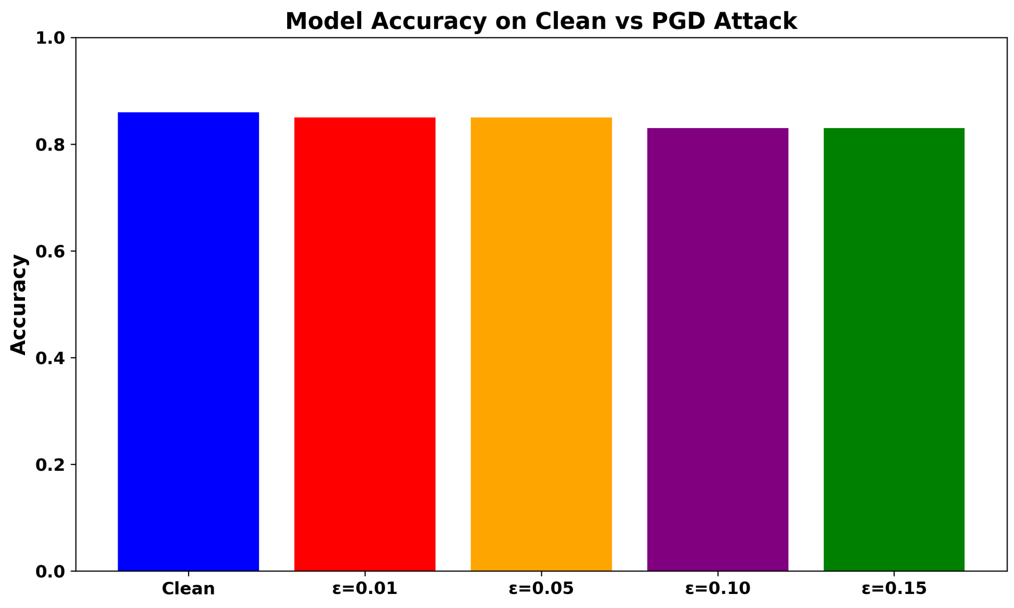
On clean test data, *ADV_NN* achieves an accuracy of **86.04%**, precision of **0.83**, recall of **0.98**, and AUC of **0.90**. The *RF* model achieves **95.18%** accuracy, **0.96** precision and recall, and an AUC of **0.99**. The baseline *NN* yields **78.56%** accuracy, **0.85** precision, **0.81** recall, and an AUC of **0.77**. The Transformer-based NIDS performs competitively, achieving **93.02%** accuracy, **0.96** precision, **0.92** recall, and an AUC of **0.99**. These results demonstrate that *RF* and Transformer models excel under clean conditions, though robustness under attack remains critical.

6.3. Adversarial Robustness Across Attacks

Model robustness was tested against adversarial attacks with perturbation levels $\epsilon \in \{0.01, 0.05, 0.10, 0.15\}$. The following subsections present the performance under each attack type.

6.3.1. PGD Attack (White-Box, Iterative)

As shown in Figure 3, *ADV_NN* retains over **83.70%** accuracy at $\epsilon = 0.15$. In comparison, *RF* achieves **49.81%**, while the baseline *NN* drops below **46.95%**. The Transformer-based NIDS achieves **61.98%** at $\epsilon = 0.15$, outperforming both *NN* and *RF*, yet still falling short of *ADV_NN*.



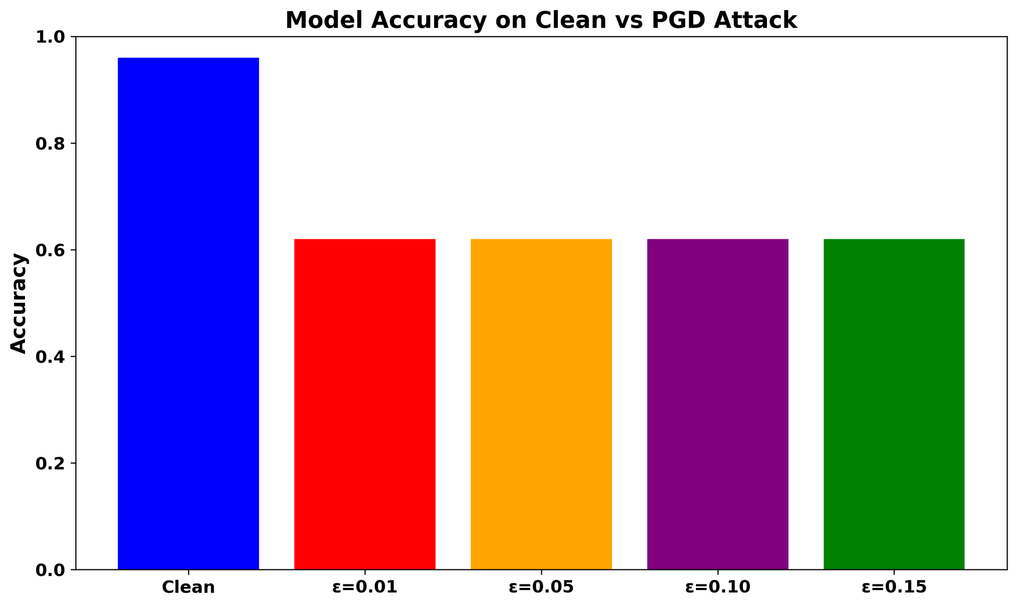
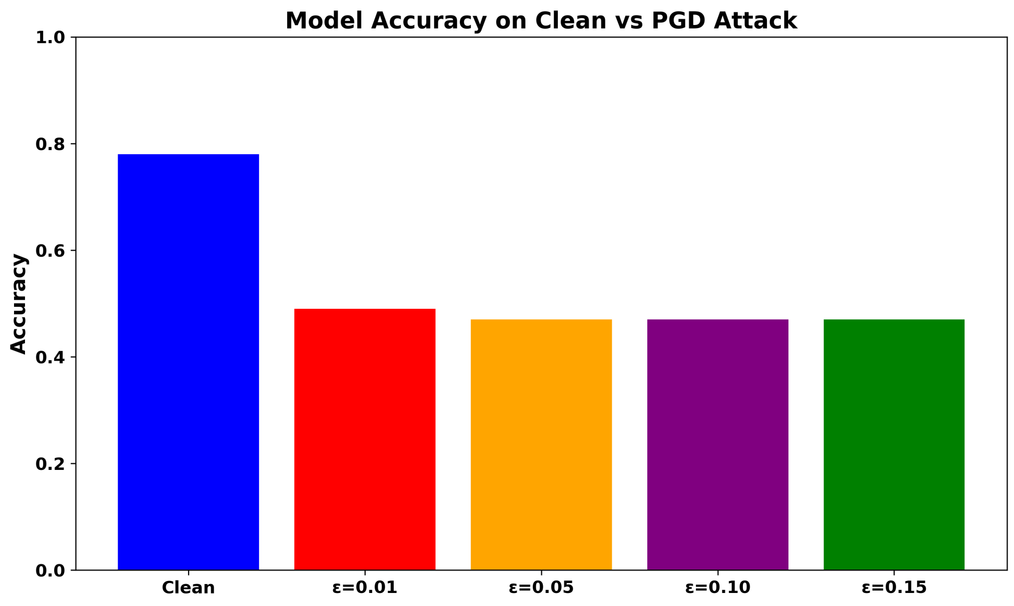


Figure 3. *Cont.*



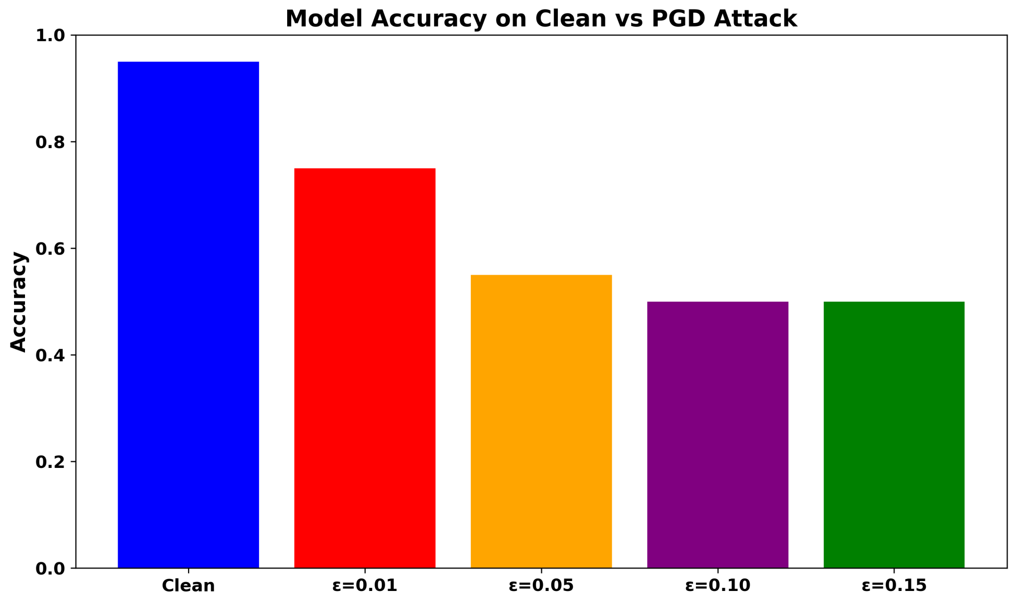


Figure 3. Accuracy under PGD Attack for *ADV_NN* (**Top**), Transformer-based model (**Upper-Middle**), *NN* (**Lower-Middle**), and *RF* (**Bottom**).

6.3.2. FGSM Attack (White-Box, One-Step)

Figure 4 shows that *ADV_NN* consistently outperforms the baseline *NN*, particularly at higher perturbation levels. While the *RF* model maintains moderate performance, its robustness declines as ϵ increases. The Transformer-based NIDS reaches **61.35%** accuracy at $\epsilon = 0.15$, showing higher recall than *NN* but lower AUC compared to *RF* and *ADV_NN*. In contrast, *ADV_NN* demonstrates strong resilience across all tested ϵ values.

6.3.3. Black-Box Attack (Transfer-Based)

To simulate a realistic *Black-Box* scenario, we implement a transfer-based attack using a substitute model trained on synthetic inputs. Specifically, synthetic samples are generated by adding Gaussian noise to a subset of clean test inputs. These samples are labeled by querying the target model (*ADV_NN*) under a fixed query budget of 10,000.

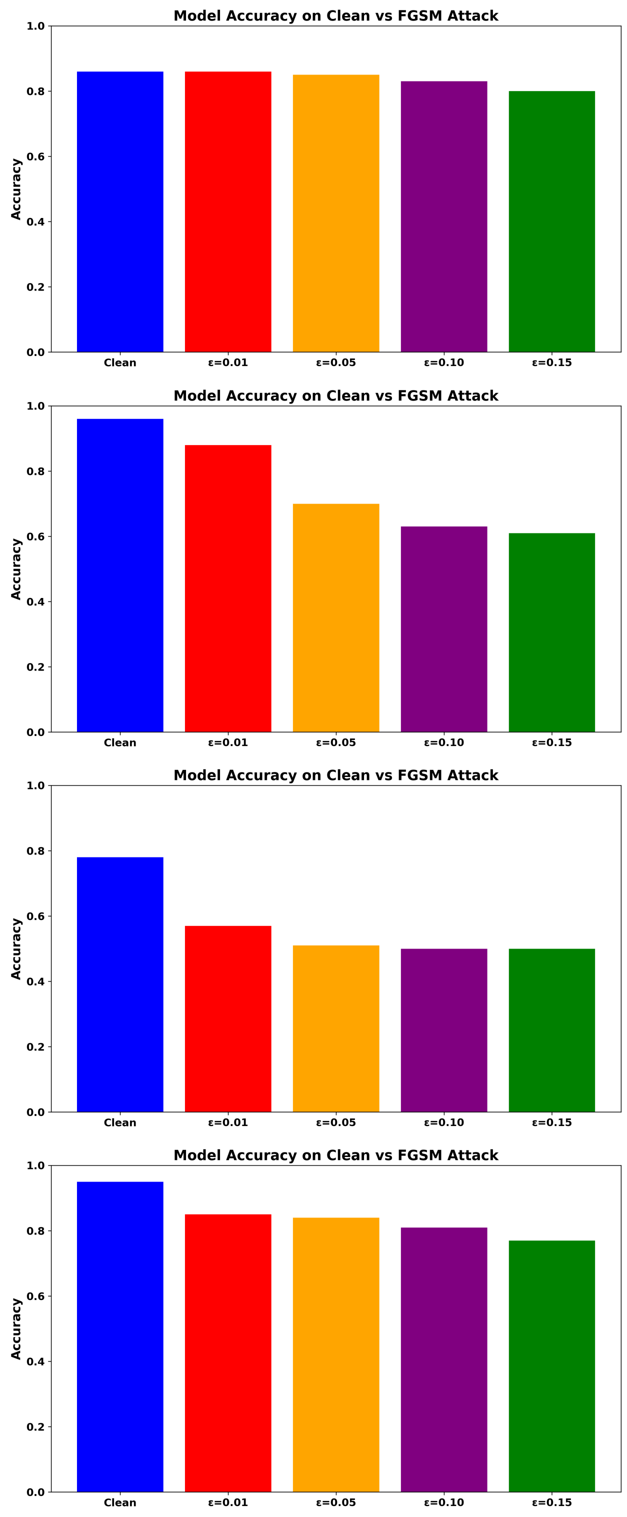


Figure 4. Accuracy under FGSM Attack for *ADV_NN* (**Top**), Transformer-based model (**Upper-Middle**), *NN* (**Lower-Middle**), and *RF* (**Bottom**).

The substitute model is a lightweight feedforward neural network with one hidden layer of 64 units and *ReLU* activations. It is trained on the synthetic input–label pairs and used to generate adversarial examples via PGD.

These adversarial examples are then evaluated against the target models, including *ADV_NN* and *RF*. As shown in Figure 5, *ADV_NN* maintains high accuracy with minimal performance degradation, demonstrating strong transfer robustness. The *RF* model also shows notable resilience, although its accuracy decreases slightly at higher perturbation levels.

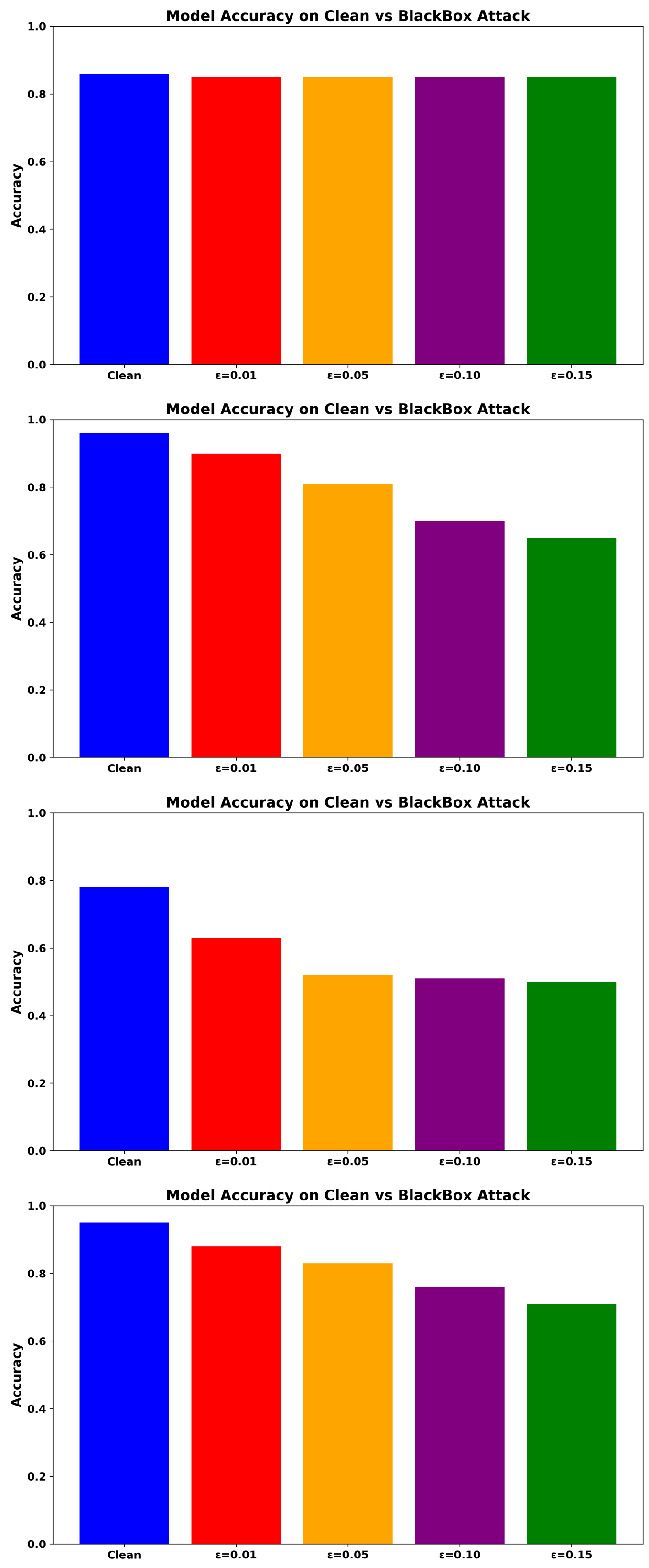


Figure 5. Accuracy under Black-Box Transfer Attack for *ADV_NN* (**Top**), Transformer-based model (**Upper-Middle**), *NN* (**Lower-Middle**), and *RF* (**Bottom**).

6.4. Comprehensive Metric Summary

Tables summarizing *Accuracy*, *Precision*, *Recall*, and *AUC* are provided separately for each adversarial attack: *PGD*, *FGSM*, and *Black-Box*. Metrics are reported for the *NN*, *ADV_NN*, *RF*, and Transformer-based models across perturbation levels $\epsilon \in \{0.01, 0.05, 0.10, 0.15\}$. The results (Tables 2–4) highlight the enhanced robustness of the adversarially trained *ADV_NN*, the moderate resilience of the *RF* model, and the partial robustness of the Transformer-based NIDS, which performs well on clean data but degrades more noticeably under adversarial perturbations.

Table 2. Performance under PGD attack.

Model	ϵ	Accuracy	Precision	Recall	AUC
NN	0.01	49.12%	0.62	0.43	0.55
NN	0.05	47.31%	0.60	0.42	0.54
NN	0.10	46.80%	0.59	0.41	0.53
NN	0.15	46.50%	0.58	0.41	0.53
Transformer-based	0.01	62.19%	0.64	0.90	0.53
Transformer-based	0.05	61.98%	0.63	0.93	0.50
Transformer-based	0.10	61.93%	0.63	0.93	0.49
Transformer-based	0.15	61.97%	0.63	0.92	0.50
ADV_NN	0.01	85.90%	0.83	0.98	0.90
ADV_NN	0.05	85.13%	0.83	0.97	0.87
ADV_NN	0.10	83.70%	0.82	0.95	0.85
ADV_NN	0.15	83.70%	0.82	0.95	0.85
RF	0.01	75.10%	0.81	0.80	0.78
RF	0.05	54.80%	0.67	0.57	0.55
RF	0.10	50.60%	0.64	0.52	0.50
RF	0.15	49.80%	0.64	0.49	0.50

Table 3. Performance under FGSM attack.

Model	ϵ	Accuracy	Precision	Recall	AUC
NN	0.01	57.12%	0.71	0.51	0.63
NN	0.05	51.14%	0.68	0.48	0.60
NN	0.10	50.00%	0.66	0.46	0.58
NN	0.15	49.78%	0.65	0.45	0.56
Transformer-based	0.01	88.30%	0.92	0.88	0.96
Transformer-based	0.05	70.57%	0.74	0.82	0.73
Transformer-based	0.10	62.94%	0.67	0.82	0.58
Transformer-based	0.15	61.35%	0.65	0.84	0.54
ADV_NN	0.01	85.90%	0.83	0.98	0.90
ADV_NN	0.05	85.15%	0.83	0.97	0.87
ADV_NN	0.10	83.77%	0.82	0.95	0.85
ADV_NN	0.15	80.49%	0.81	0.91	0.83
RF	0.01	87.94%	0.86	0.97	0.90
RF	0.05	82.86%	0.85	0.89	0.85
RF	0.10	76.98%	0.82	0.82	0.80
RF	0.15	71.57%	0.77	0.79	0.74

Table 4. Performance under Black-Box attack (Transfer-Based).

Model	ϵ	Accuracy	Precision	Recall	AUC
NN	0.01	63.54%	0.74	0.61	0.68
NN	0.05	52.35%	0.71	0.58	0.65
NN	0.10	51.29%	0.69	0.56	0.63
NN	0.15	50.42%	0.68	0.55	0.61
Transformer-based	0.01	90.47%	0.95	0.89	0.97
Transformer-based	0.05	81.04%	0.85	0.86	0.89
Transformer-based	0.10	70.77%	0.74	0.83	0.74
Transformer-based	0.15	65.40%	0.74	0.83	0.74
ADV_NN	0.01	85.99%	0.83	0.98	0.90
ADV_NN	0.05	85.83%	0.83	0.98	0.89
ADV_NN	0.10	85.38%	0.83	0.97	0.88
ADV_NN	0.15	85.12%	0.83	0.97	0.87
RF	0.01	88.01%	0.86	0.97	0.90
RF	0.05	83.07%	0.85	0.89	0.85
RF	0.10	76.80%	0.82	0.82	0.80
RF	0.15	71.70%	0.77	0.79	0.74

6.5. *ROC Curve Consistency Analysis*

Mean ROC curves were computed over five independent runs for *ADV_NN*, *NN*, the Transformer-based model, and *RF* under *PGD* attacks with $\epsilon = 0.15$. Figure 6 shows that *ADV_NN* yields a steep ROC curve with low variance, indicating strong and consistent adversarial resilience. In contrast, *RF*, *NN*, and the Transformer-based model exhibit ROC curves that are nearly diagonal with very low AUC scores and minimal variance—signaling weak and consistent failure to discriminate under attack. Among these, *RF* performs slightly better in terms of AUC, but still hovers near random guessing. These results highlight the vulnerability of conventional models and the importance of robust adversarial defenses.

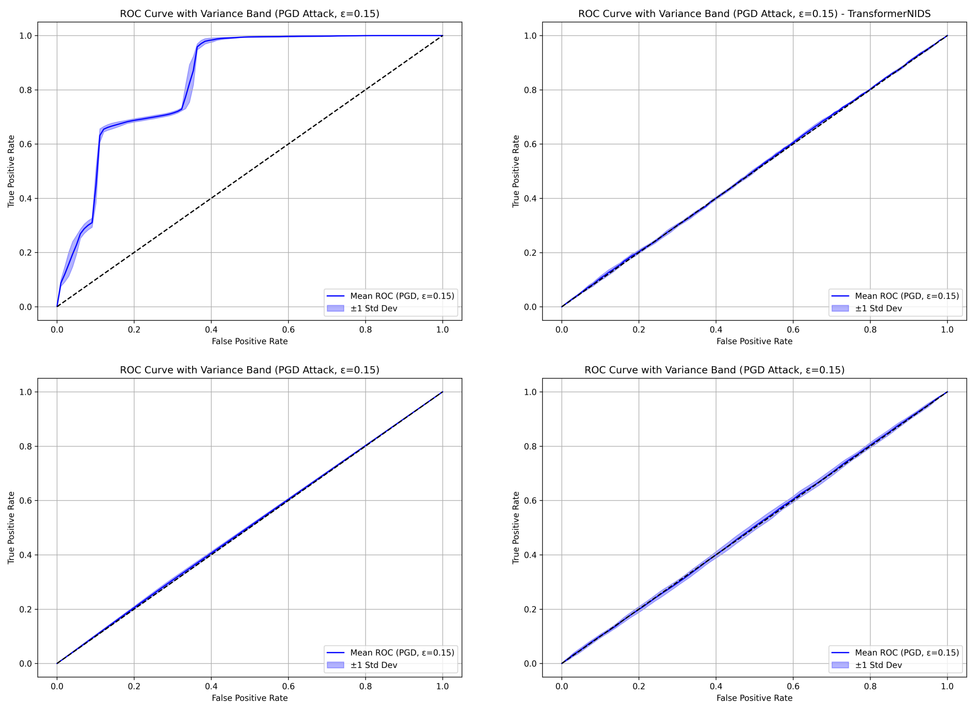


Figure 6. Mean ROC curves with shaded variance bands over five runs for PGD attack at $\epsilon = 0.15$. (**Top Left**): *ADV_NN*, (**Top Right**): *NN*, (**Bottom Left**): Transformer-based model, (**Bottom Right**): *RF*. Shaded regions indicate ±1 standard deviation.

6.6. Key Observations

- **Superior Clean Performance:** On clean data, *ADV_NN*, *RF*, and the Transformer-based model outperform the baseline *NN*. Among them, the Transformer model and *RF* achieve the highest clean accuracy and AUC.
- **PGD Robustness:** Under *PGD* attacks with $\epsilon = 0.15$, all baseline models—including *NN*, Transformer, and *RF*—suffer significant degradation. In contrast, *ADV_NN* maintains over 83% accuracy and AUC of 0.85, reflecting strong resilience.
- **FGSM Resistance:** *ADV_NN* maintains accuracy above 80% and high AUC across all ϵ levels. *RF* also performs well, showing graceful degradation. The Transformer model starts strong but deteriorates more rapidly as ϵ increases.
- **Transfer Attack Robustness:** Against transfer-based *Black-Box* attacks, *ADV_NN* again outperforms all baselines with minimal performance loss. *RF* shows competitive performance, while the Transformer model declines moderately. *NN* performs the worst across all ϵ.
- **AUC Stability:** *ADV_NN* maintains high and stable AUC values under all three attack types. *RF* performs moderately well under FGSM and Black-Box but loses effectiveness under PGD. The Transformer model exhibits high clean AUC but steep decline under PGD. The baseline *NN* degrades significantly under all adversarial settings.

6.7. Conclusion of Results

The proposed adversarial training framework significantly enhances the robustness of *ADV_NN* across a spectrum of adversarial threat models. While the *RF* model demonstrates high accuracy on clean data and moderate resilience to simpler attacks such as *FGSM* and *Black-Box* transfers, its performance deteriorates sharply under stronger, iterative attacks like *PGD*. Similarly, the Transformer-based NIDS performs well under clean conditions but experiences substantial degradation in adversarial settings, particularly under *PGD*, where its ROC curve approaches that of a random classifier. In contrast, *ADV_NN* consistently maintains stable accuracy and AUC even under worst-case perturbations, highlighting its superior robustness. These findings underscore the effectiveness of *ADV_NN* for real-world NIDS deployments where adversarial resilience is a critical requirement.

7. Discussion

The results of this study highlight a key challenge in network intrusion detection: high classification accuracy on clean data does not guarantee robustness against adversarial attacks. Prior work and our findings show that both the *RF* and standard *NN* models perform well on unperturbed inputs but suffer significant degradation when exposed to adversarial perturbations. Even small perturbations ($\epsilon = 0.01$) lead to noticeable drops in accuracy, while larger perturbations reduce performance to near-random levels. These results underscore the urgent need for more resilient machine learning-based NIDS.

The Transformer-based NIDS further illustrates this vulnerability. Although it achieves strong clean accuracy (**93.02%**), its performance drops sharply under adversarial conditions. Under *PGD* attacks with $\epsilon = 0.15$, its accuracy falls to **61.98%**, and its ROC curve becomes nearly indistinguishable from random guessing. This suggests that architectural complexity alone is insufficient for ensuring adversarial robustness.

In contrast, the proposed adversarially trained neural network (*ADV_NN*) effectively mitigates these weaknesses. It achieves a clean accuracy of **86.04%** and maintains over **80%** accuracy under *PGD* and *FGSM* attacks at $\epsilon = 0.15$. It also exceeds **85%** accuracy under *Black-Box* transfer attacks. These improvements result from the integration of adversarial training, gradient alignment, and feature smoothing, which together enhance robustness across diverse threat models.

Overall, the findings emphasize that robustness-centered training strategies are essential for deploying dependable NIDS in adversarial environments. Without such defenses, even high-performing models remain vulnerable to small but carefully crafted input manipulations.

7.1. Implications for Adversarial Robustness in NIDS

The ability of *ADV_NN* to resist *PGD*, *FGSM*, and *Black-Box* attacks marks a significant step forward in developing secure ML-based NIDS. Unlike conventional models that degrade quickly under adversarial perturbations, *ADV_NN* maintains stable performance—an essential requirement in cybersecurity, where adversaries often manipulate input features to evade detection.

The Transformer-based NIDS underscores this point. Although it achieves high accuracy on clean data, its performance declines sharply under attack, especially against *PGD*. Its near-diagonal ROC curve reveals limited discriminative capability under perturbation, indicating that architectural complexity alone does not ensure adversarial robustness.

Adversarial training improves resilience by exposing the model to perturbed examples during learning. Gradient alignment and feature smoothing further enhance robustness by stabilizing gradients and internal representations, making the model less sensitive to input changes. This multi-faceted defense aligns with current trends in adversarial machine learning, which emphasize model reliability and interpretability in the presence of attacks.

7.2. Challenges in Ensuring Robustness

Despite these improvements, several challenges remain:

Trade-off Between Robustness and Generalization: Adversarial training can lead to overfitting on specific attack types, reducing generalization to novel or unseen threats. Balancing robustness with adaptability remains a central challenge for trustworthy NIDS development.

Computational Overhead: Incorporating adversarial examples during training increases computational cost. In this study, *ADV_NN* required approximately 18.43 s to train—60% longer than the baseline *NN* (11.54 s). This overhead stems from generating adversarial inputs, computing gradients, and applying regularization terms. While acceptable for offline training, real-time deployments demand higher efficiency. Future work may explore faster training strategies such as adaptive similarity step size methods (e.g., ATSS) [31]. Although *ADV_NN* uses a standard feedforward architecture and is expected to maintain low latency, formal evaluation is needed to verify its suitability for high-throughput environments.

Adaptive Adversaries: The evaluated attacks represent common strategies. However, real-world adversaries may employ gradient-free or adaptive *Black-Box* techniques. Ensuring robust performance against evolving threats remains a pressing concern.

Complexity of Network Traffic: Network traffic is high-dimensional, context-dependent, and variable. Robustness across diverse configurations and threat scenarios requires broader evaluation on heterogeneous datasets and real-world traffic. The performance degradation observed in the Transformer-based model under attack also suggests that high-capacity architectures may be more vulnerable to complex perturbation dynamics.

7.3. Future Research Directions

Several promising avenues exist for extending this work:

- **Broader Model Benchmarking:** Future studies should benchmark *ADV_NN* against a wider range of architectures, including autoencoder-based defenses, graph neural networks, and other attention-driven models. Although the Transformer-based NIDS showed strong clean accuracy, it lacked robustness under attack. Further investigation is needed to determine whether attention mechanisms can be adapted to improve adversarial resilience.

- **Cross-Dataset Validation:** Evaluating *ADV_NN* on alternative datasets such as CICIDS2017, NSL-KDD, and real-world traffic is essential to assess generalization and practical applicability.
- **Expanded Adversarial Training:** Incorporating adaptive and gradient-free attacks (e.g., SPSA, NES) during training may strengthen defense against evolving adversarial strategies.
- **Ensemble Approaches:** Combining *ADV_NN* with other classifiers in an ensemble framework could improve detection of zero-day and evasive attacks through model diversity.
- **Explainable Robustness:** Applying explainable AI techniques (e.g., SHAP, LIME) can identify sensitive features and inform more targeted defense strategies, enhancing both interpretability and robustness.
- **Real-Time Optimization:** Techniques such as pruning, quantization, and knowledge distillation could reduce inference latency, enabling deployment in time-sensitive NIDS environments.
- **Layer-Wise Feature Alignment Analysis:** A valuable extension involves analyzing feature alignment across layers to understand internal representation shifts under adversarial perturbations. This includes computing activation similarity (e.g., cosine or Euclidean distance) between clean and adversarial inputs at different hidden layers, as well as visualizing divergence using dimensionality reduction methods like t-SNE. Such analysis can provide deeper insight into robustness beyond output metrics.

7.4. Limitations of the Study

While *ADV_NN* demonstrates clear advantages, this study has several limitations:

- **Limited Model Scope:** This work compares *ADV_NN* with *RF*, a standard *NN*, and a Transformer-based model. However, benchmarking against other architectures such as graph neural networks (GNNs), autoencoders, and hybrid attention models is needed to fully assess relative performance.
- **Dataset Dependency:** All experiments were conducted on the UNSW-NB15 dataset. Additional validation on diverse datasets is necessary to confirm generalizability and robustness in real-world settings.
- **Incomplete Attack Coverage:** The evaluation focuses on *PGD*, *FGSM*, and transfer-based *Black-Box* attacks. Robustness against adaptive, gradient-free, or real-world attack methods remains untested.
- **Training Overhead:** Adversarial training introduces computational overhead. *ADV_NN* takes longer to train due to adversarial example generation, gradient alignment, and feature smoothing. This may limit its use in time-sensitive or resource-constrained environments.
- **Unverified Adaptive Robustness:** The model has not been tested against adversaries specifically optimized to bypass its defenses. Such adaptive strategies may expose vulnerabilities not covered in this evaluation.
- **Model Complexity:** While *ADV_NN* uses a simple feedforward structure, its training pipeline is complex. The need to backpropagate through adversarial examples and multiple loss terms increases memory usage and training time. Although inference is efficient, deployment in real-time or low-resource environments may require further optimization through pruning or quantization.

7.5. Key Takeaways

- *ADV_NN* consistently outperforms baseline models—including the standard *NN*, *RF*, and Transformer-based NIDS—in adversarial robustness, while maintaining competitive accuracy on clean data.
- The combination of adversarial training, gradient alignment, and feature smoothing results in more stable predictions and improved resilience against *PGD*, *FGSM*, and transfer-based *Black-Box* attacks.
- Despite its effectiveness, challenges remain in generalization, computational efficiency, and robustness against adaptive or gradient-free adversaries. These limitations highlight key directions for future work.

Overall, the findings underscore the importance of prioritizing adversarial robustness in ML-based NIDS to ensure reliable threat detection against evolving attack strategies.

8. Conclusions

This study assessed the adversarial robustness of machine learning-based NIDS, with a focus on how neural, tree-based, and transformer-based classifiers respond to adversarial perturbations. Using the UNSW-NB15 dataset, we evaluated four models: a baseline neural network, a Random Forest, a Transformer-based NIDS, and our proposed adversarially trained neural network (*ADV_NN*). Each model was tested against three attack types: *PGD*, *FGSM*, and transfer-based *Black-Box* attacks.

The results highlight both the vulnerabilities of conventional models and the strengths of targeted adversarial defenses. Although the Transformer-based model achieved the highest accuracy on clean data, it exhibited severe degradation under adversarial conditions, underscoring the need for robustness-aware training strategies.

Key findings include the following:

- **Improved Adversarial Robustness:** *ADV_NN* consistently outperformed all other models under adversarial conditions. It maintained over **80%** accuracy under *PGD* and *FGSM* attacks at $\epsilon = 0.15$, and exceeded **85%** under transfer-based *Black-Box* attacks. In contrast, the Transformer, *RF*, and baseline *NN* models all experienced significant performance degradation, particularly under *PGD*, where their AUC values dropped to near-random levels.
- **Effectiveness of Adversarial Training:** The integration of adversarial examples, gradient alignment, and feature smoothing during training substantially enhanced robustness without significantly sacrificing clean-data performance. *ADV_NN* achieved a clean accuracy of **86.04%** while exhibiting strong resilience across all attack vectors.
- **Scalability and Practicality:** Although adversarial training introduced a 60% increase in training time (18.43s vs. 11.54s for the baseline *NN*), the resulting gains in robustness justify this overhead. Moreover, the model's compact architecture supports its feasibility for deployment in security-critical and latency-constrained environments.
- **Limitations and Future Directions:** This study focused solely on the UNSW-NB15 dataset and a limited set of adversarial scenarios. Future work will explore additional datasets (e.g., CICIDS2017, NSL-KDD), alternative attack strategies (e.g., gradient-free and adaptive methods), and diverse architectures (e.g., GNNs, autoencoders, hybrid models). Furthermore, the use of ensemble defenses, explainable AI techniques, and real-time optimization remains a promising direction.

In summary, this work demonstrates that combining adversarial training with representational and gradient-based regularization produces a robust and practical defense strategy. The proposed *ADV_NN* model provides a strong foundation for building resilient NIDS capable of operating effectively in adversarial environments.

Author Contributions: Conceptualization, V.H.; Methodology, V.H.; Software, V.H.; Formal analysis, V.H.; Writing—original draft, V.H.; Writing—review & editing, V.H. and K.N.; Visualization, V.H.; Supervision, K.N.; Project administration, V.H. and K.N.; Funding acquisition, K.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by the Center for Equitable Artificial intelligence and Machine Learning Systems (CEAMLS) at Morgan State University.

Data Availability Statement: The source code supporting the findings of this study is available from the corresponding author upon reasonable request.

Acknowledgments: This work is supported in part by the Center for Equitable Artificial intelligence and Machine Learning Systems (CEAMLS) at Morgan State University. This paper benefited from the use of OpenAI's ChatGPT (GPT 4-o) for language enhancement, including grammar corrections, rephrasing, and stylistic refinements. All AI-assisted content was subsequently reviewed and approved by the authors to ensure technical accuracy and clarity.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Xin, Y.; Kong, L.; Liu, Z.; Chen, Y.; Li, Y.; Zhu, H.; Gao, M.; Hou, H.; Wang, C. Machine Learning and Deep Learning Methods for Cybersecurity. *IEEE Access* **2018**, *6*, 35365–35381. [CrossRef]
2. Jang, M.; Lee, K. An Advanced Approach for Detecting Behavior-Based Intranet Attacks by Machine Learning. *IEEE Access* **2024**, *12*, 52480–52495. [CrossRef]
3. Fernandes Jr, G.; Rodrigues, J.J.; Carvalho, L.F.; Al-Muhtadi, J.F.; Proença Jr, M.L. A comprehensive survey on network anomaly detection. *Telecommun. Syst.* **2019**, *70*, 447–489. [CrossRef]
4. Ahmad, Z.; Shahid Khan, A.; Nisar, K.; Haider, I.; Hassan, R.; Haque, M.R.; Tarmizi, S.; Rodrigues, J.J. Anomaly detection using deep neural network for IoT architecture. *Appl. Sci.* **2021**, *11*, 7050. [CrossRef]
5. Bilot, T.; Madhoun, N.E.; Agha, K.A.; Zouaoui, A. Graph Neural Networks for Intrusion Detection: A Survey. *IEEE Access* **2023**, *11*, 49114–49139. [CrossRef]
6. Apruzzese, G.; Andreolini, M.; Ferretti, L.; Marchetti, M.; Colajanni, M. Modeling Realistic Adversarial Attacks against Network Intrusion Detection Systems. *Digit. Threat.* **2022**, *3*, 1–19. [CrossRef]
7. Yu, J.; Shvetsov, A.V.; Hamood Alsamhi, S. Leveraging Machine Learning for Cybersecurity Resilience in Industry 4.0: Challenges and Future Directions. *IEEE Access* **2024**, *12*, 159579–159596. [CrossRef]
8. Moustafa, N.; Slay, J. UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, Australia, 10–12 November 2015; pp. 1–6. [CrossRef]
9. Moustafa, N.; Slay, J. The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Inf. Secur. J. Glob. Perspect.* **2016**, *25*, 18–31. [CrossRef]
10. Moustafa, N.; Slay, J.; Creech, G. Novel Geometric Area Analysis Technique for Anomaly Detection Using Trapezoidal Area Estimation on Large-Scale Networks. *IEEE Trans. Big Data* **2019**, *5*, 481–494. [CrossRef]
11. Moustafa, N.; Creech, G.; Slay, J. Big Data Analytics for Intrusion Detection System: Statistical Decision-Making Using Finite Dirichlet Mixture Models. In *Data Analytics and Decision Support for Cybersecurity*; Springer International Publishing: Berlin/Heidelberg, Germany, 2017; pp. 127–156. [CrossRef]
12. Sarhan, M.; Layeghy, S.; Moustafa, N.; Portmann, M. NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. In *Big Data Technologies and Applications*; Springer International Publishing: Berlin/Heidelberg, Germany, 2021; pp. 117–135. [CrossRef]
13. Heydari, V.; Nyarko, K. Evaluating Network Intrusion Detection Models for Enterprise Security: Adversarial Vulnerability and Robustness Analysis. In Proceedings of the 27th International Conference on Enterprise Information Systems—Volume 1: ICEIS, INSTICC, Porto, Portugal, 4–6 April 2025; pp. 699–708. [CrossRef]
14. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
15. Ullah, F.; Javaid, Q.; Salam, A.; Ahmad, M.; Sarwar, N.; Shah, D.; Abrar, M. Modified Decision Tree Technique for Ransomware Detection at Runtime through API Calls. *Sci. Program.* **2020**, *2020*, 8853371. [CrossRef]
16. Khammas, B.M. Ransomware detection using random forest technique. *ICT Express* **2020**, *6*, 325–331. [CrossRef]
17. Akhtar, M.S.; Feng, T. Malware Analysis and Detection Using Machine Learning Algorithms. *Symmetry* **2022**, *14*, 2304. [CrossRef]

18. Ghouti, L.; Imam, M. Malware classification using compact image features and multiclass support vector machines. *IET Inf. Secur.* **2020**, *14*, 419–429. [CrossRef]
19. Arunkumar, M.; Kumar, K.A. GOSVM: Gannet optimization based support vector machine for malicious attack detection in cloud environment. *Int. J. Inf. Technol.* **2023**, *15*, 1653–1660. [CrossRef]
20. Madani, H.; Ouerdi, N.; Boumesaoud, A.; Azizi, A. Classification of ransomware using different types of neural networks. *Sci. Rep.* **2022**, *12*, 4770. [CrossRef]
21. Arivudainambi, D.; Varun Kumar, K.A.; Sibi Chakkaravarthy, S.; Visu, P. Malware traffic classification using principal component analysis and artificial neural network for extreme surveillance. *Comput. Commun.* **2019**, *147*, 50–57. [CrossRef]
22. elShehaby, M.; Matrawy, A. Introducing Perturb-ability Score (PS) to Enhance Robustness Against Problem-Space Evasion Adversarial Attacks on Flow-based ML-NIDS. *arXiv* **2024**, arXiv:2409.07448.
23. Zhang, C.; Costa-Pérez, X.; Patras, P. Adversarial Attacks Against Deep Learning-Based Network Intrusion Detection Systems and Defense Mechanisms. *IEEE/ACM Trans. Netw.* **2022**, *30*, 1294–1311. [CrossRef]
24. Holla, H.; Polepalli, S.R.; Sasikumar, A.A. Adversarial Threats to Cloud IDS: Robust Defense With Adversarial Training and Feature Selection. *IEEE Access* **2025**, *13*, 84992–85003. [CrossRef]
25. Kumar, V.; Kumar, K.; Singh, M.; Kumar, N. NIDS-DA: Detecting functionally preserved adversarial examples for network intrusion detection system using deep autoencoders. *Expert Syst. Appl.* **2025**, *270*, 126513. [CrossRef]
26. Wang, M.; Yang, N.; Gunasinghe, D.H.; Weng, N. On the Robustness of ML-Based Network Intrusion Detection Systems: An Adversarial and Distribution Shift Perspective. *Computers* **2023**, *12*, 209. [CrossRef]
27. Barik, K.; Misra, S. A Comprehensive Defense Approach of Deep Learning-Based NIDS against Adversarial Attacks. *Multimed. Tools Appl.* **2025**. [CrossRef]
28. Maseer, Z.K.; Yusof, R.; Bahaman, N.; Mostafa, S.A.; Foozy, C.F.M. Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset. *IEEE Access* **2021**, *9*, 22351–22370. [CrossRef]
29. Apruzzese, G.; Laskov, P.; Montes de Oca, E.; Mallouli, W.; Brdalo Rapa, L.; Grammatopoulos, A.V.; Di Franco, F. The Role of Machine Learning in Cybersecurity. *Digit. Threat.* **2023**, *4*, 1–38. [CrossRef]
30. Xiao, N.; Hu, X.; Liu, X.; Toh, K.C. Adam-family methods for nonsmooth optimization with convergence guarantees. *J. Mach. Learn. Res.* **2024**, *25*, 1–53.
31. Zhao, J.C.; Ding, J.; Sun, Y.Z.; Tan, P.; Ma, J.E.; Fang, Y.T. Avoiding catastrophic overfitting in fast adversarial training with adaptive similarity step size. *PLoS ONE* **2025**, *20*, e0317023. [CrossRef] [PubMed]