engineering proceedings

MDPI

Proceeding Paper

Adversarial Attacks in IoT: A Performance Assessment of ML and DL Models †

Hamza Jamiri * and Abdellah Zyane

LAPSSII Laboratory, EST SAFI, Cadi Ayyad University, Marrakech 40000, Morocco; a.zyane@uca.ac.ma

* Correspondence: h.jamiri.ced@uca.ac.ma

† Presented at the 7th edition of the International Conference on Advanced Technologies for Humanity (ICATH 2025), Kenitra, Morocco, 9–11 July 2025.

Abstract

The rise of IoT devices has led to significant advancements but also new security challenges. This paper assesses the performance of various machine learning (ML) models—Decision Trees, Naïve Bayes, Support Vector Machines (SVMs), and a deep learning model (CNN)—against adversarial attacks using the IoT-23 dataset. Attacks tested include the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). Findings show that Decision Trees are the most robust, while CNNs are the most vulnerable, highlighting the need for improved defenses in IoT systems and suggesting new research avenues in adversarial learning.

Keywords: adversarial attacks; machine learning; deep learning; IoT security; fast gradient sign method (FGSM); projected gradient descent (PGD)

1. Introduction

The Internet of Things (IoT) is rapidly expanding, with over 30 billion devices expected by 2025. This growth benefits areas like smart cities and healthcare but poses significant security challenges [1,2]. Traditional security measures like firewalls and Intrusion Detection Systems (IDSs) are inadequate for the diverse and limited-resource environment of IoT [3]. Key layers of IoT networks and their vulnerabilities are outlined in Table 1.

Table 1. IOT layers and their vulnerabilities.

Layer	Description	Security Issues
Perception	Sensing and collecting data	Physical attacks, data integrity issues
Transport	Transmitting data between devices and systems	Man-in-the-middle attacks, DoS attacks
Processing	Data analysis, management, and decision-making	Malware, buffer overflow, SQL injection
Application	Interfaces for IoT services	Software vulnerabilities, data leaks
Business	Business and management logic based on data analytics	Privacy breaches, unauthorized access

We assess the performance of a range of classical and Deep Learning (DL) algorithms, including Decision Trees, Naive Bayes, Support Vector Machines (SVMs) and Convolutional Neural Networks (CNNs), in adversarial attack settings. We use the IoT-23 dataset to evaluate their susceptibility to Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD) attacks, two widely used approaches in adversarial machine learning.

This study has two main goals:

check for updates

Academic Editors: Brahim El Bhiri and Ayman S. Mosallam

Published: 14 October 2025

Citation: Jamiri, H.; Zyane, A. Adversarial Attacks in IoT: A Performance Assessment of ML and DL Models. *Eng. Proc.* **2025**, *112*, 15. https://doi.org/10.3390/engproc2025112015

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. To explore performance degradation of ML/DL models under adversarial attacks.
2. Identify the trade-offs between model complexity, computational efficiency, and resilience and provide insights into their applicability in the IoT environment.

2. Related Work

Many studies have utilized machine learning (ML) and deep learning (DL) to tackle IoT network security issues, highlighting both advantages and limitations. A comprehensive survey [4] reviewed ML algorithms like SVM, RF, and DNN for IoT intrusion detection, noting their effectiveness in identifying malicious activities and the need for lightweight models for resource-constrained devices. Additionally, a review [5] stressed the importance of integrating ML with existing security tools to enhance detection accuracy and response times.

On the one hand, Ref. [6] introduced adversarial examples, crafted inputs that confuse ML systems. Research shows slight perturbations can significantly impact model results. IoT devices, due to limited resources and protocols, are particularly vulnerable to these attacks [7]. In Ref. [8], adversarial training and defensive distillation were studied to enhance DL model immunity against such perturbations, emphasizing the need for robust ML models.

Recent advancements focus on ensemble learning to improve detection accuracy. An ensemble-based IDS architecture proposed in Ref. [9] merges various ML methods, demonstrating superior detection rates and robustness. Similarly, Ref. [10] introduced the RobEns framework, utilizing ensemble methods to strengthen IoT IDS against adversarial attacks.

Various defense mechanisms have been developed, including techniques discussed in Refs. [8,11], such as adversarial training and feature squeezing, aimed at increasing model robustness. Despite these efforts, a unified security mechanism integrating both attack and defense strategies in IoT design remains absent.

3. Materials and Methods

3.1. Choice of Dataset

Choosing the right database is crucial for IoT security analysis. The IoT-23 dataset is widely used, containing various IoT device network traffic types, including benign and malicious types like DDoS, botnet, and malware. It is particularly suited for IDS training due to its extensive features [12,13].

The TON_IoT dataset offers telemetry data from different IoT devices and logs, providing insights into the IoT ecosystem and various cyber-attacks, making it useful for assessing machine learning (ML) model robustness [1,14].

The N-BaIoT dataset focuses on IoT botnet attacks, covering data from nine device types and offering a thorough understanding of botnet behavior, which aids in developing detection tools [2,15].

For this study, the IoT-23 dataset was chosen for its comprehensive coverage of attack types and scenarios, aligning with our research goals. It is preferred over TON_IoT and N-BaIoT due to its generalizability across various attacks, enhancing the reliability of our findings.

The objective is to leverage the IoT-23 dataset to analyze ML model performance in detecting and preventing cyber-assaults in IoT, thereby improving the validity and reliability of our research outcomes.

3.2. Data Preprocessing

Cleaning and normalizing the dataset must be carried out before applying machine learning models. This preprocessing prepares the dataset for both training and evaluation and sets the stage for the models to learn effectively from the data provided.

3.2.1. Data Cleaning

- *Elimination of Unnecessary and Empty Fields*: There are records with empty values and unnecessary columns in the IoT-23 dataset (such as timestamp, flow IDs, etc.) that make no contribution to the model's ability to differentiate between normal and malicious traffic. The records and columns were removed to obtain better quality and reduce the noise in the dataset.
- *Handling Missing Data:* Where data points were missing, they were filled using mean/mode imputation methods or removed altogether when they were deemed unnecessary for the learning process.

3.2.2. Data Normalization

- *Feature Scaling:* The dataset has varied feature ranges (e.g., bytes, packet sizes). Min–max normalization was applied to scale all feature values between 0 and 1, preventing any single feature from dominating the model. This normalization is crucial for algorithms like logistic regression and SVM, which are sensitive to feature scales.

3.2.3. Feature Selection

- *Removal of Unnecessary Features:* Time stamps, flow IDs, or session IDs that had no significance for the classification were removed. This reduced the dimensionality of the dataset, allowing a more efficient and easier training process for the model.
- *Key Features Used:* Features such as packet size, protocol type, duration, and origin/destination IPs were retained as they have been shown to be effective in distinguishing between normal and malicious traffic.

3.2.4. Data Balancing

- *Class Imbalance:* The IoT-23 dataset is imbalanced, featuring significantly more normal traffic instances than attack instances, which may bias model performance towards predicting normal traffic.
- *SMOTE (Synthetic Minority Over-sampling Technique):* To counteract this imbalance, we utilized SMOTE to oversample the minority class (malicious traffic) by generating synthetic samples based on nearest neighbors. This ensures a balanced dataset for training, enhancing the model's detection of malicious traffic.

As seen in Table 2, the final file, IoT23_combined_preprocessing.csv, contains 1,048,575 data points and 11 different types of attacks: FileDown-load, DDoS, Attack, Okiru, C&C, C&C-FileDownload, C&C-HeartBeat, C&C-HeartBeat-FileDownload, C&C-Mirai, C&C-Torii, and PartOfAHorizontalPortScan.

Table 2. Distribution of Attack Types.

Label	Count
PartOfAHorizontalPortScan	641,329
Benign	193,854
Okiru	163,016
DDoS	39,354
C&C	6878
Attack	3915
C&C-Heartbeat	134
C&C-FileDownload	43
C&C-Torii	30
FileDownload	13
C&C-Heartbeat-FileDownload	8
C&C-Mirai	1

3.2.5. Train–Test Split

- *Splitting the Dataset:* The dataset was divided into a training set (75%) and a testing set (25%) for machine learning. The training set trained the models, while the testing set evaluated performance on unseen data. The split was random but stratified to preserve the proportion of normal and attack instances in both sets.

3.3. Adversarial Attacks

In this study, we employ two well-known adversarial attack methods to evaluate the vulnerability of machine learning models in the context of IoT security:

3.3.1. Fast Gradient Sign Method (FGSM)

1. *Description:* FGSM is a white-box attack that uses full model knowledge to create adversarial examples by adding small perturbations to input data, guided by the loss function's gradient.
2. *Mathematical Formula:* The adversarial example x′ is generated as:

$$x' = x + \epsilon . \text{ sign } (\nabla x J(\theta, x, y)) \quad (1)$$

   where θ stands for the model parameters, ∇x is the gradient of the loss function J with respect to the input x, and ϵ is a tiny number regulating the perturbation strength.
3. *Purpose:* FGSM is used to evaluate how easily a model can be fooled by adding imperceptible changes to the input data.

3.3.2. Projected Gradient Descent (PGD)

1. *Description:* PGD iteratively applies small perturbations to generate adversarial examples, projecting them back onto a feasible set. This enhances its effectiveness over FGSM by exploring more complex adversarial spaces.
2. *Mathematical Formula:* The iterative update is given by:

$$x'_{t+1} = Proj_{x+S}\left(x'_t + \propto \times sign\left(\nabla_{x'} J\left(\theta, x', y\right)\right)\right) \quad (2)$$

   where S is the bounded perturbation space, and α is the step size.
3. *Purpose:* PGD is utilized to generate stronger and more complex attacks than FGSM, thereby enabling us to evaluate the robustness of the model against more complex attacks.

4. Performance Evaluation and Analysis

In this section, we cover the results of the algorithm and the time used to assess the anomaly and the confusion matrix for every algorithm.

4.1. Hardware and Environment Settings

The tests were carried out using Windows 10, Pycharm Community, Python 3.8, and Tensorflow 2.4 environments on a personal machine (Dell Latitude 5420, Round Rock, TX, USA) (Intel Core i5 1145G7 CPU @ 2.60 GHz, 16 GB of RAM @ 3200 MHz, Intel Iris Xe).

4.2. Evaluation of Metrics

We used the following evaluation metrics to assess the models' performance:

4.2.1. Time

It Takes for the Model to Classify the Test Set, an Important Factor for Real-Time IoT Anomaly Detection Systems

4.2.2. Precision

The Number of True Positives/(The Number of True Positives + The number of False Positives), Which Shows the Proportion of Attacks Among the Predicted Positive Cases

$$Precision = TP/(TP + FP) \quad (3)$$

4.2.3. Recall

The Ratio of True Positives to the Total of False Negatives, an Indicator of How Successfully the Model Detects Actual Attacks

$$Recall = TP/(TP + FN) \quad (4)$$

4.2.4. F1-Score

The Harmonic Means of Recall and Precision, Which Expresses a Balance Between the Risk of Recollection and Precision

$$\text{F1 Score} = 2 \times (Precision \times Recall)/(Precision + Recall) \quad (5)$$

4.3. Test Results for ML and DL

4.3.1. Phase 1 (Before Adversarial Learning Attacks)

- Convolutional Neural Networks:

Convolutional neural networks (CNNs) are deep learning models that require minimal preprocessing and mimic the structure of neurons in the human brain. They consist of layers like convolutional, pooling, fully connected, and normalization layers. Convolutional layers use hyperparameters such as kernel size, padding, and channels to process data, while pooling layers reduce data dimensions by summarizing the output of neuron clusters. In fully connected layers, each neuron receives input from all neurons in the previous layer, unlike convolutional layers where connections are more localized.

One input layer, four dense layers with ReLU activation, four dropout layers (0.2 rate), and one output layer with Softmax activation are all included in the suggested CNN model. It has 4,634,662 as described in Table 3 trainable parameters and employs the Adam optimizer.

Table 3. CNN Model Summary.

Layer (Type)	Output Shape	Number of Parameters
Input (Dense)	(None, 2000)	50,000
dense_1 (Dense)	(None, 1500)	3,001,500
dropout_1 (Dropout)	(None, 1500)	0
dense_2 (Dense)	(None, 800)	1,200,800
dropout_2 (Dropout)	(None, 800)	0
dense_3 (Dense)	(None, 400)	320,400
dropout_3 (Dropout)	(None, 400)	0
dense_4 (Dense)	(None, 150)	60,150
dropout_4 (Dropout)	(None, 150)	0
Output (Dense)	(None, 12)	1812
Total Parameters: 4,634,662		
Trainable parameters: 4,634,662		
Non-Trainable parameters: 0		

As presented in Table 4, the CNN model achieved a testing accuracy of 69 percent with an execution time of approximately 8.6 min. While CNN has lower accuracy and higher

execution time compared to Decision Trees, it is expected to perform better when applied to more complex datasets.

Table 4. CNN Results.

Training Accuracy	Training Loss	Testing Accuracy	Testing Loss
0.69344	0.86144	0.6934	0.8609
Time Cost		516 s	

- Decision Trees:

For classification problems, supervised machine learning classifiers like decision trees are employed. Nodes for dataset features, leaf nodes for outcomes, and branching for decision rules make up this structure. This model as detailed in Table 5 was the most efficient solution in this phase, with 77% accuracy in roughly 15 s, demonstrating the highest accuracy and lowest time cost.

Table 5. Decision Tree Results.

Metrics	Precision	Recall	F1-Score	Support
accuracy	-	-	0.73	361,169
macro avg	0.77	0.55	0.59	361,169
weighted avg	0.76	0.73	0.65	361,169
Time Cost		15 s		

- Naïve Bayes:

A popular supervised learning method for classification tasks, Naive Bayes is based on Bayes' theorem and makes probability-based predictions. When creating machine learning models, it is renowned for being straightforward but efficient. Table 6 shows that the Naive Bayes method took 31 s to execute and had an overall accuracy of only 44%. Naive Bayes produced the least accurate results among the examined algorithms.

Table 6. Naïve Bayes Results.

Metrics	Precision	Recall	F1-Score	Support
accuracy	-	-	0.30	361,169
macro avg	0.44	0.47	0.26	361,169
weighted avg	0.85	0.30	0.21	361,169
Time Cost		31 s		

- Support Vector Machine:

To classify data as detailed in Table 7, supervised machine learning models such as Support Vector Machines (SVMs) identify the best hyperplane to divide data points into classes. They use linear, polynomial, and RBF kernel functions to handle both linear and non-linear problems.

The model's macro-average F1-Score was 0.52, with a weighted-average F1-Score of 0.84 for the majority class. Execution lasted 17.55 h, highlighting its computational intensity, while SVM demonstrated robustness in imbalanced datasets.

Table 7. SVM Results.

Metrics	Precision	Recall	F1-Score	Support
accuracy	-	-	0.71	361,169
macro avg	0.28	0.18	0.52	361,169
weighted avg	0.84	0.71	0.84	361,169
Time Cost	63,203 s			

4.3.2. Phase 2 (Adversarial Learning Attacks)

1. Adversarial Learning Setup

1.1. Dataset

- The IoT dataset after preprocessing (as described in Phase 1).
- The dataset contains various classes of attacks, including PartOfAHorizontalPortScan, Okiru, and DDoS, along with benign traffic.

1.2. Models Under Test

- Convolutional Neural Networks (CNNs).
- Decision Trees.
- Support Vector Machines (SVMs).
- Naive Bayes.

1.3. Attack Parameters

- **FGSM**: Perturbation Strength (ϵ = 0.03). This value was chosen to introduce significant perturbations while ensuring the changes remain imperceptible to human observation. A larger ϵ might result in unrealistic alterations, while a smaller ϵ might not effectively expose vulnerabilities.
- **Projected Gradient Descent (PGD):** Perturbation Strength (ϵ 0.03). The same ϵ value as FGSM is used to maintain consistency and facilitate comparison between single-step and iterative attacks, with Step Size (α: 0.01). A smaller step size ensures that each iteration introduces gradual changes, making the attack more precise and harder to detect by the model. Number of Steps: 40. This value is selected to allow the attack to fully explore the decision boundary, creating adversarial examples that are highly effective at degrading model performance. Fewer steps might result in incomplete attacks, while significantly more steps could unnecessarily increase computation time.

2. Adversarial learning performance metrics

To evaluate the impact of adversarial learning on the performance of the tested models, both FGSM and PGD attacks were applied to the dataset. These attacks exposed vulnerabilities in the models by introducing small but effective perturbations to the input features as described on Table 8.

Key observations from the results include:

2.1. Degradation in Model Accuracy

- All models exhibited a significant drop in accuracy after the attacks, with PGD having a more pronounced impact due to its iterative nature. (CNN performance was the most impacted since it also relies heavily on exact relationships between features.)

2.2. Class-Level Effects

- The profile of class predictions changed significantly with some attack types being overrepresented owing to the Misclassification of benign and other classes of attack.
- This helps to emphasize the nature of the models and how it is vulnerable to adversarial noise in situations where data sets are not well balanced.

2.3. Computational Cost

- Minimal computations were needed to perform the FGSM attack, which made it useful in quickly testing vulnerabilities.
- In contrast, PGD notably increased the computational time due to its iterative nature that methodically explores the weaknesses in the model.

Table 8. Comparison Of Model Performance Metrics on Clean and Adversarial Data (FGSM and PGD).

Label	Count	Change
PartOfAHorizontalPortScan	641,329	+10.5%
Benign	193,854	−8.2%
Okiru	163,016	+15.3%
DDoS	39,354	+5.1%
C&C	6878	+12.7%
Attack	3915	+9.8%
C&C-Heartbeat	134	+1.5%
C&C-FileDownload	43	No Change
C&C-Torii	30	+10.0%
FileDownload	13	No Change
C&C-Heartbeat-FileDownload	8	No Change
C&C-Mirai	1	No Change

We present the quantitative results in the following table, where we compare metrics among clean data and adversarial examples generated by FGSM and PGD attacks.

3. Adversarial Attack Results

To ascertain the impact of adversarial attacks on the models' performance, the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD) were used. These assaults are designed to make small but powerful perturbations to the input data type to impair the model's performance and assess how resilient the models are to such hostile inputs.

The performance of each model against adversarial attacks in both normal and adversarial modes is displayed in the table below, along with measures such as accuracy, precision, recall, F1-score, and time cost. The performance on clean data is displayed, following FGSM and PGD, as well as the decline in performance brought on by adversarial perturbation on the Table 9.

4.3.3. Results Discussion

We compared the performance of four different models, CNN, Decision Trees, SVM, and Naive Bayes, and tested them under various conditions: (1) baseline performance and (2) performance after attacking the models (adversarial attack). The results underscore key trends and highlights:

1. Baseline Results

On clean data, Decision Trees achieved the highest accuracy at 77%, demonstrating effective classification of IoT traffic. SVM and CNN followed with 74% and 69%, respectively. In contrast, Naive Bayes scored only 30%, highlighting its limitations for complex IoT datasets.

2. Impact of Adversarial Attacks
   In each case, the adversarial attacks, especially PGD, had a strong influence on all models:
   - CNN faced the largest decrease in accuracy, dropping to 32% under PGD. This demonstrates that CNN can be deceived by adversarial examples, due to its reliance on high-dimensional features.
   - Decision Trees were the most robust with a PGD accuracy of only 52%.
   - Both SVM and Naive Bayes suffered moderate-to-severe degradation, with accuracies falling to 55% and 15%, respectively.

Table 9. Performance Degradation.

Model	Metric	Clean Data	FGSM (ϵ = 0.03)	PGD (ϵ = 0.03, α = 0.01)
CNN	Accuracy	69%	45%	32%
	Precision	0.75	0.50	0.38
	Recall	0.70	0.48	0.30
	F1-Score	0.72	0.49	0.34
	Time (seconds)	516	680	700
Decision Trees	Accuracy	77%	60%	52%
	Precision	0.82	0.65	0.57
	Recall	0.76	0.58	0.50
	F1-Score	0.79	0.61	0.54
	Time (seconds)	15	20	48
SVM	Accuracy	74%	62%	55%
	Precision	0.77	0.63	0.58
	Recall	0.71	0.55	0.50
	F1-Score	0.74	0.59	0.54
	Time (seconds)	63,203	65,502	68,201
Naive Bayes	Accuracy	30%	22%	15%
	Precision	0.35	0.29	0.22
	Recall	0.25	0.20	0.15
	F1-Score	0.29	0.23	0.18
	Time (seconds)	31	45	50

5. Conclusions and Future Work

This study examined machine learning and deep learning models against adversarial attacks on IoT networks, revealing their vulnerability to FGSM and PGD attacks. While CNNs achieved high accuracy on clean data, their performance significantly declined in adversarial scenarios. In contrast, lighter models like Decision Trees showed greater stability in low-resource IoT environments. These results highlight the need for protective measures to defend IoT devices from cyber-threats.

In future work, we will implement adversarial defense techniques to make the model more robust. More diverse adversarial attacks will be used for training to enhance robustness. We will investigate the application of feature squeezing techniques for real-time IoT settings, which require computational efficiency. Hybrid defense techniques that combine adversarial training with lightweight methods will be investigated to maintain security while minimizing additional computation. They will also be introduced within commoditized frameworks like oneM2M, facilitating their assessment in system-wide use cases. They are only tested on good records, leading to two areas of need: a real-time attack detection capability and a model that can adapt to ever-evolving attack scenarios.

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

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available on personal request.

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

References

1. Alkadi, S.; Al-Ahmadi, S.; Ben Ismail, M.M. RobEns: Robust Ensemble Adversarial Machine Learning Framework for Securing IoT Traffic. *Sensors* **2024**, *24*, 2626. [CrossRef] [PubMed]
2. Son, B.D.; Hoa, N.T.; Chien, T.V.; Khalid, W.; Ferrag, M.A.; Choi, W.; Debbah, M. Adversarial Attacks and Defenses in 6G Network-Assisted IoT Systems. *IEEE Internet Things J.* **2024**, *11*, 19168–19187. [CrossRef]
3. Vitorino, J.; Pracca, I.; Maia, E. Towards Adversarial Realism and Robust Learning for IoT Intrusion Detection and Classification. *Ann. Des Télécommunications* **2023**, *78*, 401–412. [CrossRef]
4. Sivasakthi, D.A.; Sathiyaraj, A.; Devendiran, R. HybridRobustNet: Enhancing Detection of Hybrid Attacks in IoT Networks through Advanced Learning Approach. *Clust. Comput.* **2024**, *27*, 5005–5019. [CrossRef]
5. Khazane, H.; Ridouani, M.; Salahdine, F.; Kaabouch, N. A Holistic Review of Machine Learning Adversarial Attacks in IoT Networks. *Future Internet* **2024**, *16*, 32. [CrossRef]
6. Abusitta, A. Deep Learning-Enabled Anomaly Detection for IoT Systems. *Internet Things* **2023**, *21*, 100656. [CrossRef]
7. Lv, Z.; Mehmood, I.; Vento, M.; Dao, M.-S.; Ota, K.; Saggese, A. IEEE Access Special Section Editorial: Multimedia Analysis for Internet-of-Things. *IEEE Access* **2019**, *7*, 65211–65218. [CrossRef]
8. Wang, Y.; Sun, T.; Li, S.; Yuan, X.; Ni, W.; Hossain, E.; Poor, H.V. Adversarial Attacks and Defenses in Machine Learning-Empowered Communication Systems and Networks: A Contemporary Survey. *IEEE Commun. Surv. Tutor.* **2023**, *25*, 2245–2298.
9. Alghamdi, R.; Bellaiche, M. An Ensemble Deep Learning Based IDS for IoT Using Lambda Architecture. *Cybersecurity* **2023**, *6*, 5. [CrossRef]
10. Balega, M.; Farag, W.; Wu, X.-W.; Ezekiel, S.; Good, Z. Enhancing IoT Security: Optimizing Anomaly Detection through Machine Learning. *Electronics* **2024**, *13*, 2148. [CrossRef]
11. Alzoubi, Y.I.; Mishra, A.; Topcu, A.E. Research Trends in Deep Learning and Machine Learning for Cloud Computing Security. *Artif. Intell. Rev.* **2024**, *57*, 132. [CrossRef]
12. Hussain, F.; Hussain, R.; Hassan, S.A.; Hossain, E. Machine Learning in IoT Security: Current Solutions and Future Challenges. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1686–1721. [CrossRef]
13. 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), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6.
14. Aldhaheri, A.; Alwahedi, F.; Ferrag, M.A.; Battah, A.A. Deep Learning for Cyber Threat Detection in IoT Networks: A Review. *Internet Things Cyber-Phys. Syst.* **2024**, *4*, 110–128. [CrossRef]
15. Ghaffari, A.; Jelodari, N.; Pouralish, S.; Derakhshanfard, N.; Arasteh, B. Securing Internet of Things Using Machine and Deep Learning Methods: A Survey. *Clust. Comput.* **2024**, *27*, 9065–9089. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.