I. H. Meskini, C. Alcaraz, R. Roman and J. Lopez, "Analysis of Transferable Adversarial Evasion Attack Detection in IoT and Industrial ADS", *IEEE Open Journal of the Computer Society*, vol. 7, pp. 142-153, 2025.
http://doi.org/10.1109/OJCS.2025.3649157
NICS Lab. Publications: https://www.nics.uma.es/publications

Analysis of Transferable Adversarial Evasion Attack Detection in IoT and Industrial ADS

Iman H. Meskini, Cristina Alcaraz, Rodrigo Roman, and Javier Lopez *

Abstract

Anomaly Detection Systems (ADS) are essential in Industrial and Internet of Things (IIoT) environments by identifying equipment failures, environmental anomalies, operational irregularities, and cyberattacks. However, the increasing reliance on Machine Learning and Deep Learning (DL) exposes ADS to adversarial attacks, particularly transferable evasion attacks, where Adversarial Examples (AE) crafted for one model can deceive others. Despite their importance, limited research has examined the transferability of adversarial attacks in industrial and IoT contexts or the effectiveness of defense strategies against them. This work systematically evaluates the transferability of *adversarial evasion attacks* across six ADS models, including both tree-based and neural network architectures, trained on industrial and IIoT scenarios datasets. We also analyze multiple adversarial detection methods, measuring not only their performance, but also their computational efficiency in terms of execution time, processor utilization, and energy consumption. Our results show that most ADS are vulnerable to transferable evasion attacks and that existing detection methods fail in model- and attack-agnostic settings. We further demonstrate that incorporating adversarial learning with a small set of low-perturbation examples significantly improves detection while maintaining low computational overhead, enabling practical and efficient real-time deployment.

Keywords · *Anomaly Detection Systems, Adversarial Attack, Detection, Transferability, Industrial IoT.*

1 Introduction

Evasion attacks remain a major concern for detection system design. Within the field of network security, Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) have been targeted for adversarial evasion. Attackers exploit weaknesses by manipulating rule activation pipelines or duplicating packet segments to mislead detectors [1]. In industrial settings, the literature similarly addresses evasion attacks against Anomaly Detection Systems (ADS), including stealthy attacks that gradually alter control systems to avoid alarms until failure, and replay attacks, where recorded normal sensor readings are injected to conceal malicious actions such as Denial of Service (DoS) [2].

The adoption of Machine Learning (ML) and Deep Learning (DL) for anomaly detection has introduced a new attack surface: adversarial evasion. These attacks exploit Neural Networks' (NN)

*Email: {imanb,alcaraz,rroman,jlm@lcc.uma.es}

susceptibility to subtle perturbations that alter predictions [3]. Research has shown that such attacks extend beyond Computer Vision (CV), affecting ML and DL models in domains including Network IDS (NIDS), Internet of Things (IoT), and industrial cybersecurity [4–6]. Consequently, robust detection and defense mechanisms are critical to maintain the resilience of modern detection systems. In this context, research on adversarial AI in industrial ADS is still at an early stage. Existing studies largely address ML-based fault detection, diagnosis, and electricity theft, while domains such as anomaly detection in Electric Vehicle Charging Infrastructure (EVCI) and IoT-based monitoring remain underexplored [7]. Moreover, most works assume glass-box or oracle closed-box threat models, overlooking adversarial transferability [8]. To bridge this gap, we examine transferable adversarial evasion and their detection under gray-box settings, constraining perturbations to preserve data functionality and yield realistic attack scenarios.

The transferability evaluation involves 6 target models: 4 tree-based ML methods including CatBoost, Light Gradient Boosting Machine (LGBM), Random Forest (RF) and eXtreme Gradient Boosting (XGB), and two NN, namely Multi-Layer Perceptron (MLP) and Deep NN (DNN). Four transferable adversarial attacks, Momentum Iterative Fast Gradient Sign Method (MIFGSM), Projected Gradient Descent (PGD), Translation-Invariant Method (TIM) and Diverse Input Method (DIM), are evaluated on two datasets: one with EVCI usage data for detecting anomalies such as malfunction, electricity theft and data leakage, and another with IoT sensor data from a clean room simulation testbed for monitoring air quality, temperature and humidity. We further assess state-of-the-art Adversarial example AE detection methods in industrial and IoT contexts. Therefore the **main contributions** of this paper are (i) a systematic evaluation of transferability in *adversarial evasion attacks* against 6 ADS trained on two datasets representing real-world scenarios; (ii) an extensive experimental study of AE detection techniques, assessing their effectiveness across models, attacks, and use cases; (iii) and an analysis of the computational efficiency of top detectors, including execution time, energy consumption and processor usage, to assess real-time suitability.

The remainder of this paper is organized as follows. Section 2 reviews related work; Section 3 covers adversarial evasion attacks and transferability; Section 4 outlines adversarial detection methods; Section 5 evaluates them; and Section 6 presents conclusions and future directions.

2 Related work

Adversarial attacks and defenses have been extensively studied in the CV domain [5]. Recent work extends this research to ML/DL-based NIDS, IoT ADS, and industrial settings, showing that NN vulnerabilities also affect tabular data such as network traffic and sensor measurements [4]. While adversarial detection is increasingly referenced in these domains [5], most approaches remain developed for CV, highlighting the early stage of research on non-visual data. Nevertheless, few studies have specifically addressed AE detection crafted to evade anomaly detection in industrial or network environments. For instance, Elgarhy *et al.* [9] propose ensemble-based detectors for smart grids, combining majority voting or stacking with adversarial training and clustering. Their method outperforms benchmarks in accuracy and resilience but remains limited to glass-box scenarios, leaving vulnerability to transferable gray- and closed-box attacks. Likewise, in [10] the authors propose a robust ensemble-based electricity theft detector trained on smart meter readings that resists evasion attacks and maintains performance across glass-, gray-, and closed-box settings. This approach allows a single ADS to detect both anomalies and attacks, avoiding separate detectors. However, the ensemble's complexity may increase computational demands, requiring validation under real-time conditions. Also, Babadi *et al.* [11] propose an ensemble framework combining

RF, AdaBoost, Decision Trees, and a Convolutional NN (CNN) to detect decision-based closed-box attacks on industrial IDS. Trained on clean and AEs, it achieves 98–99% detection accuracy and improved robustness. However, evaluation is limited to two attack types with oracle access, without considering transferability, restricting applicability to realistic scenarios.

Lastly, although Wang *et al.* [12] focus on NIDS for general network infrastructures rather than industrial or IoT-based ADS, they introduce MANifold and Decision boundary-based Adversarial example detection (MANDA), an adversarial detection method combining statistical analysis with ML. MANDA leverages inconsistencies between manifold evaluation and model inference, along with model uncertainty under small perturbations, to detect problem-based adversarial attacks. Experimental results on NSL-KDD and MNIST datasets show higher accuracy and improved robustness compared to conventional statistical baselines. Nonetheless, the approach is limited to glass-box scenarios and remains untested against transferable attacks, which are more representative of realistic adversarial conditions.

To the best of our knowledge, only a limited number of studies have explored adversarial evasion or adversarial detection in this domain, and while results show high detection rates, key gaps remain in the current literature. While constrained or problem-based attacks are increasingly being considered, transferable attacks under realistic constraints are rarely addressed. Moreover, evaluations on use cases across different scenarios are lacking. The effectiveness of adversarial detection for anomaly detection is largely unexplored. This paper addresses these gaps by assessing the robustness of multiple ADS models, generating constrained transferable AEs on industrial and IoT datasets, and comparing state-of-the-art detection techniques in terms of accuracy, detection rate, false positives, and computational performance for real-time deployment.

3 Transferability of adversarial examples

This section overviews adversarial attacks and transferability, outlines the transfer attacks, presents the industrial and IoT datasets, defines the threat model, and concludes with an evaluation of ADS vulnerabilities under the crafted attacks.

3.1 Background on Evasion Attacks and Transferability

Evasion attacks are imperceptible perturbations crafted to deceive AI/ML models with the objective to generate incorrect predictions on test time [3]. These perturbed inputs are called AEs. Given an input x and a classifier $f(\cdot)$ with prediction $y = f(x)$, an attacker seeks a minimum perturbation δ such that $f(x + \delta) \neq f(x)$ while $\|\delta\|$ remains constrained. AEs can be generated under varying assumptions about an attacker's knowledge, but a key property enabling attacks in closed box and gray box scenarios is transferability, which is the ability of an AE crafted for one model to remain effective against another model with different architecture or training data [8]. This property allows attackers in network security and industrial sensor applications to train a surrogate model under a glass box setting, generate AEs, and then transfer them to the target model to induce misclassification, even when the attacker has limited knowledge of the target system [4, 8]. To this end, the literature has introduced optimization techniques to improve the transferability of adversarial attacks, including MIFGSM which stabilizes gradient directions [13], PGD, TIM which optimizes perturbations over translated images [14], and DIM which applies random variations to inputs [15]. These optimizations increase the likelihood that AEs crafted on a surrogate model remain effective on unseen target models. In our threat model, we consider these optimized transferable attacks

to evaluate their impact on industrial ADS and test state-of-the-art detection methods, while constraining perturbations to ranges that are physically or operationally plausible.

3.2 Datasets

To establish a realistic threat model, this study employs two datasets for training and evaluating the ADS. The first dataset concerns an Electric Vehicle Charging Station (EVCS), which records infrastructure usage information. Since the original dataset contained only benign data, we relied on the modified version presented in [16], where random perturbations were introduced into 20% of the data and labeled as anomalous. This dataset comprises 6 features: total energy consumed, cost or fee, charge duration, session duration, connector type, and charging speed. These features represent common aspects of charging sessions and provide a realistic basis for industrial ADS and adversarial attack evaluation. The second dataset, termed ENVironment Measurements (ENVM), was collected from a custom IoT testbed designed to replicate clean room conditions like those in semiconductor manufacturing, where strict control of air quality, temperature, and humidity is required [17]. It includes 11 features, consisting of actuator values (lights, lamp, alert light), user inputs (lamp potentiometer and light potentiometer), and sensor readings (room temperature, accurate temperature, gas sensor, air quality sensor, humidity sensor, and ambient light). Anomalies were generated in real time during sensor readings, capturing events such as high temperature, excessive humidity, degraded air quality, and increased gas concentration. Labels were generated using an unsupervised detector trained on benign data to build the labeled dataset, then used to train the environmental ADS.

3.3 Threat model

The threat model of the study assumes a gray-box setting, in which the model architecture and the internal details of the detection system are unknown to the adversary. It is assumed that the adversary gets access to both training and testing data, consistent with practical scenarios in which parts of the system may be accessible to them. For the EVCS dataset, the attacker is considered to have exploited inherent vulnerabilities in the EVCI, gaining access to user usage data [18]. Similarly, for the ENVM dataset, the attacker is assumed to have compromised deployed IoT devices, obtaining access to environmental sensor readings [19].

To preserve functional behavior, perturbations are applied only to continuous features, while categorical features remain unchanged. Modified values are constrained to valid feature ranges via masking and clipping, which restrict perturbations to selected features and bound values within a predefined range.

3.4 Evaluation of attacks

Following the threat model, a surrogate NN was built to approximate the target classifier, consisting of three fully connected layers with 128 neurons in the first layer and 64 in the second, both using ReLU activation, and two output neurons for binary classification of normal versus anomalous samples. The model was trained for 10 epochs with a batch size of 32 using the Adam optimizer, a learning rate of 0.001, and cross-entropy loss. The surrogate model is trained separately on EVCS or ENVM dataset depending on the evaluated dataset. To evaluate transferability, the Detection Rate (DR) metric, equivalent to recall, was employed as defined in (1) [20], pointing out the importance

of False Negatives (FN) among True Positives (TP), critical since evasion attacks seek to increase undetected anomalous samples.

$$DR = \frac{TP}{TP} + FN \tag{1}$$

Moreover, the F1-score is employed to assess the overall performance of anomaly detection. The F1-score represents the harmonic mean of precision and DR, providing a balanced measure that reflects both false positives and false negatives. In this way, it overcomes the limitations of accuracy in imbalanced datasets, which are common in anomaly detection tasks. The definitions of F1-score and Precision are presented in (2) and (3) respectively.

$$F_1 = 2 \cdot \frac{Precision \cdot DR}{Precision + DR} \tag{2}$$

$$Precision = \frac{TP}{TP + FalsePositives} \tag{3}$$

Six ADS models were trained and evaluated against 4 transferable attacks across perturbation levels (ϵ) from 0.01 to 0.91 in steps of 0.1. While perturbations above $\epsilon = 0.3$ are usually not considered stealthy due to being perceptible to the human eye [21], they still represent stronger attacks with higher success rates in evading detection. Since prior studies have shown that defenses like *feature squeezing* and statistical methods can be bypassed under stronger perturbations [22], testing across different levels of aggressiveness allows for a more complete assessment of ADS robustness and evasion detection performance.

Tables 1 and 2 report the DR for the EVCS and ENVM datasets respectively, considering 6 different ADS trained with their corresponding datasets. Adversarial examples were transferred at test time to the target models, ranging from no attack ($\epsilon = 0$) up to the maximum perturbation strength. For the EVCS dataset, results indicate that MIFGSM, PGD, and TIM are the most effective attacks, with DR showing a clear linear decline as the perturbation magnitude increases. Even ensemble ADS such as RF or CatBoost, which are more robust in adversarial settings, were significantly affected. By contrast, the DIM attack is considerably less effective, suggesting that random input transformations are less effective against data coming from EVCI usage, where features values are restricted. The ENVM dataset presents a contrasting outcome, where tree-based classifiers, which perform best under clean conditions, experience a sharper decline in detection rate under transferable adversarial perturbations, while NNs, despite weaker clean performance, show smaller degradation. This difference arises from the dataset's design, where anomalies were generated in real time during sensor operation, producing perturbations consistent with the data distribution. Although tree-based models generally achieve higher performance on tabular data [23], their decision boundaries are more sensitive to perturbations, making them more vulnerable than NNs, whose predictions are less influenced by subtle correlations in this particular case. The present results are further contrasted in Table 3, where Attack Success Rate (ASR) in (4) for both datasets is reported per model.

$$ASR = \frac{FN}{TP + FN} \cdot 100\% \tag{4}$$

The ASR values were averaged per attack depending on the similarity, hence, $ASR_{1,3}$ represents the mean of the ASR values from MIFGSM, PGD and TIM attacks. ASR_4 represents the value for DIM attack. ASR increases as perturbation values increase, matching inversely with the linear

decline observed in the DR. Models trained on EVCS dataset are more vulnerable to transfer attacks than ENVM, since there is less evasion across perturbation levels. ASR results are also reported for $\epsilon = 0$, since there exists an inherent misclassification of the model that should be considered. Based on the preliminary ASR, for EVCS dataset starting from $\epsilon > 0.11$ values start to reach around 2% of success and, above that, evasion becomes threatening. For ENVM dataset, threatening ASR values are noticeable around $\epsilon = 0.21$, forcing the attacker to perform more aggressive attacks to have more opportunities for evasion.

The findings therefore determine that under limited knowledge, AEs transfer across model types, highlighting the importance of defense mechanisms in ML- and AI-based ADS. The following sections examine methods for detecting such attacks, evaluating their effectiveness and suitability for the presented datasets.

4 Adversarial evasion detection

This section outlines adversarial attack detection methods from CV, network, and industrial domains, adapted to sensor tabular data and evaluated in Section 5. Statistical methods are excluded because their dependence on training data limits effectiveness in industrial and IoT settings, where sensor drift and environmental shifts occur [24].

4.1 Statistical methods

Statistical methods for AE detection mainly detect if the input data distribution matches the training data in order to detect AEs [25]. Since AEs are crafted adding subtle perturbations optimized to fool the model, they are not intended to follow the benign data distribution. Hence, statistically the detection is possible. In [25] the authors detect AEs by measuring the distance between the sample and the training data manifold, while in [26] and [27] the distance through maximum mean discrepancy and kernel density estimation is measured. The main downside is that they are either expensive or not able to detect individual adversarial samples. Moreover, they might generate false positives caused by benign but out-of-distribution samples [28]. Therefore, while plausible for offline batch detection, statistical methods are not practical for real time adversarial attack detection.

4.2 Feature squeezing

Feature squeezing is an input transformation technique that reduces the feature space to limit feasible perturbations, thereby mitigating the impact of adversarial attacks by restricting the adversary's ability to generate effective AEs [29]. It has been primarily applied to protect DL models through enhanced data robustness and AE detection. For robustness, models achieve better prediction performance when adversarial inputs are squeezed preventing a higher evasion rate, while for detection, AEs can be identified by comparing predictions on original and modified inputs. The main advantage of *feature squeezing* is that it safeguards models without requiring architectural changes, making it suitable for already trained and deployed systems. In [30], the authors evaluate adaptive noise reduction methods for adversarial detection using scalar quantization and smoothing filters, showing that input modifications, such as resizing or blurring, do not degrade model performance. Firstly, one type of *feature squeezing* is *Quantization* compresses a wide range of values into fewer representations (5), while spatial smoothing modifies pixel values based on their neighbors

Table 1: DR of multiple models under several transferable adversarial attacks for EVCS dataset

Dataset	Model	Attack	0.00	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91	Avg. f1
EVCS	**CATBOOST**	MIFGSM	0.921	0.918	0.835	0.747	0.685	0.624	0.586	0.563	0.543	0.525	0.508	0.77
		PGD	0.921	0.918	0.836	0.754	0.685	0.628	0.590	0.554	0.531	0.516	0.507	*0.76*
		TIM	0.921	0.916	0.849	0.774	0.714	0.654	0.610	0.572	0.545	0.527	0.518	0.78
		DIM	0.921	0.920	0.908	0.894	0.877	0.859	0.844	0.833	0.819	0.810	0.798	0.90
	LGBM	MIFGSM	0.892	0.889	0.812	0.724	0.655	0.607	0.565	0.535	0.517	0.499	0.493	0.75
		PGD	0.892	0.889	0.813	0.727	0.656	0.609	0.568	0.529	0.506	0.490	0.479	*0.74*
		TIM	0.892	0,890	0.831	0.751	0.677	0.628	0.591	0.552	0.528	0.505	0.491	0.76
		DIM	0.892	0.892	0.886	0.866	0.845	0.827	0.811	0.797	0.787	0.779	0.761	0.88
	MLP	MIFGSM	0.927	0.923	0.886	0.830	0.776	0.730	0.682	0.648	0.629	0.608	0.597	*0.82*
		PGD	0.927	0.923	0.890	0.844	0.790	0.740	0.701	0.658	0.635	0.631	0.618	0.83
		TIM	0.927	0.923	0.889	0.842	0.792	0.744	0.702	0.664	0.643	0.621	0.605	0.83
		DIM	0.927	0.926	0.923	0.916	0.904	0.889	0.886	0.869	0.863	0.857	0.847	0.92
	RF	MIFGSM	0.910	0.908	0.828	0.744	0.677	0.627	0.573	0.541	0.522	0.503	0.492	*0.75*
		PGD	0.910	0.907	0.833	0.753	0.682	0.632	0.585	0.547	0.522	0.513	0.499	0.76
		TIM	0.910	0.908	0.841	0.773	0.711	0.657	0.613	0.571	0.540	0.522	0.511	0.77
		DIM	0.910	0.912	0.898	0.877	0.859	0.837	0.821	0.813	0.792	0.776	0.762	0.89
	XGB	MIFGSM	0.908	0.903	0.829	0.746	0.680	0.627	0.581	0.558	0.546	0.517	0.506	*0.76*
		PGD	0.908	0.903	0.834	0.756	0.688	0.629	0.591	0.555	0.532	0.513	0.504	*0.76*
		TIM	0.908	0.903	0.838	0.768	0.714	0.654	0.613	0.570	0.554	0.529	0.526	0.78
		DIM	0.908	0.907	0.900	0.884	0.868	0.853	0.830	0.824	0.813	0.803	0.798	0.90
	DNN	MIFGSM	0.725	0.723	0.680	0.633	0.600	0.569	0.551	0.531	0.516	0.509	0.499	*0.70*
		PGD	0.725	0.722	0.680	0.638	0.600	0.571	0.543	0.525	0.509	0.500	0.497	*0.70*
		TIM	0.725	0.721	0.690	0.653	0.612	0.577	0.553	0.533	0.525	0.520	0.515	0.71
		DIM	0.725	0.723	0.708	0.697	0.679	0.666	0.659	0.647	0.641	0.631	0.619	0.77

Table 2: DR of multiple models under several transferable adversarial attacks for ENVM dataset

Dataset	Model	Attack	0.00	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91	Avg. f1
ENVM	**CATBOOST**	MIFGSM	0.978	0.978	0.940	0.896	0.851	0.828	0.821	0.799	0.791	0.784	0.754	*0.91*
		PGD	0.978	0.978	0.940	0.896	0.851	0.828	0.821	0.799	0.791	0.784	0.754	*0.91*
		TIM	0.978	0.978	0.940	0.896	0.851	0.828	0.821	0.799	0.791	0.784	0.754	*0.91*
		DIM	0.978	0.978	0.933	0.888	0.881	0.836	0.843	0.821	0.784	0.776	0.761	*0.91*
	LGBM	MIFGSM	0.963	0.963	0.896	0.881	0.851	0.806	0.761	0.746	0.739	0.701	0.679	0.88
		PGD	0.963	0.963	0.896	0.881	0.851	0.806	0.761	0.746	0.739	0.701	0.679	0.88
		TIM	0.963	0.963	0.903	0.888	0.858	0.813	0.769	0.746	0.739	0.716	0.709	0.89
		DIM	0.963	0.963	0.896	0.828	0.821	0.776	0.746	0.739	0.687	0.657	0.619	*0.86*
	MLP	MIFGSM	0.582	0.582	0.567	0.560	0.560	0.545	0.530	0.500	0.500	0.500	0.493	*0.67*
		PGD	0.582	0.582	0.567	0.560	0.560	0.545	0.530	0.500	0.500	0.500	0.493	*0.67*
		TIM	0.582	0.582	0.567	0.560	0.560	0.545	0.530	0.507	0.500	0.500	0.493	*0.67*
		DIM	0.582	0.582	0.575	0.567	0.560	0.560	0.552	0.545	0.545	0.537	0.537	0.69
	RF	MIFGSM	0.955	0.955	0.903	0.866	0.843	0.799	0.769	0.701	0.701	0.687	0.664	*0.87*
		PGD	0.955	0.955	0.903	0.866	0.843	0.799	0.769	0.701	0.701	0.687	0.664	*0.87*
		TIM	0.955	0.955	0.903	0.866	0.843	0.799	0.769	0.709	0.709	0.694	0.664	*0.87*
		DIM	0,955	0.948	0.896	0.858	0.836	0.813	0.813	0.784	0.784	0.739	0.754	0.89
	XGB	MIFGSM	0.970	0.903	0.881	0.851	0.813	0.769	0.761	0.746	0.739	0.731	0.687	0.88
		PGD	0.970	0.903	0.881	0.851	0.813	0.769	0.761	0.746	0.739	0.731	0.679	0.88
		TIM	0.970	0.903	0.888	0.858	0.821	0.784	0.776	0.761	0.754	0.746	0.709	0.88
		DIM	0.970	0.903	0.866	0.828	0.776	0.731	0.731	0.761	0.709	0.672	0.612	*0.86*
	DNN	MIFGSM	0.478	0.478	0.478	0.478	0.470	0.478	0.478	0.485	0.470	0.478	0.478	*0.63*
		PGD	0.478	0.470	0.478	0.478	0.485	0.478	0.478	0.478	0.478	0.478	0.470	*0.63*
		TIM	0.478	0.478	0.470	0.478	0.478	0.470	0.478	0.478	0.478	0.478	0.470	*0.63*
		DIM	0.478	0.478	0.470	0.478	0.478	0.478	0.478	0.470	0.478	0.478	0.485	*0.63*

Table 3: ASR of multiple models under several transferable adversarial attacks for EVCS and ENVM dataset

Dataset	Model	ASR	0.00	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
EVCS	**CATBOOST**	$ASR_{1,3}$	1.22	1.30	2.38	3.69	4.69	5.62	6.37	6.92	7.33	7.52	7.61
		ASR_4	1.22	1.24	1.38	1.57	1.93	2.04	2.32	2.44	2.80	2.78	3.18
	LGBM	$ASR_{1,3}$	1.67	1.71	2.77	4.04	5.11	5.95	6.57	7.19	7.61	7.82	7.95
		ASR_4	1.67	1.65	1.77	2.06	2.42	2.58	2.80	3.12	3.23	3.28	3.61
	MLP	$ASR_{1,3}$	1.05	1.08	1.56	2.05	2.78	3.36	3.86	4.37	4.82	5.15	5.37
		ASR_4	1.05	1.05	1.07	1.19	1.34	1.41	1.60	1.78	1.98	2.07	2.26
	RF	$ASR_{1,3}$	1.39	1.48	2.54	3.67	4.71	5.55	6.29	6.87	7.35	7.53	7.69
		ASR_4	1.39	1.40	1.62	1.92	2.22	2.35	2.61	2.97	3.17	3.33	3.58
	XGB	$ASR_{1,3}$	1.41	1.49	2.43	3.62	4.54	5.45	6.20	6.82	7.23	7.47	7.59
		ASR_4	1.41	1.45	1.56	1.78	2.09	2.16	2.51	2.66	2.82	2.89	3.15
	DNN	$ASR_{1,3}$	5.11	5.19	5.70	6.21	6.67	7.06	7.41	7.74	7.94	8.12	8.25
		ASR_4	5.11	5.13	5.29	5.43	5.56	5.64	5.78	5.91	5.97	5.98	5.93
ENVM	**CATBOOST**	$ASR_{1,3}$	0.36	0.36	0.95	1.35	1.87	2.34	2.42	2.70	2.82	2.98	3.61
		ASR_4	0.36	0.36	0.71	0.71	0.95	1.55	1.67	1.67	1.90	2.14	2.50
	LGBM	$ASR_{1,3}$	0.60	0.60	1.43	1.79	2.14	3.06	3.65	3.81	3.81	4.21	4.40
		ASR_4	0.60	0.60	1.19	1.43	1.67	2.38	2.98	3.10	3.10	3.69	3.93
	MLP	$ASR_{1,3}$	6.07	6.19	6.51	6.55	6.55	6.67	6.79	6.90	7.02	7.02	7.14
		ASR_4	6.07	6.19	6.43	6.55	6.55	6.55	6.79	6.90	7.02	7.02	7.38
	RF	$ASR_{1,3}$	0.83	0.83	1.55	2.02	2.46	2.78	3.77	4.13	4.01	4.13	4.64
		ASR_4	0.83	0.83	1.31	1.43	1.31	2.02	2.50	2.86	2.62	2.74	2.74
	XGB	$ASR_{1,3}$	0.48	1.55	1.90	2.46	3.02	3.61	3.73	3.89	3.93	4.13	4.72
		ASR_4	0.48	0.95	1.55	1.79	2.38	2.74	3.10	3.10	3.33	3.69	4.29
	DNN	$ASR_{1,3}$	10.00	9.92	9.96	10.00	10.08	10.04	9.88	9.96	10.04	10.04	10.04
		ASR_4	10.00	9.88	10.00	9.88	10.00	10.24	9.88	9.88	10.00	10.00	10.00

with implementations such as median, mean, and Gaussian smoothing [29, 31]. Another approach is bit-depth reduction, which lowers image color depth without compromising interpretability and improves robust *feature squeezing* defenses [29].

$$Q(x) = Q(x)\Delta \cdot \left\lfloor \frac{x}{\Delta} + \frac{1}{2} \right\rfloor \tag{5}$$

Bit-depth reduction is a form of quantization where the bit level k controls precision and noise reduction. In CV, it refers to color depth, while for tabular continuous data values are approximated to the nearest allowed level. The reduction level is defined in (6), with higher k producing greater precision and lower k increasing noise suppression. As shown in [29], $k < 4$ causes human-perceptible distortions; thus, we set $k = 4$ ($2^4 = 16$ levels) to balance precision and feature space reduction. Data are first scaled to $[0, 1]$ using (7), squeezed as in (8), and then reverted to the original distribution, ensuring uniform quantization across features.

$$L = 2^k - 1 \tag{6}$$

$$x_{scaled} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{7}$$

$$Q_{bitdepth}(x) = \frac{\text{round}(x_{scaled} \cdot L)}{L} \cdot (x_{\max} - x_{\min}) + x_{\min} \tag{8}$$

The final explored method for *feature squeezing* is *smoothing*, which in the practice in the CV field, it results in a blurring filter due to the average of nearby pixels. However, in the case of our tabular datasets, each feature is independent. Hence, *temporal smoothing* was applied column-wise, averaging the neighboring values for each feature in periods of three seconds. In the literature, median filter is widely used, but we explored more filters such as mean, median, and gaussian filter to assess its effectiveness, as well as the differences in the detection of different attacks. Equations (9), (10), and (11) represent the *temporal smoothing* functions adapted to tabular datasets whose data is time-based. Firstly, (9) displays that for each time step $t \in T$, where T is the set of all time indices (3 seconds window in our use case), the smoothed signal $s_{\text{mean}}(x)(t)$ is obtained by averaging the values $x(\tau)$ in the local window $\mathcal{W}_t \subseteq T$ centered at t, with $|\mathcal{W}_t|$ denoting the window size. Likewise, (10) represents that for each time step $t \in T$, the smoothed signal $s_{\text{median}}(x)(t)$ is computed as the median of the values $x(\tau)$ in the local window $\mathcal{W}_t$. Finally, (11) shows that for each time step $t \in T$, the smoothed signal $s_{\text{gauss}}(x)(t)$ is obtained as a weighted average of the values in the local window $\mathcal{W}_t$, where the weights $G(\tau - t)$ follow a Gaussian distribution centered at t (12), giving higher importance to nearby points and less to distant ones.

$$s_{\text{mean}}(x)(t) = \frac{1}{|\mathcal{W}_t|} \sum_{\tau \in \mathcal{W}_t} x(\tau), \quad \forall\, t \in T \tag{9}$$

$$s_{\text{median}}(x)(t) = \text{median}\{x(\tau) \mid \tau \in \mathcal{W}_t\}, \quad \forall\, t \in T \tag{10}$$

$$s_{\text{gaussian}}(x)(t) = \sum_{\tau \in \mathcal{W}_t} x(\tau) \cdot w(\tau, t), \quad \forall\, t \in T \tag{11}$$

$$w(\tau, t) = G(\tau - t) = \frac{1}{\sqrt{2\pi}\sigma} \left(-\frac{(\tau - t)^2}{2\sigma^2}\right) \tag{12}$$

4.3 Discretization

Discretization (or binning) is an input transformation technique that reduces the feature space by grouping values into intervals. While commonly applied in traditional ML preprocessing, it has also been shown to improve robustness against adversarial attacks. Two prevalent approaches are *Equal-Width* (EW) binning, which divides the feature range into fixed-size intervals, and *Equal-Frequency* (EF) binning, which allocates samples evenly across bins, yielding variable interval widths. EW binning preserves the numeric domain but may produce sparsely populated bins under skewed distributions, whereas EF binning ensures balanced sample counts at the cost of uneven bin widths. Prior work [32] finds discretization effective for simple image datasets but prejudicial to accuracy and robustness on more complex ones. In contrast, [33] reports that, for IoT tabular data, EF binning reduces performance, while EW binning improves both accuracy and robustness on clean and adversarial inputs.

Discretization maps continuous features into a finite set of bins, reducing sensitivity to small variations. In images, it typically operates on pixel intensities, but for tabular data *discretization* methods such as EW or EF binning are already adapted since they operate per feature to address skewed distributions and preserve meaningful intervals across heterogeneous measurements. EW binning is represented in (13) where the numerical feature rage is divided into k intervals of equal size, assigning each value to its corresponding interval. EF binning is the division of the total

number of observations T into k bins so that each bin contains approximately the same number of observations, which is represented in (14).

$$Q_{\mathrm{EW}}(x) = \frac{\max(X) - \min(X)}{k} \tag{13}$$

$$Q_{\mathrm{EF}}(x) = \frac{T}{k} \tag{14}$$

For input transformation methods such as *feature squeezing* or *discretization*, AE detection is based on comparing the prediction probability distributions before and after the transformation. More specifically, one distribution corresponds to the model's output when the original sample is provided, and the other corresponds to the transformed sample. The discrepancy between these two distributions can be quantified using different distance metrics, including L_1 and L_2 norms or the Kullback–Leibler (KL) divergence [29]. Because the detection strategy in this work is model- and attack-agnostic and given that the adversarial attacks considered in this paper are constrained by the L_∞ norm, we adopt KL-divergence as the distance metric. A threshold $\tau = 0.1$ is then applied to decide whether an input is adversarial. Equation (15) defines the KL-divergence used in our detection pipeline.

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \ln \frac{P(x)}{Q(x)} \tag{15}$$

4.4 Auxiliary detectors

Auxiliary detectors comprise the training of a detector in order to detect adversarial samples. They can be either embedded in the model or deployed as a different model that classifies adversarial samples before inputting the samples to the main model. For instance, in [34] the authors propose the use of an unsupervised detector built on a class-conditional Generative Adversarial Network (GAN) trained exclusively on clean data, which reduces the computational cost of the defender since no AEs have to be generated in order to train the detector. This detector uses the reconstruction error of the image to detect the anomalies caused by AEs and it is embedded on a layer of the NN. On the other hand, under the context of Medical IoT (MIoT), in [35] the authors train a detector based on transformer architecture with AEs generated from specific medical datasets attacked with closed- and glass-box approaches achieving a high performance of the detector. Other approaches rely on simpler but still efficient ML models as in [36], where the authors train an unsupervised model (Isolation Forest) combined with an autoencoder with benign data achieving detection of AEs in a lightweight manner. In the case of supervised ML, in [37] they employ representation learning to generate new representations of the data and train a tree based ensemble auxiliary detector with eighteen different tabular datasets. Since our approach is model-agnostic and seeks simplicity in deployment within existing ADS, we evaluate supervised models (including tree-based and NN models), GAN-based semi-supervised detection, and Isolation Forest–based unsupervised learning, while excluding modifications to already deployed models, as they fall outside the scope of this study.

4.5 Other approaches

Several approaches utilize new or hybrid methods to detect AEs. Wu *et al.* use adversarial gradient directions for detection and evaluate their method on ImageNet and CIFAR-10 against state-of-

the-art detectors [38]. Based on adding noise to samples for adversarial detection, the authors consider that gradient directions used to craft AEs are sufficiently discriminative to detect them. Additionally, in [39] the authors propose a meta-learning framework using a prototypical network trained on known AEs and extend to unknown attacks via feature transformation and finally few-shot maximum a posteriori algorithm. These approaches are not aligned with the model- and attack-agnostic perspective adopted in this paper. Hence, we consider the MANDA [21], discussed in Section 2.

5 Evaluation results

5.1 Evaluation of input transformation methods

Tables 4 and 5 summarize F1-scores for adversarial detection by attack, ADS model, and dataset, with accuracy or DR included in the following descriptions for context. Overall, input transformation methods effectiveness is limited. Despite some methods like *quantization* achieving relatively high accuracy ($\sim$ 0.84–0.85), low DR and precision indicate that accuracy alone is unreliable for evaluating detection performance, especially in imbalanced anomaly detection datasets. For the EVCS dataset, *temporal smoothing* with mean and Gaussian filters improves F1-scores and DR (0.61) for the MLP model, though with higher false positives. In the ENVM dataset, *bit-depth reduction*, *median smoothing*, and EW binning perform best, with EW binning providing the best DR–false positive trade-off for MLP and bit-depth reduction for XGB. Median *temporal smoothing* benefits CatBoost but achieves DR below 0.5. These results indicate that EW binning and bit-depth reduction offer computationally efficient, easily integrated detection for ADS, while temporal smoothing and median filters could be further optimized for MLP and CatBoost models.

Since input transformation is also regarded as a robustness technique against AEs, we evaluated the F1 score to examine whether ADS trained on the EVCS and ENVM datasets demonstrate improved robustness. The results, presented in Tables 6 and 7, show that overall performance declines for most methods. *Quantization* yields slightly lower performance across both datasets, while other methods perform significantly worse, indicating that input transformations do not enhance robustness in the examined scenarios.

5.2 Evaluation of auxiliary detectors

Moving to auxiliary detectors, we evaluate both unsupervised and supervised learning approaches, since directly modifying the architecture of the main models to incorporate adversarial detection is complex and not model-agnostic. For the unsupervised approach, an Isolation Forest (IF) is trained exclusively on clean benign and anomalous data, aiming to flag outliers, which in this context correspond to AEs.

For supervised detection, ML and DL models are trained on clean and adversarial samples to identify attacks at test time. CatBoost was chosen as the ML detector for its high detection with few false positives. To limit adversarial training overhead [40], low-intensity perturbations ($0.003 \leq \epsilon \leq 0.3$) were used. Separate detectors were trained with $MIFGSM + TIM$ and $PGD + TIM$ examples to evaluate generalization to unseen attacks [41]. DL detectors consist of Keras feedforward DNNs with one hidden layer of 64 ReLU neurons and a sigmoid output, trained with Adam and binary cross-entropy. Both ML and DL detectors were trained under the same procedure, enabling systematic comparison.

Table 4: *Feature squeezing* performance (F1) (EVCS)

Dataset	Detector	Model	MIFGSM	PGD	TIM	DIM
EVCS	$Q_(x)$	CB	0.187	0.203	0.190	0.176
		LGBM	0.161	0.171	0.164	0.159
		MLP	0.155	0.173	0.154	0.123
		RF	0.090	0.108	0.093	0.082
		XGB	0.174	0.182	0.172	0.164
		DNN	0.129	0.128	0.132	0.136
	$Q_{bitdepth}(x)$	CB	0.158	0.158	0.157	0.149
		LGBM	0.161	0.162	0.158	0.130
		MLP	0.110	0.110	0.111	0.087
		RF	0.159	0.158	0.155	0.130
		XGB	0.143	0.144	0.142	0.108
		DNN	0.168	0.168	0.168	0.153
	$s_{mean}(x)$	CB	0.158	0.157	0.157	0.140
		LGBM	0.161	0.160	0.161	0.149
		MLP	0.419	0.425	0.441	**0.461**
		RF	0.172	0.173	0.175	0.182
		XGB	0.155	0.156	0.157	0.134
		DNN	0.236	0.237	0.239	0.236
	$s_{median}(x)$	CB	0.130	0.131	0.127	0.104
		LGBM	0.140	0.141	0.139	0.121
		MLP	0.199	0.199	0.202	0.198
		RF	0.169	0.171	0.171	0.175
		XGB	0.126	0.127	0.125	0.101
		DNN	0.171	0.172	0.171	0.158
	$s_{gaussian}(x)$	CB	0.193	0.192	0.192	0.176
		LGBM	0.203	0.200	0.201	0.191
		MLP	0.474	0.480	0.496	**0.517**
		RF	0.177	0.176	0.177	0.186
		XGB	0.193	0.192	0.194	175
		DNN	0.213	0.213	0.215	0.209
	$Q_{EW}(x)$	CB	0.204	0.202	0.199	0.210
		LGBM	0.152	0.153	0.149	0.118
		MLP	0.244	0.244	0.243	0.273
		RF	0.144	0.145	0.142	0.122
		XGB	0.242	0.245	0.247	0.275
		DNN	0.197	0.195	0.190	0.185
	$Q_{EF}(x)$	CB	0.171	0.170	0.168	0.148
		LGBM	0.178	0.178	0.140	0.158
		MLP	0.112	0.113	0.115	0.094
		RF	0.159	0.158	0.156	0.149
		XGB	0.172	0.172	0.170	0.146
		DNN	0.170	0.170	0.169	0.154

Other approaches for auxiliary detection were also evaluated. GAN reconstruction-based detection was implemented using a PyTorch framework, where the architecture consists of three core components: an *encoder*, a *generator (decoder)*, and a *discriminator*. The encoder maps the input data x, which lies in $\mathbb{R}^d$ (that is, a d-dimensional real-valued vector space representing the features of each sample), into a lower-dimensional latent representation z. The generator then reconstructs

Table 5: *Feature squeezing* performance (F1) (ENVM)

Dataset	Detector	Model	MIFGSM	PGD	TIM	DIM
ENVM	$Q_(x)$	CB	0.097	0.097	0.093	0.117
		LGBM	0.095	0.095	0.096	0.125
		MLP	0.000	0.000	0.000	0.000
		RF	0.085	0.085	0.091	0.122
		XGB	0.338	0.339	0.338	0.375
		DNN	0.072	0.088	0.092	0.079
	$Q_{bitdepth}(x)$	CB	0.313	0.308	0.301	0.346
		LGBM	0.635	0.635	0.641	0.607
		MLP	0.458	0.458	0.458	0.472
		RF	0.140	0.140	0.137	0.139
		XGB	**0.794**	**0.794**	0.791	0.755
		DNN	0.265	0.256	0.272	0.259
	$s_{mean}(x)$	CB	0.163	0.163	0.166	0.169
		LGBM	0.078	0.078	0.077	0.088
		MLP	0.128	0.128	0.128	0.102
		RF	0.112	0.112	0.113	0.142
		XGB	0.148	0.149	0.152	0.140
		DNN	0.150	0.147	0.142	0.133
	$s_{median}(x)$	CB	0.591	0.591	**0.593**	0.569
		LGBM	0.078	0.078	0.076	0.087
		MLP	0.497	0.498	0.495	0.513
		RF	0.103	0.103	0.110	0.142
		XGB	0.543	0.543	0.517	0.503
		DNN	0.340	0.327	0.329	0.325
	$s_{gaussian}(x)$	CB	0.073	0.073	0.073	0.077
		LGBM	0.077	0.077	0.075	0.087
		MLP	0.173	0.173	0.172	0.172
		RF	0.103	0.103	0.104	0.130
		XGB	0.097	0.097	0.095	0.113
		DNN	0.104	0.118	0.119	0.119
	$Q_{EW}(x)$	CB	0.254	0.254	0.253	0.254
		LGBM	0.178	0.178	0.180	0.176
		MLP	0.703	0.703	0.703	**0.745**
		RF	0.145	0.145	0.151	0.173
		XGB	0.252	0.252	0.252	0.245
		DNN	0.227	0.227	0.228	0.212
	$Q_{EF}(x)$	CB	0.251	0.251	0.252	0.243
		LGBM	0.248	0.249	0.252	0.242
		MLP	0.212	0.212	0.216	0.204
		RF	0.146	0.146	0.152	0.173
		XGB	0.250	0.250	0.251	0.234
		DNN	0.244	0.245	0.243	0.247

the original input from this latent code, while the discriminator learns to distinguish between genuine input-latent pairs and those produced by the generator. Training data consists of labeled benign normal and anomalous data, allowing the model to learn reconstructions that are faithful for benign inputs while producing larger reconstruction errors for anomalies. For detection, the model leverages the *reconstruction error*, defined in (16) where E is the encoder and G is the gen-

Table 6: Robustness (F1) for squeezed samples(EVCS)

Dataset	Detector	Model	MIFGSM	PGD	TIM	DIM
EVCS	$Q_(x)$	CB	0.70	0.71	0.72	0.83
		LGBM	0.67	0.67	0.68	0.79
		MLP	0.74	0.75	0.76	**0.85**
		RF	0.69	0.70	0.72	0.81
		XGB	0.69	0.69	0.70	0.81
		DNN	0.70	0.70	0.71	0.76
	$Q_{bitdepth}(x)$	CB	0.28	0.28	0.28	0.27
		LGBM	0.27	0.27	0.27	0.27
		MLP	0.27	0.27	0.27	0.26
		RF	0.28	0.28	0.28	0.28
		XGB	0.27	0.27	0.27	0.27
		DNN	0.26	0.27	0.27	0.26
	$s_{mean}(x)$	CB	0.28	0.27	0.28	0.27
		LGBM	0.29	0.29	0.29	0.29
		MLP	0.33	0.33	0.33	0.34
		RF	0.30	0.30	0.30	0.31
		XGB	0.28	0.28	0.28	0.27
		DNN	0.24	0.24	0.25	0.25
	$s_{median}(x)$	CB	0.25	0.25	0.25	0.25
		LGBM	0.24	0.24	0.24	0.24
		MLP	0.21	0.21	0.20	0.20
		RF	0.24	0.24	0.24	0.24
		XGB	0.25	0.25	0.25	0.25
		DNN	0.21	0.22	0.22	0.21
	$s_{gaussian}(x)$	CB	0.27	0.27	0.27	0.26
		LGBM	0.29	0.28	0.29	0.29
		MLP	0.28	0.28	0.27	0.27
		RF	0.30	0.30	0.31	0.30
		XGB	0.26	0.26	0.27	0.26
		DNN	0.29	0.29	0.29	0.29
	$Q_{EW}(x)$	CB	0.27	0.27	0.26	0.26
		LGBM	0.26	0.26	0.26	0.26
		MLP	0.28	0.27	0.25	0.25
		RF	0.26	0.26	0.26	0.26
		XGB	0.25	0.25	0.26	0.25
		DNN	0.28	0.27	0.26	0.25
	$Q_{EF}(x)$	CB	0.26	0.25	0.26	0.25
		LGBM	0.26	0.26	0.26	0.26
		MLP	0.25	0.25	0.25	0.25
		RF	0.26	0.26	0.26	0.26
		XGB	0.26	0.26	0.26	0.26
		DNN	0.26	0.26	0.26	0.26

erator. This measures how different the reconstructed sample is from the original input. Samples that belong to the normal data distribution are reconstructed accurately and thus produce small values of $r(x)$, whereas anomalous or out-of-distribution samples produce larger reconstruction errors. By applying a threshold τ, samples with $r(x) > \tau$ are flagged as anomalies, enabling the GAN to detect deviations from expected patterns in the data. The chosen threshold is set to $\tau = 0.6$,

Table 7: Robustness (F1) for squeezed samples (ENVM)

Dataset	Detector	Model	MIFGSM	PGD	TIM	DIM
ENVM	$Q_(x)$	CB	0.87	0.87	0.87	**0.88**
		LGBM	0.84	0.84	0.85	0.83
		MLP	0.68	0.68	0.68	0.69
		RF	0.86	0.86	0.87	0.87
		XGB	0.76	0.76	0.76	0.72
		DNN	0.63	0.63	0.63	0.63
	$Q_{bitdepth}(x)$	CB	0.60	0.60	**0.61**	0.60
		LGBM	0.59	0.59	0.59	0.59
		MLP	0.54	0.54	0.54	0.54
		RF	0.58	0.57	0.57	0.57
		XGB	0.33	0.20	0.20	0.34
		DNN	0.54	0.54	0.54	0.54
	$s_{mean}(x)$	CB	0.27	0.27	0.27	0.27
		LGBM	0.27	0.27	0.27	0.27
		MLP	0.39	0.39	0.39	0.39
		RF	0.27	0.27	0.27	0.27
		XGB	0.26	0.26	0.25	0.26
		DNN	0.46	**0.47**	0.46	0.46
	$s_{median}(x)$	CB	0.69	0.69	0.69	**0.72**
		LGBM	0.27	0.27	0.27	0.27
		MLP	0.51	0.51	0.51	0.51
		RF	0.61	0.61	0.61	0.63
		XGB	0.54	0.54	0.56	0.54
		DNN	0.36	0.36	0.36	0.36
	$s_{gaussian}(x)$	CB	0.27	0.27	0.27	0.27
		LGBM	0.27	0.27	0.27	0.27
		MLP	0.26	0.26	0.26	0.26
		RF	0.27	0.27	0.27	0.27
		XGB	0.27	0.27	0.27	0.27
		DNN	**0.52**	**0.52**	**0.52**	**0.52**
	$Q_{EW}(x)$	CB	0.02	0.02	0.02	0.02
		LGBM	0.27	0.27	0.27	0.27
		MLP	0.01	0.01	0.01	0.01
		RF	0.27	0.27	0.27	0.27
		XGB	0.27	0.27	0.27	0.27
		DNN	0.17	0.17	0.16	0.19
	$Q_{EF}(x)$	CB	0.30	0.30	0.29	0.30
		LGBM	0.27	0.27	0.27	0.27
		MLP	0.28	0.28	0.28	0.28
		RF	0.27	0.27	0.27	0.27
		XGB	0.27	0.27	0.27	0.27
		DNN	0.27	0.27	0.27	0.27

as experimental evaluation indicated that this value provides the best balance between accurate anomaly detection and a low false positive rate.

$$r(x) = \|x - G(E(x))\|_2, \tag{16}$$

Table 8 presents detection results by detector, attack, and dataset. Target ADS are excluded, as

detectors operate exclusively on AEs. Unsupervised IF and semi-supervised GAN reconstruction perform poorly, indicating that detecting out-of-distribution AEs requires more advanced methods. ML and DL models trained with AEs perform significantly better and generalize to unseen transferable attacks. CatBoost achieves the highest detection across both datasets, while the DNN performs well on EVCS, particularly with PGD and TIM training, but shows lower performance on ENVM, likely due to more realistic anomalies.

5.3 Evaluation of other methods

MANDA was described in Section 2 as a statistical adversarial detection method for NIDS. We implement MANDA on both datasets following [12]. The method computes two scores for manifold evaluation: $score_1$, measuring inconsistency between manifold evaluation and ADS output, and $score_2$, capturing variance in model confidence. Detection proceeds in two stages: (i) a logistic regression is trained on these scores, and (ii) test samples are classified using the trained model. The authors also report strong performance using $score_1$ alone, flagging a sample as adversarial when $score_1 > \tau_1$. The performance of MANDA and MAnifold-based AE detection is presented in Table 9.Similar to input transformation, these statistical methods perform weakly overall. However, the detection results for the MLP model and, especially the DNN model, are stronger on the ENVM dataset. This observation aligns with the outcomes of squeezing and binning methods, which also achieved better performance for the MLP model. These results suggest that such methods hold potential and could be further refined and optimized for NN–based ADS.

5.4 Computational Efficiency

Overall, the best-performing models are CatBoost and the DNN trained with PGD and TIM attacks. To evaluate real-time efficiency, we deployed the pretrained models in a simulation of sequential inputs over time scenario with different ADS and measured CPU utilization, execution time, and energy consumption across 10 perturbation levels and 4 attacks. Results show consistent efficiency on both EVCS and ENVM datasets: CPU usage remains around 2.5–3%, execution time below 10^{-4} seconds, and energy consumption under $4 \cdot 10^{-5}$ kWh. Such low computational and energy costs confirm their practicality in resource-constrained or real-time industrial environments. However, there are some particularities worth noting. For both datasets, the DNN model reveals occasional peaks in execution time and energy consumption at specific points when processing perturbation values associated with the TIM attack, as illustrated in Figures 1 and 2. These fluctuations suggest that the DNN model is less stable under TIM attack scenarios and may require additional resources at certain moments, making it slightly less efficient compared to the CatBoost model. Finally, Table 10 summarizes the detection results. Input transformation methods perform poorly, provide no robustness gains over raw inputs, and incur high computational cost. Supervised auxiliary detectors achieve the best performance at low cost, while unsupervised methods perform poorly. Hybrid statistical–ML detectors show moderate performance but are more computationally expensive than supervised detectors.

6 Conclusion and Future Work

Transferable *adversarial evasion attacks* on EVCI and IoT sensor environments were assessed across ADS models, confirming their vulnerabilities. Among detection methods, adversarial learning with

Table 8: Auxiliary detector performance (F1)

Dataset	Detector	MIFGSM	PGD	TIM	DIM
EVCS	**Isolation Forest**	0.33	0.33	0.33	0.34
	Catboost MIFGSM/TIM	0.93	0.93	0.93	**0.95**
	Catboost PGD/TIM	0.93	0.94	0.94	**0.95**
	DNN MIFGSM/DIM	0.85	0.86	0.85	0.89
	DNN PGD/TIM	0.89	0.90	0.90	0.92
	GAN	0.35	0.35	0.35	0.36
ENVM	**Isolation Forest**	0.42	0.42	0.40	0.40
	Catboost MIFGSM/TIM	**0.99**	**0.99**	**0.99**	**0.99**
	Catboost PGD/TIM	**0.99**	**0.99**	**0.99**	**0.99**
	DNN MIFGSM/DIM	0.62	0.62	0.62	0.62
	DNN PGD/TIM	0.66	0.66	0.65	0.66
	GAN	0.28	0.28	0.28	0.28

Table 9: Hybrid detection performance (F1) (EVCS)

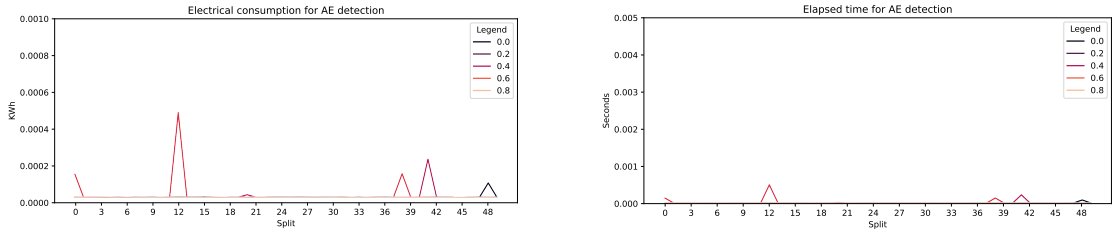
Dataset	Detector	Model	MIFGSM	PGD	TIM	DIM
EVCS	**MANDA**	CB	0.34	0.34	0.34	0.44
		LGBM	0.31	0.31	0.31	0.40
		MLP	0.35	0.36	0.34	**0.43**
		RF	0.24	0.25	0.25	0.34
		XGB	0.31	0.32	0.32	0.41
		DNN	0.22	0.22	0.23	0.27
	MANIFOLD	CB	0.34	0.35	0.35	0.45
		LGBM	0.34	0.34	0.34	0.43
		MLP	0.40	0.41	0.39	**0.47**
		RF	0.32	0.33	0.33	0.41
		XGB	0.34	0.35	0.35	0.44
		DNN	0.29	0.29	0.29	0.34
ENVM	**MANDA**	CB	0.09	0.08	0.08	0.06
		LGBM	0.01	0.00	0.01	0.02
		MLP	0.42	0.42	0.42	0.44
		RF	0.19	0.18	0.23	0.14
		XGB	0.04	0.07	0.10	0.07
		DNN	0.55	0.56	0.54	**0.57**
	MANIFOLD	CB	0.12	0.12	0.11	0.11
		LGBM	0.15	0.14	0.15	0.16
		MLP	0.43	0.43	0.43	0.46
		RF	0.11	0.10	0.10	0.08
		XGB	0.19	0.19	0.20	0.18
		DNN	0.60	0.61	0.60	**0.64**

CatBoost achieved the best performance and generalization, while input transformation and statistical approaches were less effective but demonstrated potential when combined with NN outputs. Limitations of this study include the exclusion of non–gradient-based attacks, untested robustness under concept drift with retraining, unmeasured retraining overhead, and analysis of IDS datasets, which will be addressed in future work.

Table 10: Summary table of the results

Detector		Detection	Robustness	Comp. Cost
Feature squeezing		–	–	––
Auxiliary	Supervised	++	/	++
	Unsupervised	–	/	++
Hybrid		+	/	–

++: Excellent +: Improvable –: Poor ––: Inefficient /: Not applicable

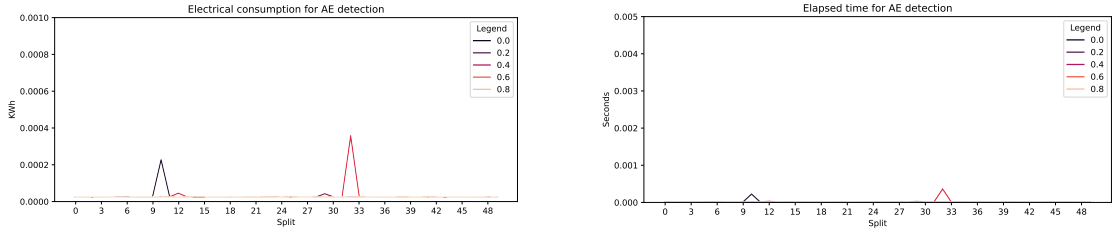


(a) DNN energy consumption (kWh) on EVCS (b) DNN execution time (seconds) on EVCS

Figure 1: DNN performance under TIM attack (EVCS)

7 Aknowledgements

This work was supported in part by the project "CiberIA: Investigación e Innovación para la Integración de Ciberseguridad e Inteligencia Artificial" (Proyecto C079/23), financed by the "European Union NextGeneration-EU, the Recovery Plan, Transformation and Resilience", through INCIBE,in part by the Partial funding for open access charge: Universidad de Málaga/CBUA, in part by the project AIAS funded by the European Union under Grant 101131292 (HORIZON-MSCA-2022-SE-01-01), and in part by the project SYNAPSE funded by the European Union under Grant 101120853 (HORIZON-CL3-2022-CS-01).



(a) DNN energy consumption (kWh) on ENVM (b) DNN execution time (seconds) on ENVM

Figure 2: DNN performance under TIM attack (ENVM)

References

[1] T.-H. Cheng, Y.-D. Lin, Y.-C. Lai, and P.-C. Lin, "Evasion techniques: Sneaking through your intrusion detection/prevention systems," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 4, pp. 1011–1020, 2012.

[2] A. Erba and N. O. Tippenhauer, "Assessing model-free anomaly detection in industrial control systems against generic concealment attacks," in *Proceedings of the 38th Annual Computer Security Applications Conference*, ser. ACSAC '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 412–426. [Online]. Available: https://doi.org/10.1145/3564625.3564633

[3] A. Vassilev, A. Oprea, A. Fordyce, H. Anderson, X. Davies, and M. Hamin, "Adversarial machine learning: A taxonomy and terminology of attacks and mitigations," National Institute of Standards and Technology, Tech. Rep., 2025.

[4] M. Mbow, R. Roman, A. Seal, K. I.-K. Wang, S. K. Mohanty, and K. Sakurai, "Investigating the transferability of evasion attacks in network intrusion detection systems considering domain-specific constraints," in *International Conference on Information Security Applications*. Springer, 2024, pp. 44–55.

[5] S. Wang, R. K. L. Ko, G. Bai, N. Dong, T. Choi, and Y. Zhang, "Evasion attack and defense on machine learning models in cyber-physical systems: A survey," *IEEE Communications Surveys & Tutorials*, vol. 26, no. 2, pp. 930–966, 2024.

[6] Y. E. Sagduyu, Y. Shi, and T. Erpek, "Iot network security from the perspective of adversarial deep learning," in *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, 2019, pp. 1–9.

[7] V. Pozdnyakov, A. Kovalenko, I. Makarov, M. Drobyshevskiy, and K. Lukyanov, "Adversarial attacks and defenses in fault detection and diagnosis: A comprehensive benchmark on the tennessee eastman process," *IEEE Open Journal of the Industrial Electronics Society*, vol. 5, pp. 428–440, 2024.

[8] R. K. Sah and H. Ghasemzadeh, "Adversarial transferability in embedded sensor systems: An activity recognition perspective," *ACM Trans. Embed. Comput. Syst.*, vol. 23, no. 2, Mar. 2024. [Online]. Available: https://doi.org/10.1145/3641861

[9] I. Elgarhy, M. M. Badr, M. M. Mahmoud, M. N. Mahmoud, M. Alsabaan, and M. I. Ibrahem, "Securing smart grid false data detectors against white-box evasion attacks without sacrificing accuracy," *IEEE Internet of Things Journal*, vol. 11, no. 20, pp. 33 873–33 889, 2024.

[10] A. Takiddin, M. Ismail, and E. Serpedin, "Robust data-driven detection of electricity theft adversarial evasion attacks in smart grids," *IEEE Transactions on Smart Grid*, vol. 14, no. 1, pp. 663–676, 2023.

[11] N. Babadi, H. Karimipour, and A. Islam, "An ensemble learning to detect decision-based adversarial attacks in industrial control systems," in *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2023, pp. 879–884.

[12] N. Wang, Y. Chen, Y. Hu, W. Lou, and Y. T. Hou, "Manda: On adversarial example detection for network intrusion detection system," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, 2021, pp. 1–10.

[13] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9185–9193.

[14] Y. Dong, T. Pang, H. Su, and J. Zhu, "Evading defenses to transferable adversarial examples by translation-invariant attacks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4312–4321.

[15] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille, "Improving transferability of adversarial examples with input diversity," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[16] J. Cumplido, C. Alcaraz, and J. Lopez, "Collaborative anomaly detection system for charging stations," in *European Symposium on Research in Computer Security*. Springer, 2022, pp. 716–736.

[17] C. ching Wan, J.-J. Lee, and L. fang Wu, "Outdoor airborne contamination control for semiconductor manufacturing process," in *2010 International Symposium on Semiconductor Manufacturing (ISSM)*, 2010, pp. 1–4.

[18] C. Alcaraz, J. Cumplido, and A. Trivino, "Ocpp in the spotlight: threats and countermeasures for electric vehicle charging infrastructures 4.0," *International Journal of Information Security*, vol. 22, no. 5, pp. 1395–1421, 2023.

[19] I. Butun, P. Österberg, and H. Song, "Security of the internet of things: Vulnerabilities, attacks, and countermeasures," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 616–644, 2020.

[20] M. Bai, P. Liu, F. Lv, D. Fang, S. Lv, W. Zhang, and L. Sun, "Adversarial attack against intrusion detectors in cyber-physical systems with minimal perturbations," in *2024 IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA)*. IEEE, 2024, pp. 816–825.

[21] S. Mandal, "Defense against adversarial attacks using convolutional auto-encoders," *arXiv preprint arXiv:2312.03520*, 2023.

[22] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 2017, pp. 3–14.

[23] D. McElfresh, S. Khandagale, J. Valverde, V. Prasad C, G. Ramakrishnan, M. Goldblum, and C. White, "When do neural nets outperform boosted trees on tabular data?" *Advances in Neural Information Processing Systems*, vol. 36, pp. 76 336–76 369, 2023.

[24] M. A. Shyaa, N. F. Ibrahim, Z. Zainol, R. Abdullah, M. Anbar, and L. Alzubaidi, "Evolving cybersecurity frontiers: A comprehensive survey on concept drift and feature dynamics aware machine and deep learning in intrusion detection systems," *Engineering Applications of Artificial Intelligence*, vol. 137, p. 109143, 2024.

[25] S. Jha, U. Jang, S. Jha, and B. Jalaian, "Detecting adversarial examples using data manifolds," in *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, 2018, pp. 547–552.

[26] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, "On the (statistical) detection of adversarial examples," *arXiv preprint arXiv:1702.06280*, 2017.

[27] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner, "Detecting adversarial samples from artifacts," *arXiv preprint arXiv:1703.00410*, 2017.

[28] W. Chen, R. A. Yeh, S. Mou, and Y. Gu, "Leveraging perturbation robustness to enhance out-of-distribution detection," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 4724–4733.

[29] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," *arXiv preprint arXiv:1704.01155*, 2017.

[30] B. Liang, H. Li, M. Su, X. Li, W. Shi, and X. Wang, "Detecting adversarial image examples in deep neural networks with adaptive noise reduction," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 1, pp. 72–85, 2021.

[31] J. Wang, J. Zhao, Q. Yin, X. Luo, Y. Zheng, Y.-Q. Shi, and S. K. Jha, "Smsnet: A new deep convolutional neural network model for adversarial example detection," *IEEE Transactions on Multimedia*, vol. 24, pp. 230–244, 2022.

[32] J. Chen, X. Wu, V. Rastogi, Y. Liang, and S. Jha, "Towards understanding limitations of pixel discretization against adversarial attacks," in *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2019, pp. 480–495.

[33] A. Namvar, C. Thapa, and S. S. Kanhere, "Discretization-based ensemble model for robust learning in iot," in *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Springer, 2023, pp. 353–367.

[34] H. Wang, D. J. Miller, and G. Kesidis, "Anomaly detection of adversarial examples using class-conditional generative adversarial networks," *Computers & Security*, vol. 124, p. 102956, 2023.

[35] S. Rahman, S. Pal, A. Fallah, R. Doss, and C. Karmakar, "Rad-iomt: Robust adversarial defense mechanisms for iomt medical image analysis," *Ad Hoc Networks*, p. 103935, 2025.

[36] H. Liu, B. Zhao, J. Guo, K. Zhang, and P. Liu, "A lightweight unsupervised adversarial detector based on autoencoder and isolation forest," *Pattern Recognition*, vol. 147, p. 110127, 2024.

[37] G. Braun, S. Cohen, and L. Rokach, "Adversarial evasion attacks detection for tree-based ensembles: A representation learning approach," *Information Fusion*, vol. 118, p. 102964, 2025.

[38] Y. Wu, S. S. Arora, Y. Wu, and H. Yang, "Beating attackers at their own games: Adversarial example detection using adversarial gradient directions," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 4, 2021, pp. 2969–2977.

[39] W. Liu, W. Zhang, K. Yang, Y. Chen, K. Guo, and J. Wei, "Enhancing generalization in few-shot learning for detecting unknown adversarial examples," *Neural Processing Letters*, vol. 56, no. 2, p. 85, 2024.

[40] H. Zheng, Z. Zhang, J. Gu, H. Lee, and A. Prakash, "Efficient adversarial training with transferable adversarial examples," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[41] H. Zhang, H. Chen, Z. Song, D. Boning, I. S. Dhillon, and C.-J. Hsieh, "The limitations of adversarial training and the blind-spot attack," *arXiv preprint arXiv:1901.04684*, 2019.