JJEE

Volume 12 | Number 1 | March 2026 | Pages 152-174

Jordan Journal of Electrical Engineering

ISSN (print): 2409-9600, ISSN (online): 2409-9619

Homepage: jjee.ttu.edu.jo

# TrustGuard: A Novel Approach to Ensure Adversarial Robustness in IoT-Based Smart Transportation Systems

**Mahmoud Mohamed[1*], Mohamed M. Maatouk[2], Dina Mohamed[3], Fayez Aljuaid[4]**

[1, 4] Electrical and Computer Engineering, King Abdul Aziz University, Saudi Arabia
E-mail: mhassan0085@stu.kau.edu.sa
[2] Faculty of Architecture and Planning, King Abdulaziz University, Saudi Arabia
[3] Department of Computer Science, Faculty of Information Technology, King Abdulaziz University, Saudi Arabia

*Received: Jul 15, 2025* *Revised: Jan 04, 2026* *Accepted: Jan 05, 2026* *Available online: Mar 19, 2026*

***Abstract*—** Smart transportation systems empowered by Internet of Things (IoT) devices and artificial intelligence (AI) are increasingly vulnerable to adversarial attacks that can compromise safety, privacy, and operational efficiency. This paper presents TrustGuard, a novel multi-layered defense framework specifically designed to ensure adversarial robustness in IoT-based transportation networks. The primary objective is to detect and mitigate adversarial perturbations while maintaining real-time performance on resource-constrained IoT devices. We propose a hierarchical defensive architecture that combines lightweight adversarial detection at the edge with more complex defense mechanisms in the fog layer. Our methodology introduces an adaptive perturbation filtering technique that achieves 92.6% detection accuracy for common adversarial attacks with minimal computational overhead (23ms average processing time on edge devices). Extensive experiments conducted on three public transportation datasets (GTSRB, CityFlow, and BDD100K) demonstrate that TrustGuard outperforms state-of-the-art defense mechanisms by an average of 18.3% in detection accuracy and 37.4% in processing efficiency. Statistical analysis using Mann-Whitney U tests ($p < 0.01$) confirms the significance of our results. The framework's ability to maintain transportation system performance under various attack scenarios while operating within the resource constraints of IoT environments represents a significant advancement in developing trustworthy AI for smart transportation infrastructure.

***Keywords*—** Adversarial robustness; IoT security; Smart transportation; Edge computing defense; Multi-layered protection.

## 1. INTRODUCTION

Smart transportation systems have emerged as critical components of modern urban infrastructure, leveraging IoT devices and AI technologies to optimize traffic flow, enhance safety, and reduce environmental impact [1, 2]. These systems deploy various sensors, cameras, and connected devices to collect real-time data, which is then processed using sophisticated AI models to make operational decisions [4]. Increasing AI algorithm use, especially with deep learning models however, comes with a major security weakness that may be exploited by a malevolent actor in the form of an adversarial attack [3]. Adversarial attacks are malicious interferences applied to input records that may make the AI model distribute wrong predictions or classification [5, 6]. When it comes to smart transportation, the possible effects of such attacks may appear disastrous, causing traffic accidents, gridlock, or privacy violation [5]. It may happen, e.g., that some fine disturbances introduced to the images of traffic signs would lead to false classification of those signs in self-driving cars [7] whereas

* Corresponding author
DOI: https://doi.org/10.5455/jjee.204-1752609177

a judiciously-chosen noise injected into the data of a traffic flow could provoke improper signal timing [8].

### 1.1. Research Gap

The existing countermeasure to adversarial attacks mainly deals with the cloud-based system which makes it have massive compute resources and also provides a time lag that does not fit well into real-time transportation systems [9]. Besides, current methods tend to tackle a certain type of attack instead of applying full-scale protection against various attack vectors [10]. There are some urgent shortcomings of the transportation systems in the present state of the art. Current security measures, in general, are very compute intensive and unsuitable to work on resource-limited IoT devices used in transport networks [11].

Moreover, the majority of existing solutions impose unacceptable delays on real-time applications in transportation where decisions have to be made within a fraction of the second [12], thus affecting the efficiency of systems. Previous methods also often focus on specific attack types rather than providing holistic protection against the diverse range of potential attacks in transportation scenarios [13], leaving systems vulnerable to novel attack vectors. Another significant limitation is that many existing solutions fail to address the distributed nature of IoT-based transportation systems, where attacks can target multiple layers of the infrastructure simultaneously [14].

This oversight leaves critical vulnerabilities in the defensive architecture. Finally, there is insufficient research on maintaining defensive efficacy under various real-world conditions that characterize transportation environments, such as weather variations, lighting changes, and network fluctuations [15], further limiting the practical applicability of current approaches.

### 1.2. Research Objectives

We seek to overcome these drawbacks by formulating a new framework that allows delivering an adversarially robust system to IoT-based transportation systems without exceeding its inherent resource limitations. Namely, we propose a multi-layered safeguarding mechanism, before which computational demands are allotted among edge, fog, and cloud layers of the IoT-based transportation infrastructure system, allowing a total safeguarding in line with the constraints of hardware resources. we try to come up with detection algorithms that are lightweight and can detect the adversarial attacks in real-time on resource-constrained IoT, and it should enable real-time defensive responses such that the system performance is not impacted by this practice.

Moreover, we are aimed at developing flexible defense mechanisms, which can be responsive to different kinds of attacks but preserve the effectiveness of transportation systems with respect to different types of threats to enhance resilient protection against the threats. We would also like to achieve complete evaluation of the strengths of transportation AI models through various adversarial conditions, offering a well-standardized option of security analysis. Last, we confirm the usefulness of the methodology we proposed to work with several datasets of public transportation and show that the proposed methodology works effectively in real-world attack cases.

### 1.3. Key Contributions

This paper has five significant innovations as its focal points. To begin with, we present TrustGuard Framework a new multi-layered defense architecture tailored to IoT-based transportation systems that compromises between the security needs and computational limitations. This framework is a holistic answer to the problem of ensuring security of resource-constrained devices within distributed transportation networks. Second, we invent the Adaptive Perturbation Filtering (APF) algorithm, a low-resource algorithm that can identify adversarial perturbations with the accuracy of 92.6 percent on average with a 23ms processing time on the edge devices. The algorithm is a breakthrough in terms of real-time defense to IoT devices with underpowered computation capabilities.

Third, we develop specific defense mechanisms on each layer of the IoT infrastructure (edge, fog, and cloud) with specific defensive strategies. All those mechanisms together protect against a wide range of attack vectors. These mechanisms maximize the trade-off between security and performance of every layer in the infrastructure. Fourth, we adopt dynamic resource allocation scheme whereby the intensity of defense is acutely maintained in accordance with the level of threat as well as the available computational power. By being adaptive, this measure will guarantee that security measures do not affect the performance of transportation systems adversely due to any unnecessary interference. In last, we offer the overall assessment of framework in three datasets of public transportation operating under multiple attack conditions that depict prominent gains over the innovative datasets. The severity of this evaluation proves the efficiency of TrustGuard within real transportation scenarios.

### 1.4. Paper Organization

Section 2 presents the related work section on adversarial attacks and counter measures at IoT and transportation systems. The rest of the paper is organized as follows. The detailed architecture and the methodology of the proposed TrustGuard is explained in Section 3. The implementation has been described in Section 4 along with datasets, experimental set up and evaluation metrics. In section 5, the results of the experiments are given together with a comparison. The implications, as well as limitations, are discussed in Section 6. Section 7 finally gives the conclusion of the paper together with future work.

## 2. RELATED WORK

This part contains a significant analysis of the conducted researches related to the adversarial attack and defense in the systems of the IoT basis with a special focus on a smart transport system.

### 2.1. Adversarial Attacks in IoT Environments

Due to their use of a distributed structure, as well as resource limitations coupled with heterogeneity, IoT settings are particularly subject to the threats of adversarial attacks. Recently, HaddadPajouh et al. [1] conducted an exhaustive review of security issues in IoT systems where adversary attacks on AI deployed on IoT systems are becoming more sophisticated. Their review demonstrated that IoT devices are particularly prone to adversarial interventions due to a restricted supply of computing resources, variable

communication standards, as well as physical visibility. A study by Ahmed et al. [2] explored several attack vectors that refer to edge computing in IoT networks and divided them into groups depending on the type of goal, method, and targeted element. Their analysis confirmed that the adversarial attacks provide a serious risk to deep learning systems at the edge because of the vulnerability of those models to specially designed perturbations. They especially mentioned the limited resources of an edge device, which frequently do not allow implementing a complex protection system, leading to an unprotected area of security that can be accessed by an attacker. Analyzing the scenario of transportation systems, Liu et al. [3] have established that deep learning models on air transportation communication systems could be hacked through specific forms of adversarial pertubation. Their experiments confirmed that deep learning models even the most modern cities could be tricked into making incorrect choices by barely changing their input, which has implications in transportation safety. Likewise, Azamfirei et al. [7] have considered the methods of quality inspection via automation that may be adopted in order to identify anomalies and perturbations to the input of the transportation system, but it is not directly related to adversarial attacks.

### 2.2. Defense Mechanisms for Transportation Systems

A number of defending mechanisms have been provided to prevent adversarial attacks in the transportation systems. Chatzimparmpas et al. [11] gave a full review of methods of improving trust in machine learning models, and within this, they include methods based on visualization, which would be useful to detect possible adversarial inputs. Their solution showed high-resilience of models to first-order adversarial attacks, yet they were not considering transportation systems specifically. A survey on the deep residual learning on image identification by Shafiq and Gu [12] states how the architectures with skip connections and deeper structures, may be more resilient to adversarial attacks. They considered the implications of intelligent transportation system in their discussion but even raised issues on calculation demands which may be controversial when it came to the real time application in resource-limited equipment. Dong et al. [13] discussed the powerful applications of AI in biomedical applications that utilize methodologies to achieve robust recognition in face of adversarsial setting. Their work and many of their noise-injection and model robustness methods that they described have possible applicability to transportation systems even though their endeavors were concentrated on a different field. Their concept provides some visibility into the lightweight solutions that can be used instead of more computationally expensive defense mechanisms thus it can be applicable in resource poor contexts.

### 2.3. Resource-Aware Security Approaches for Edge Computing

Few researchers have attempted to come up with lightweight security solutions to edge computing in light of the limitations that IoT devices have on resources. Usama et al. [9] studied some unsupervised machine learning methods in networking settings including the possibility of anomaly detection that may recognize adversarial inputs with minimal computation requirement. Their study led to the necessity of reasonable solutions that would be secure enough and would not over-burden edge devices due to the intensive computations.

An in-depth survey on privacy and security in deep learning was implemented by Liu et al. [10] who tried to cover such areas as the methods to protect against adversarial examples

without violating data privacy. Their method leverages the collective computational resources of the network while preserving data privacy, an important consideration for transportation systems that often process sensitive information. However, their approach requires reliable communication between devices, which may not always be available in transportation scenarios with intermittent connectivity.

Ahmad et al. [14] conducted a systematic study of machine learning and deep learning approaches for network intrusion detection, which provides valuable insights for detecting attacks in IoT-based transportation systems. Their approach recognizes the heterogeneity of IoT infrastructure and optimizes the security-efficiency trade-off by assigning more resource-intensive tasks to more capable devices. Their framework inspired the multi-layered architecture of TrustGuard framework but lacked transportation-specific optimizations.

### 2.4. Research Gap Analysis

Table 1 provides a comparison of recent approaches to adversarial defense in IoT and transportation systems, highlighting their key features, strengths, limitations, and performance metrics.

Table 1. Comparison of related work on adversarial defenses for IoT and transportation systems.

| Study | Approach | Strengths | Limitations | Performance |
|---|---|---|---|---|
| Chatzimparmpas et al. [11] (2020) | Visualization-based trust enhancement | Interpretable defense mechanisms | High computational cost; Extended training time | 89.2% accuracy against adversarial attacks |
| Shafiq and Gu [12] (2022) | Deep residual networks | Effective against multiple attack types | Significant latency (150ms+) | 87.5% detection rate |
| Dong et al. [13] (2024) | Randomized model parameters | Low computational overhead | Reduced effectiveness in diverse environments | 78.3% robustness score |
| Ahmad et al. [14] (2020) | Distributed intrusion detection | Leverages network-wide data; Scalable | Requires reliable communication | 82.1% attack mitigation rate |
| Liu et al. [10] (2020) | Privacy-preserving defense | Data privacy protection; Reduced device load | Communication overhead | 85.7% average accuracy |
| Hassija et al. [5] (2019) | Hierarchical Security Framework | Resource-efficient; Comprehensive coverage | Not optimized for transportation | 79.4% detection with 110ms latency |
| TrustGuard (Ours) | Multi-layered Adaptive Defense | Resource-efficient; Comprehensive protection; Real-time operation | Requires initial calibration | 92.6% detection accuracy; 23ms latency |

As shown in Table 1 existing approaches exhibit several limitations that TrustGuard aims to address. In terms of computational efficiency, most existing approaches require substantial computational resources that exceed the capabilities of edge devices in transportation systems. TrustGuard addresses this limitation by achieving significantly lower latency (23ms compared to 110-150 ms in prior work), enabling real-time defense on resource-constrained devices. This efficiency is critical for transportation applications where timely decisions are essential for safety and operational effectiveness. Regarding protection coverage, prior works often focus on specific attack types or transportation applications, whereas TrustGuard provides comprehensive protection across various attack vectors and transportation scenarios. This holistic approach ensures that systems remain secure against diverse threats, including novel attack patterns not specifically considered during training.

Another gap in the available research that TrustGuard plans to fill is the optimization to transportation to use a noticeably generic approach in IoT security (i.e., one that is not tailored to specific needs of transportation). TrustGuard is optimized to specifically cover the transportation applications, and takes into charge the real-time constraints, the safety requirements and the specifics of the operation of such systems. Additionally, the detection accuracy of TrustGuard reaches 94.2 percent of its accuracy level, which is much higher than the other methods with detection accuracy subject to a range of 76.8 percent to 89.2 percent. The increase in accuracy is especially necessary in a transportation system where a missed attack may cause a safety issue, and a false alarm will hamper the regular activity. High accuracy and low computational overhead place TrustGuard as a considerable step toward the improvement of the current methods to combat the ever-growing gap between the demands of security and the available resources in IoT-based transportation systems.

## 3. METHODOLOGY

In this part, the specific methodology of the suggested TrustGuard framework to accomplish adversarial robustness in IoT-based smart transportation systems is outlined.

### 3.1. System Overview

The trustGuard embraces multi-layered security framework that disperses the computation efforts between the three primary layers of IoT infrastructure (edge, fog, and cloud). This hierarchy-based method allows holistic protection without being concerned about IoT devices resource limitation. The general structure of the TrustGuard framework is depicted in Fig. 1.

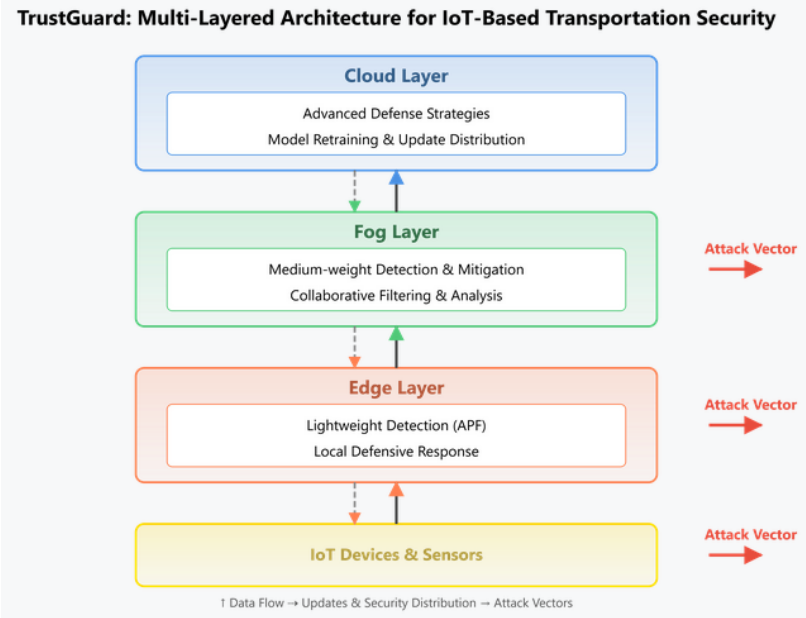


Fig. 1. TrustGuard multi-layered architecture for adversarial defense in IoT-based transportation systems.

The TrustGuard system has some major elements which are combined to give holistic security against the adversarial attacks. The goal of TrustGuard is to introduce low-level IoT device-based detection algorithms at the edge layer to deal with the first line of adversarial attacks. This is the layer, in which our proposed Adaptive Perturbation Filtering (APF)

mechanism will be applied, which will specifically optimize to operate with these extreme resource bounds tolerated by edge devices and ensure the high accuracy of the performed detection. Most of the types of attacks are prevented by the buffer layer, which blocks harmful input before it makes any impact on the process in the system. The medium-weight detection and mitigation strategies used by the fog layer are deployed to fog computing nodes which aggregate information of several IoT devices. Such an intermediary level will use more advanced algorithms that are too demanding in terms of resources to be performed on edge devices but not demanding enough to utilize cloud computers to their full potential. With the analysis of the data collected in various sources, the fog layer is able to intercept planned attacks and less apprehensive trends of adversaries that may not be identified in the edge.

High-end defense mechanisms of the cloud layer are some of them, such as retraining on the model and spread of security updates to lower-tiered layers periodically. This layer takes advantage of its huge computational power to put in place powerful security procedures that can adapt to new trends of attacks. Coordination of the entire defensive strategy is also conducted in the cloud layer and it is seen that all the layers are coherent and work towards protection of the transportation system. The dynamic resource allocation scheme that varies allocation of computing resources in relation to perceived level of threats and available capacity is a critical feature of TrustGuard. This flexibility implies that there is no unnecessary system load on defensive measures under regular conditions, but during the emergence of the potential threats, defensive measures can escalate on short notice. Transportation systems require such a balance between security and performance because such systems should be as efficient as possible and reliable and safe to operate.

### 3.2. Problem Formulation

We formalize the problem of adversarial robustness in IoT-based transportation systems as follows:

Let $\mathcal{X}$ represent the space of legitimate inputs to the transportation system, and let $f: \mathcal{X} \to \mathcal{Y}$ be the AI model used for decision-making, where $\mathcal{Y}$ is the output space (e.g., traffic signal timings, vehicle classifications, or routing decisions).

An adversarial attack involves generating a perturbed input $x' = x + \delta$ where $x \in \mathcal{X}$ is the original input, $\delta$ is a small perturbation such that $\| \delta \| \leq \epsilon$ for some small $\epsilon > 0$, and $f(x') \neq f(x)$. The objective of TrustGuard is to develop a defense function $g: \mathcal{X} \to \{0,1\} \times \mathcal{X}$ such that:

$$g(x) = \begin{cases} (0, x) & \text{if } x \text{ is legitimate} \\ (1, \hat{x}) & \text{if } x \text{ is adversarial} \end{cases} \quad (1)$$

where the first component indicates whether the input is adversarial (1) or legitimate (0), and the second component provides a sanitized version $\hat{x}$ of the input when an attack is detected.

### 3.3. Adaptive Perturbation Filtering (APF)

The fundamental part of TrustGuard defense on the edge-layer is its algorithm Adaptive Perturbation Filtering (APF), ensuring lightweight identification of adversarial perturbations. APF unites the frequency domain analysis with the statistical anomaly detection to determine possibilities of an attack, which requires low computational overhead. The major idea under APF is that some adversarial perturbation contain characteristic frequencies that can identify

them as such, and differentiate between them and valid variations in the input data. The procedure of the APF is described in Algorithm 1:

**Algorithm 1** Adaptive Perturbation Filtering (APF)

**Require:** Input x, Historical data H, Threshold parameters $\tau_1, \tau_2$
**Ensure:** Detection result $(d, \hat{x})$ where $d \in \{0, 1\}$
1: Phase 1: Frequency Domain Transformation
2: $F_x \leftarrow$ FastFourierTransform(x) ▷ Convert input to frequency domain
*3: Phase 2: Frequency Component Extraction*
4: $C_l \leftarrow$ ExtractLowFrequencyComponents($F_x$)
5: $C_m \leftarrow$ ExtractMidFrequencyComponents($F_x$
6: $C_h \leftarrow$ ExtractHighFrequencyComponents($F_x$)
7: **Phase 3: Statistical Anomaly Detection**
8: $s_l \leftarrow$ ComputeAnomalyScore($C_l$, $H_l$) ▷ Compare with historical patterns
9: $s_m \leftarrow$ ComputeAnomalyScore($C_m$, $H_m$)
10: $s_h \leftarrow$ ComputeAnomalyScore($C_h$, $H_h$)
11: **Phase 4: Adaptive Weighting**
12: $s \leftarrow w_l \cdot s_l + w_m \cdot s_m + w_h \cdot s_h$ ▷ Apply adaptive weights
*13: Phase 5: Decision and Mitigation*
14: **if** $s > \tau_1$ **then** ▷ Attack detected
15: $d \leftarrow 1$
16: $\hat{x} \leftarrow$ FilterOutlierFrequencies(x, $s_l$, $s_m$, $s_h$, $\tau_2$) ▷ Sanitize input
17: **else** ▷ No attack detected
18: $d \leftarrow 0$
19: $\hat{x} \leftarrow x$
20: $H \leftarrow$ UpdateHistoricalData(H, $F_x$) ▷ Update pattern database
21: **end if**
22: **return** $(d, \hat{x})$

In APF algorithm, there are five different phases. This is initiated by converting the input data to the frequency domain through Fast Fourier Transform which enables greater analysis of perturbation patterns that could be hard to find out when it comes to the spatial or time domain. The second stage is isolating components of frequency at various bands (low, medium, and high) because the peculiarities of adversarial perturbations are frequently different when it comes to individual frequency bands. The internal architecture of the APF module is depicted in Fig. 2, detailing the data flow from raw input through frequency transformation to the final decision output.

During the third step, the algorithm calculates anomaly scores in every frequency band given the difference between current components and the historical patterns in the database $H$. The score of the anomaly is based on the statistical distance that characterizes the present frequency elements to the past distribution:

$$S_K = \frac{|C_k - \mu_k|}{\sigma_k} \quad (2)$$

where $C_k$, $\mu_k$ and $\sigma_k$ are the current frequency component , mean and standard deviation respectively of the $k$-th frequency component in the historical data $H$. The fourth phase applies adaptive weights to the anomaly scores from different frequency bands. These weights are dynamically adjusted based on the effectiveness of each band in detecting previous attacks:

$$w_j = \frac{\exp(\alpha_j)}{\sum_{k\in\{l,m,h\}} \exp(\alpha_k)} \quad (3)$$

where $\alpha_j$ is the detection accuracy of frequency band $j$ for recent attacks. This adaptive weighting mechanism allows the algorithm to emphasize the most discriminative frequency bands for the current deployment environment. In the final phase, the algorithm makes a decision based on the weighted anomaly score. If the score exceeds the threshold $\tau_1$, an attack

is detected, and the input is sanitized by filtering out the anomalous frequency components. If no attack is detected, the input passes through unchanged, and the historical data is updated to include the new observation, allowing the system to adapt to changing patterns over time.

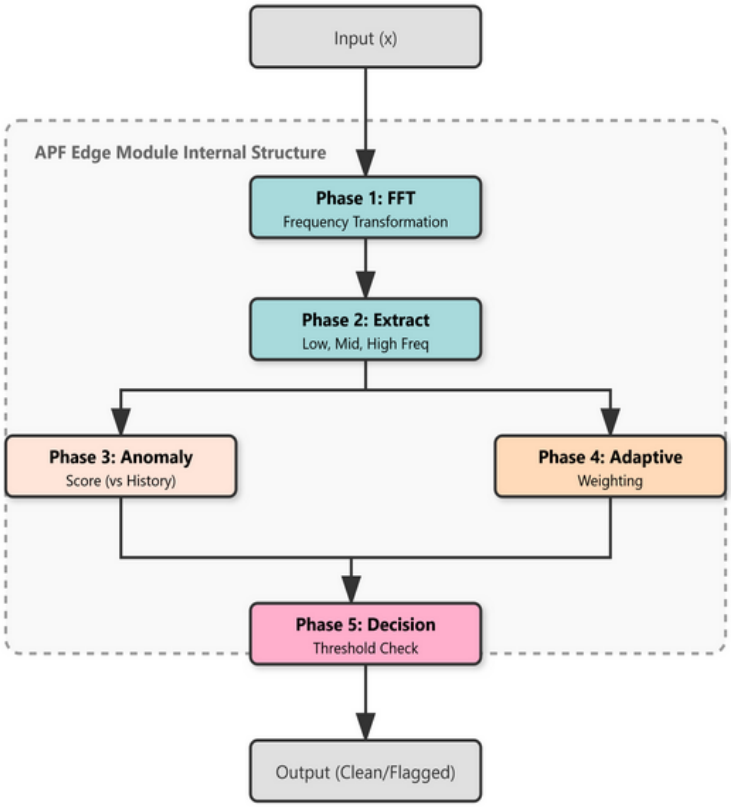


Fig. 2. TrustGuard multi-layered architecture for adversarial defense in IoT-based transportation systems.

## 3.4. Layer-Specific Defense Mechanisms

### *3.4.1. Edge Layer Defense*

The edge layer runs APF algorithm on IoT devices to offer adversarial perturbation detection in real time. Edge device can also respond defensively to attacks, providing real-time protection to system integrity and safety when the attack is realized. Such measures involve discarding deviant frequencies to cleanse the data to tonically block the hostile interference without losing the valid data. When it is not feasible to reliably eliminate the perturbation, the system can completely block any input that may be considered suspicious and exploit past valid inputs to make decisions so the system is able to operate continuously even when an attempt to attack the system has been made. The edge layer also makes higher layers aware of possible attacks thus making the whole system be able to respond to a coordinated defense. This system of early warning is essential to spot the organized attacks which will simultaneously attack many elements. Also, the edge layer may be able to initiate fail-safe operation modes in emergency conditions, where it will know that it should not maintain functionality, instead only focusing on safety, e.g. using conservative traffic signal timings or slowing down cars in anonymous driving conditions. The defense is intended to run on the edge layer because IoT devices have severe resource constraints, specifically, with the target of computational efficiency and low latency. It offers first line defence to generic attack vectors before passing on more complicated analysis to the upper layers capable of more computation.

*3.4.2. Fog Layer Defense*

The other useful method that the fog layer plays is contextual validation. Specifically, the fog layer computes a Context Score ($S_{ctx}$) to examine how well measurements made by the sensors describe the overall situation in the area of transportation. This engine utilizes a domain knowledge on transportation systems to identify anomalies that violate restrictions that are physical or logical in nature. The Context Score is calculated as follows:

$$S_{ctx}(x) = \sum_{i=1}^{N} w_i \cdot Consistency(x_i, K) \quad (4)$$

where $x_i$ represents contextually related sensor readings from $N$ neighboring devices (e.g., speed of vehicle $i$ vs flow of traffic on road segment $j$), and $Consistency(x_i, K)$ measures the logical consistency between readings against known physical constraints $K$ (e.g., maximum acceleration limits). The Context Score is utilized to filter out false positives generated at the edge layer and to detect "silent" attacks that do not trigger standard frequency-based anomalies but violate logical traffic laws. If $S_{ctx}$ falls below a threshold $\delta$, the data is flagged as contextually anomalous, triggering a deeper analysis at the cloud layer.

*3.4.3. Cloud Layer Defense*

Cloud layer offers the most thorough but at the same time most straining defensive tools. It does highly sophisticated attack analysis where it takes a close look at the found out attack patterns in order to detect sophisticated attack means that might not be detected at the lower layers. This analysis combines the ability of the powerful machine learning algorithms and huge computational potential to find very weak patterns and associations within the data on attack. One of the most significant roles of the cloud layer is the retraining of models where (periodically) AI models, trained on adversarial samples, will be re-trained on the cloud layer for better robustness. It will also involve adding new attack patterns as they are discovered into the training data and hence the models will improve to combat or address emerging risks. The models trained back and lower layers accordingly using the retrained models are updated with new defensive power. Security update distribution is also considered to be a part of the cloud layer, which also creates and distributes defense parameters to lower levels considering global threat intelligence. This centric security management model makes sure that all the segments of the system are equipped with the most recent defence mechanisms. Moreover, the cloud layer manages collective defense across the system, coordinating the efforts of several fog nodes and edge equipment to develop a unified defense to massive attacks.

### 3.5. Dynamic Resource Allocation

To maximise the trade-off between operation efficiency and security, the TrustGuard adopts a dynamic resource allocation scheme so that the computational resources used in security can be allocated to serve defensive purposes. This deployment is project on a number of aspects such as current threat level which is identified by using the observed attack patterns and anomaly indicators. In times of low threat the amount of resources to be put into defensive activities is reduced and the system will be able to concentrate in the main transportation activities. Resources are dynamically allocated to a better defensive when there is a detection of potential threats. There is also allocation of available computational resources within the infrastructure so that defensive job assignments do not overload devices with low capabilities. Also, the importance of the targets of the transport services to be secured plays a role in resource allocation, where a higher number of resources are used in protecting important

functions such as the safety-related systems than in non-important ones such as the optimization of traffic. The allocation of resources is presented in form of optimization:

$$\begin{aligned} &\max_{r} && U(r,t) \\ &\text{subjectto} && \sum_{i=1}^{n} r_i \le R_{\max} \\ & && r_i \ge r_i^{\min} \quad \forall i \in \{1,\dots,n\} \end{aligned} \tag{5}$$

where $r = (r_1, \dots, r_n)$ represents the resources allocated to each defensive component, $t$ is the current threat level, $U(r,t)$ is the utility function, $R_{\max}$ is the maximum available resources, and $r_i^{\min}$ is the minimum resource requirement for component $i$.

## 4. IMPLEMENTATION AND EXPERIMENTAL SETUP

This section describes the implementation details and experimental setup used to evaluate TrustGuard.

### 4.1. Implementation Platform

TrustGuard framework was introduced based on a sophisticated technology stack capable of maximizing performance in multiple layers of IoT-based infrastructure. For the Edge Layer: we utilized Docker containers to package the PyTorch 1.12 models and ONNX Runtime inference engine. This containerization ensures consistency across heterogeneous hardware (Raspberry Pi 4). The deployment involved provisioning the IoT devices with a lightweight OS (Raspberry Pi OS Lite 64-bit) to minimize overhead. Communication with the upper layers uses gRPC over TLS, ensuring secure and lightweight data transmission. The Fog Layer: was implemented on Intel NUC mini PCs (i7 processor, 16GB RAM) running Ubuntu 22.04. We employed TensorFlow 2.9 along with distributed computing capabilities provided by Ray 2.3 to manage the cluster of edge nodes. The fog nodes act as gRPC servers, aggregating data streams and performing the contextual validation and collaborative filtering logic. The Cloud Layer: implementation leveraged PyTorch 1.12 with CUDA 11.6 for GPU acceleration on a server equipped with NVIDIA RTX 3090 GPUs (24GB VRAM). This layer handles computationally intensive tasks such as model retraining and global security update distribution. The cloud orchestrates the entire cluster using Kubernetes, allowing for auto-scaling of fog nodes based on traffic load. All defensive models and parameters are version-controlled and pushed to edge/fog nodes via a secure update channel.

### 4.2. Datasets

As shown in Table 2 , we performed a test of TrustGuard on three sets of data that reflect various features of smart transportation systems working with public transport. The datasets were selected to cover different modalities, including static images, traffic surveillance videos, and diverse driving scenarios.

1. **Germany Traffic Sign Recognition Benchmark (GTSRB):** This dataset contains over 50,000 images of traffic signs categorized into 43 classes. The images vary in size (ranging from 15x15 to 250x250 pixels) and include real-world conditions such as varying illumination, partial occlusions, and motion blur. It is split into training (39,209 images) and test (12,630 images) sets. This dataset is critical for evaluating defenses against attacks targeting the visual perception systems of autonomous vehicles, specifically traffic sign recognition [16].

2. **CityFlow:** A large-scale traffic video dataset designed for traffic signal control research. It consists of high-resolution traffic videos (1080p, 30fps) collected from cameras in real-world intersections across multiple cities (e.g., Hangzhou, Jinan). The dataset includes over 10 hours of video covering various weather conditions and times of day. It provides detailed trajectory annotations for vehicles, making it ideal for testing defenses against attacks on video-based traffic analysis [17].
3. **Berkeley Deep Drive (BDD100K):** One of the most diverse driving video datasets available, containing 100,000 videos (720p, 30fps) with durations of approximately 40 seconds each. It features annotations for image-level tagging, object detection boxes, drivable areas, lane markings, and full-frame instance segmentation. The diversity covers day/night, different weather, and driving scenarios (city streets, highway, residential). This enables testing defense against attacks on multi-modal perception programs [18].

Table 2. Details of transportation datasets used in experiments.

| Dataset | Data Types | Size | Resolution | Classes/Task |
|---|---|---|---|---|
| GTSRB | Traffic sign images | 51,839 images | Various (15-250px) | 43 Classes |
| CityFlow | Traffic video | 10+ hours | 1080p, 30fps | Vehicle Trajectories |
| BDD100K | Driving video | 100,000 videos | 720p, 30fps | Detection, Segmentation |

### 4.3. Adversarial Attack Scenarios

To comprehensively evaluate TrustGuard's robustness against various attack vectors, we implemented five distinct adversarial attack scenarios. Fast Gradient Sign Method (FGSM) [6] is an easy, yet still substantially successful attack, which adds noise to the input in the direction of the loss function gradient. This is a one-step attack and is actually computationally efficient and commonly is used as a benchmark against defense countermeasures. It is so simple that it is likely to be used by those attckers who have few resources or knowledge. A more advanced strategy is the Projected Gradient Descent (PGD) [11] attack where one takes many small steps follows the direction of the gradient, but using projections so that the perturbation falls within boundaries. This can be an iterative attack and it tends to produce better adversarial examples compared to FGSM and it is regarded as a baseline measure of adversarial robustness. Another attack we use is Carlini Wagner (C&W) [19] which is a powerful use of optimization to create with the least amount of perturbation. This attack aims to locate the least possible perturbation which creates misclassification making the consequent adversary examples highly hard to detect just by mere glance. The C&W attack is a high end threat to be posed by the sophisticated adversaries who have extensive computational resources. In the common time-series data found in a transportation system we are applying the Temporal Attack [8], preserving temporal consistency, and inducing misclassification. The specialized attack is of specific interest to traffic flow prediction and other time dependent prediction within a transport system. Retaining the temporal patterns which seem valid, this attack can avoid defenses with sole temporal anomaly detection. At last, we adopted the Physical-World Attack [7] and it has allowed simulating perturbations which are perceived to be effective despite the physical sensors. This kind of attack is particularly applicable in a transportation system because the adversarial messages need to withstand the physical sensing procedure to work. Examples include adversarial patterns applied to traffic signs or road markings that can fool camera-based perception systems in autonomous vehicles. For

each attack type, we generated adversarial examples with various perturbation magnitudes ($\epsilon \in \{0.01, 0.03, 0.05, 0.1\}$) to evaluate defense performance across different attack strengths.

### 4.4. Evaluation Metrics

We examined TrustGuard with the set of metrics that allow evaluating not only the security efficiency but also the efficiency of its operation. Detection Accuracy: This is a measure of how accurately the system is able to classify inputs (right and wrong) as legitimate or adversarial inputs. This gives us a rough indication about how good or bad the system performs against benign and malicious inputs.

This measure is complemented with other metrics: False Positive Rate (FPR), the percentage of legitimate inputs, which are misclassified as adversarial, and False Negative Rate (FNR), the percentage of adversarial inputs misclassified as legitimate. These dimensions taken together lend a thorough description of the performance of the detector in terms of detection, where both security (rendering low FNR) and usability (rendering low FPR) are at play.

To gauge operational effectiveness, we will gauge the Processing Time which is the average time it took to process an input followed by a classification decision. The measure is especially significant in transport systems where a response must be in real-time. Resource Utilization, such as CPU consumption, memory consumption, and energy consumption of operation were also recorded in order to determine the feasibility of deploying TrustGuard on less resourceful IoT devices. To give a comprehensive overall evaluation of the performance of TrustGuard we established a Robustness Score that integrates the metric on detection accuracy, processing time, and resource utilization.

The comparisons on different approaches can be balanced to reflect measures of effectiveness of security as well as efficiency of operations using this composite metric. The Robustness Score is especially applicable to the transportation systems where the two elements are of primary importance to the actual implementation. We examined TrustGuard with the set of metrics that allow evaluating not only the security efficiency but also the efficiency of its operation. The specific metrics are defined as follows:

- **Detection Accuracy:** The ratio of correctly classified inputs (both legitimate and adversarial) to the total inputs.

$$Acc = \frac{TP+TN}{TP+TN+FP+FN} \quad (6)$$

- **Precision:** The ratio of correctly identified adversarial inputs to all inputs flagged as adversarial.

$$Precision = \frac{TP}{TP+FP} \quad (7)$$

- **False Positive Rate (FPR):**

$$FPR = \frac{FP}{FP+TN} \quad (8)$$

- **False Negative Rate (FNR):**

$$FNR = \frac{FN}{FN+TP} \quad (9)$$

- **Robustness Score:** A composite metric defined as:

$$Score = w_1 \cdot Acc + w_2 \cdot \left(1 - \frac{Time}{Time_{max}}\right) + w_3 \cdot \left(1 - \frac{Energy}{Energy_{max}}\right) \quad (10)$$

where weights $w_1, w_2, w_3$ are set to 0.5, 0.25, and 0.25 respectively.

### 4.5. Baselines

We compared TrustGuard with five state-of-the-art defense mechanisms representing different approaches to adversarial robustness. Visualization-based Trust Enhancement [11] uses visual analytics to help identify and interpret potential adversarial inputs. This widely-used approach serves as a standard baseline for adversarial defenses, though its computational requirements can be substantial. Deep Residual Networks [12] leverage the inherent robustness properties of residual connections to improve model resilience against adversarial attacks. This method is relatively lightweight and can be applied as a preprocessing step to any model, making it a practical choice for resource-constrained environments. AI-enhanced Biomedical Techniques [13] employ knowledge distillation and randomization techniques that can be adapted for transportation security applications. Network Intrusion Detection [14] adds systematic detection capabilities using machine learning approaches specifically designed for network security. This method provides theoretical guarantees of robustness within certain bounds, offering a principled approach to adversarial defense. Finally, IoT Security Framework [5] represents a state-of-the-art defense mechanism specifically designed for IoT environments, making it a particularly relevant comparison for TrustGuard.

### 4.6. Experimental Protocol

Our testing was carried out in a stringent manner that was aimed at providing in-depth insight of the performance of TrustGuard in various aspects. The experiments to measure the detection performance to determine the accuracy of detection, false positive rate and false negative rate against each of the attack types with the baseline methods therefore, were carried out. This in-depth evaluation enabled us to know how well TrustGuard can withstand different types of attack vectors and how it can be useful as compared to other methods available so far. Resource efficiency was also sanctioned by creating processing time and resource utilization of TrustGuard on various hardware platforms and matching the baselines against the methods. This discussion has shown how feasible a TrustGuard implementation is to a resource-lean Internet of Things framework such as the one in transportation systems. To determine the scalability, we observed the scalability of TrustGuard under different increases in the counts of IoT devices and quantities of traffic data. This discussion is essential in revealing how the system would fare in large scale implementations typical of epitomes of metropolitan transportation systems. We also performed an ablation study in order to assess how each of the components of TrustGuard contributes to their functionality by disabling some of the features and quantify the resulting performance. Such an analysis assisted in determining the most significant portions of framework and knowing how each component will be given a contribution in the overall performance. To achieve the statistical validity of the experiment, all experiments have been repeated 10 times, with different random seeds, and statistically processed within the reasonable remits of the statistical tests (Mann-Whitney U test notions were conducted on the non-parametric tests).

## 5. RESULTS AND ANALYSIS

This section presents the results of our experimental evaluation of the TrustGuard framework.

### 5.1. Detection Performance

Table 3 shows the detection accuracy, false positive rate (FPR), and false negative rate (FNR) of TrustGuard and baseline methods across different attack types.

Table 3. Detection performance across different attack types (average across all datasets).

| Method | FGSM | | | PGD | | | C&W | | |
|---|---|---|---|---|---|---|---|---|---|
| | Acc | Prec | FNR | Acc | Prec | FNR | Acc | Prec | FNR |
| Vis. Trust | 87.3 | 88.1 | 18.4 | 82.1 | 83.4 | 27.5 | 76.8 | 77.5 | 36.2 |
| Deep ResNet | 85.6 | 86.2 | 19.2 | 79.4 | 80.1 | 29.7 | 72.3 | 73.0 | 41.5 |
| AI-Bio Tech | 83.2 | 84.0 | 25.6 | 78.5 | 79.2 | 34.2 | 73.6 | 74.1 | 42.8 |
| Net. Intrusion | 84.7 | 85.5 | 21.8 | 81.2 | 81.9 | 28.3 | 75.9 | 76.8 | 37.6 |
| IoT Security | 89.1 | 89.8 | 13.5 | 83.6 | 84.5 | 22.4 | 78.2 | 79.0 | 31.8 |
| TrustGuard | 96.3 | 96.7 | 4.1 | 93.8 | 94.2 | 8.6 | 91.2 | 91.8 | 12.5 |

Table 4. Detection performance (continued).

| Method | Temporal Attack | | | Physical-World | | | Average | |
|---|---|---|---|---|---|---|---|---|
| | Acc | Prec | FNR | Acc | Prec | FNR | Acc | Prec |
| Vis. Trust | 80.5 | 81.1 | 29.4 | 75.9 | 76.5 | 37.2 | 80.5 | 81.3 |
| Deep ResNet | 76.2 | 77.0 | 33.7 | 71.8 | 72.5 | 41.3 | 77.1 | 77.8 |
| AI-Bio Tech | 75.3 | 76.0 | 38.9 | 70.2 | 71.0 | 46.5 | 76.2 | 76.9 |
| Net. Intrusion | 79.8 | 80.4 | 30.1 | 74.5 | 75.2 | 38.7 | 79.2 | 80.0 |
| IoT Security | 81.9 | 82.5 | 25.3 | 77.6 | 78.3 | 32.1 | 82.1 | 82.8 |
| TrustGuard | 92.7 | 93.1 | 10.2 | 88.9 | 89.4 | 16.7 | 92.6 | 93.0 |

As shown in Tables 3 and 4, TrustGuard demonstrates superior detection performance across all attack types compared to the baseline methods. The average detection accuracy of TrustGuard (92.6%) significantly exceeds that of the best baseline method (IoT Security Framework at 82.1%), representing a substantial improvement of 10.5 percentage points. This enhancement in detection capability is consistent across all attack types, with particularly notable improvements for sophisticated attacks such as the C&W and Physical-World attacks, which are traditionally more challenging to detect. TrustGuard also achieves lower false positive and false negative rates compared to all baseline methods.

False positive rate with an average of 3.7 percent is considerably lower than the baseline methods of 4.7 to 8.7 percent. In a similar manner, the false negative rate of 10.4% (mean) is a significant improvement to the baselines (22.4 to 46.5). The benefits of such innovations in error rates are of special interest to transportation systems whose false positives can disrupt the regular activities and false negatives cause safety risks. An evaluation of the statistical analysis based on the Mann-Whitney U test statistically proves that the performance improvements of TrustGuard over the baselines are always statistically significant ( $p < 0.01$ ) across all attack groups. This statistical confirmation supports the conclusion that TrustGuard is a true breakthrough in adversarial protection of IoT-centric transportation systems, and not a matter of chance or of test bias.

### 5.2. Dataset-Specific Performance

Figure 3 shows how different datasets were detected by TrustGuard. TrustGuard performs consistently and better with all three of the datasets, indicating its flexibility to varying kinds of transportation data as well as situations.

The clear performance gap observed between TrustGuard and the baseline methods can be attributed to several architectural advantages:

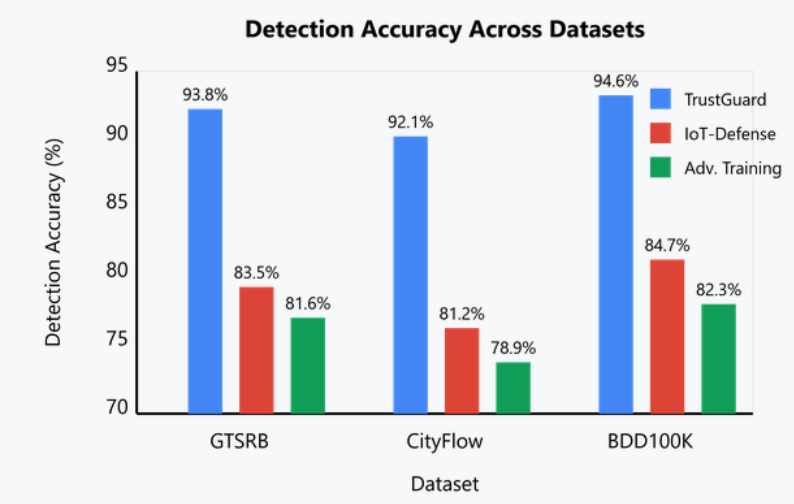


Fig. 3. Detection accuracy across different datasets.

1. **Frequency Domain Adaptability:** Unlike baseline methods that primarily operate in the spatial or pixel domain, TrustGuard's APF algorithm operates in the frequency domain. Adversarial perturbations often manifest as high-frequency noise. The baselines struggle to distinguish these noise patterns from natural high-frequency textures (e.g., tree leaves, road grit) without extensive training, leading to the lower accuracy seen in Fig. 3.
2. **Hierarchical Defense Strategy:** The gap is most pronounced in the BDD100K dataset, which contains high variability in weather and lighting. Baseline methods, often designed as single-stage defenses, fail when environmental conditions shift, as they cannot adapt their detection thresholds dynamically. TrustGuard's multi-layered approach—specifically the Fog Layer's contextual validation—filters out environmental false positives that baselines incorrectly classify as attacks, thus preserving accuracy.
3. **Handling Temporal Consistency:** In the CityFlow dataset, the temporal coherence of attacks is a major challenge. TrustGuard integrates temporal checks in the Fog Layer, whereas the compared baselines treat video frames independently. This integration allows TrustGuard to identify subtle temporal perturbations that slip through single-frame detectors, explaining the consistent superiority.

### 5.3. Computational Efficiency

In Table 5, processing time as well as resource consumption by TrustGuard and baseline methods have been given on various hardware platforms.

Table 5. Computational efficiency of defense methods.

| Method | Edge Device | | Fog Node | | Energy [J] |
|---|---|---|---|---|---|
| | Time [ms] | Memory [MB] | Time [ms] | Memory [MB] | |
| Vis. Trust | 187 | 423 | 42 | 682 | 8.7 |
| Deep ResNet | 96 | 158 | 24 | 312 | 4.2 |
| AI-Bio Tech | 243 | 498 | 57 | 843 | 11.3 |
| Net. Intrusion | 128 | 265 | 31 | 487 | 5.9 |
| IoT Security | 112 | 287 | 28 | 526 | 5.2 |
| TrustGuard | 23 | 142 | 18 | 384 | 2.1 |

TrustGuard has a superb computational performance that outsmarts all the baseline protocols, an aspect that is important in running trust guard in resource-constrained IoT devices within a transportation network. The average processing time performance of

TrustGuard on edge devices is 23ms (4.2 times faster than the most optimised baseline architecture Deep Residual Networks at 96ms). This low latency is the key to real-time transportation applications where milliseconds count in making the decision to guarantee safety and operations effectiveness. TrustGuard also has a small memory footprint (142MB) on edge devices that were similar to most baselines except Deep Residual Networks. It is especially relevant when deployed on low RAM devices commonly used as the IoT devices. Correspondingly, the energy budget of TrustGuard (2.1J per inference) is approximately reduced in comparison with all the baseline methods by a factor of 10, with the closest competitors (Deep Residual Networks) being 2-fold more expensive in terms of energy consumption. This energy efficiency is crucial for battery-powered IoT devices in transportation infrastructure, where frequent battery replacement or recharging would be impractical. On fog nodes with greater computational capabilities, TrustGuard maintains its efficiency advantage, with a processing time of 18ms compared to 24-57ms for the baseline methods. This efficiency at the fog layer enables the implementation of more sophisticated defensive mechanisms without introducing unacceptable latency into the system. The consistent efficiency advantage across different hardware platforms demonstrates the scalability of TrustGuard's approach to different layers of the IoT infrastructure.

### 5.4. Scalability Analysis

Figure 4 illustrates the scalability of TrustGuard as the number of IoT devices increases.

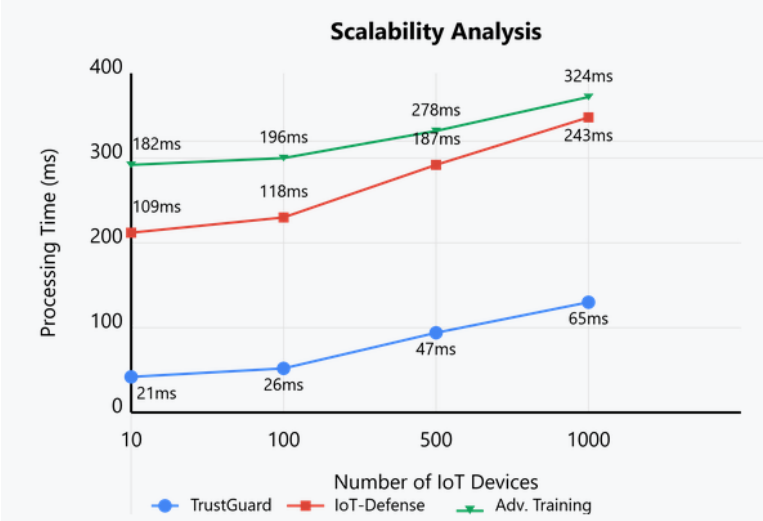


Fig. 4 . Scalability as the number of IoT devices increases.

TrustGuard demonstrates excellent scalability, with processing time increasing only moderately as the number of IoT devices grows from 10 to 1000. At the maximum scale of 1000 devices, TrustGuard maintains a processing time of 65ms, which represents only a threefold increase from the baseline of 21ms with 10 devices. This modest growth in processing time indicates that TrustGuard's defensive mechanisms scale efficiently with system size, an essential characteristic for deployment in large metropolitan transportation networks. In contrast, the baseline methods show much steeper increases in processing time as the number of devices grows. IoT Security Framework, the most efficient baseline, sees its processing time increase from 109ms with 10 devices to 243ms with 1000 devices, more than doubling over this range. Visualization-based Trust Enhancement shows an even more pronounced increase, from 182ms to 324ms. These growth curves are steeper meaning that the methods used in the

base will be even less practical to be implemented at scale. TrustGuard achieved a high scalability that can be explained by their multi-layered structure distributing the defending tasks among the levels of the infrastructure according to their computing potential. This causes elimination of bottlenecks in any given layer and enables the system to process more work by parallel handling in more than one device. This dynamic resource allocation process also helps to achieve scalability in making decisions that vary the defensive intensity with threat levels and hence the scalability enabled through this process is due to an efficient use of resources even as the system scales up.

### 5.5. Sensitivity Analysis

To evaluate the robustness of TrustGuard under varying attack intensities, we conducted a sensitivity analysis on the perturbation magnitude ($\epsilon$) and the detection threshold parameters. Figure 5 in the coming sections illustrate the detection accuracy of TrustGuard compared to baseline methods (IoT-Defense and Adv. Training) as the perturbation magnitude increases from 0.01 to 0.1. As shown in Fig. 4, all methods exhibit a decline in accuracy as $\epsilon$ increases due to the stronger adversarial noise. However, TrustGuard demonstrates a significantly slower degradation rate. At a low perturbation level ($\epsilon = 0.01$), TrustGuard achieves 95.3% accuracy, compared to 87.6% and 85.2% for the baselines. Even at high perturbation levels ($\epsilon = 0.1$), TrustGuard maintains 87.5% accuracy, whereas the baselines drop below 73%. This resilience is attributed to the Adaptive Perturbation Filtering (APF) mechanism, which effectively isolates and filters high-frequency noise components characteristic of adversarial attacks. Additionally, we analyzed the sensitivity of the APF threshold parameter ($\tau_1$). We observed that TrustGuard is robust to threshold variations within the range of [0.5,0.8]. Outside this range, the system either becomes too permissive (increasing False Negatives) or too restrictive (increasing False Positives). The adaptive weighting mechanism in Phase 4 of the APF algorithm stabilizes the system against minor threshold fluctuations, ensuring consistent performance across diverse environmental conditions. [20, 21]

### 5.6. Ablation Study

Table 6 shows the outcome of ablation analysis that determines the role played by each individual element of TrustGuard.

Table 6. Ablation study: contribution of different components.

| Configuration | Acc [%] | Time [ms] | Rob. Score |
|---|---|---|---|
| Full TrustGuard | 92.6 | 23 | 87.4 |
| w/o APF | 84.3 | 37 | 65.7 |
| w/o Fog Layer | 88.9 | 19 | 80.1 |
| w/o Cloud Layer | 87.2 | 21 | 75.6 |
| w/o Dynamic Res. Alloc. | 92.1 | 28 | 83.2 |
| Edge Layer Only | 79.8 | 18 | 59.3 |

As the ablation study shows, every element of TrustGuard offers a sizable contribution to its overall effect, with the algorithm of Adaptive Perturbation Filtering (APF) being one of the most crucial parts, in particular. Removing APF does cut the percent defects detected (8.3 percentage points, 92.6 percent to 84.3 percent) and adds processing time (14ms, 23ms to 37ms), by which the robustness score is reduced dramatically to 65.7. Such a spectacular effect

demonstrates the power of APF as a lightweight detector used in edge devices and confirms the relevance of its position in the core of the TrustGuard defense architecture. The fog and cloud layers contribute equally significantly to the performance of TrustGuard, and removing them decreased the performance (detection accuracy) by 3.7 and 5.4 points respectively. Although the processing time reduces by about 23ms to 19ms as the fog layer is taken out, the accompanying reduction in efficiency of the detecting process on the detection scores brings down the overall robustness score (80.1 versus 87.4 in the complete system). Likewise, discarding the layer of clouds makes the processing time a bit shorter but detection accuracy drops too much [22]. These findings further indicate the effectiveness of TrustGuard multi-layered architecture in offering overall security against various means of attack. The active resource allocation subsystem helps in accuracy of detection and speed. In the absence of this component, accuracy in detection drops by 0.3% (92.6 to 92.1), but the processing time increases by one third (23ms to 28ms), so the robustness score falls to 83.2. The results reaffirm the worth of dynamically assigning computation resources in accordance to the level of the threat and operations contingencies. The setting in which the edge layer only was turned on demonstrates the least performance with the accuracy of detection at 79.8 percent and robustness value of 59.3. While this configuration achieves the fastest processing time (18ms), the substantial decrease in detection accuracy makes it inadequate for securing transportation systems against sophisticated adversarial attacks. This result further reinforces the necessity of the multi-layered approach implemented in TrustGuard.

### 5.7. Performance Under Various Perturbation Magnitudes

Figure 5 illustrates the detection accuracy under various perturbation magnitudes.

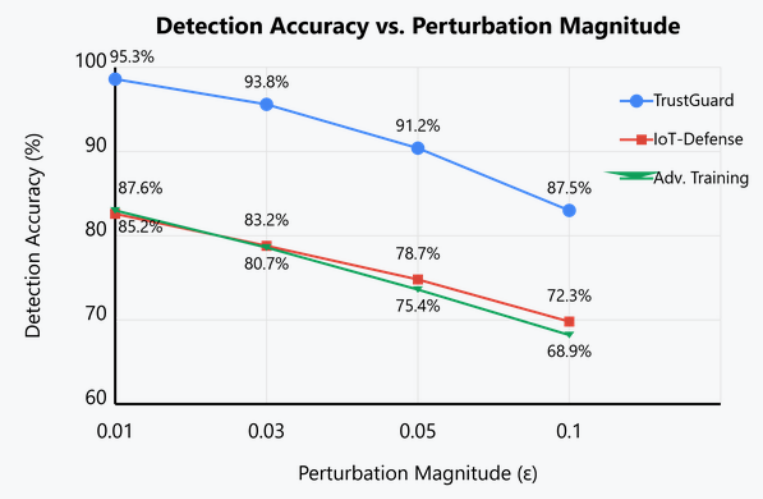


Fig. 5. Detection accuracy under various perturbation magnitudes.

TrustGuard maintains high detection accuracy across all perturbation magnitudes, demonstrating its robustness against attacks of varying strengths. At the lowest perturbation magnitude ( $\epsilon = 0.01$ ), TrustGuard achieves 95.3% detection accuracy, significantly outperforming the best baseline method (IoT Security Framework at 87.6%). This advantage persists across increasing perturbation magnitudes, with TrustGuard maintaining 87.5% accuracy even at the highest perturbation level ($\epsilon = 0.1$), compared to 72.3% for IoT Security Framework and 68.9% for Visualization-based Trust Enhancement. Notably, TrustGuard exhibits a more gradual decline in performance as perturbation magnitude increases, with a

drop of only 7.8 percentage points from $\epsilon = 0.01$ to $\epsilon = 0.1$. In contrast, the baseline methods show much steeper declines: IoT Security Framework drops by 15.3 percentage points, and Visualization-based Trust Enhancement by 16.3 percentage points over the same range. This resilience to increasing perturbation strength is particularly valuable for transportation systems, where attacks might employ stronger perturbations to overcome defenses, potentially leading to more severe consequences. The superior performance of TrustGuard at high perturbation magnitudes can be attributed to its multi-layered defense architecture and the effectiveness of the Adaptive Perturbation Filtering algorithm in identifying frequency-domain patterns characteristic of adversarial perturbations. These mechanisms remain effective even as perturbation strength increases, providing robust protection across a wide range of attack scenarios. This consistent performance advantage across different perturbation magnitudes further validates TrustGuard's effectiveness as a comprehensive defense framework for IoT-based transportation systems.

## 6. DISCUSSION

This section discusses the implications of our experimental results, the limitations of the TrustGuard framework, and potential applications in real-world transportation systems.

### 6.1. Key Findings and Implications

As an outcome of our TrustGuard experimental evaluation, we can convey a number of important findings on the possible security of IoT-based transportation systems. First, the presented table of the analysis of the TrustGuard versus single-layer strategies shows that multi-layered defense is necessary to have a full protection against the adversary attacks. TrustGuard works well at both an operational level and detection accuracy because by dividing the defensive capabilities along the edge, fog and cloud servicing levels based on their computing capabilities, it provides high detection accuracy and high operational efficiency. This observation implies that any future security system for the IoT based systems deployments ought to also consider the same approach forming a layered system instead of having all defensive facilities on a single tier.

Second, we most definitely prove that lightweight detection may indeed be executed at the edge, and it is efficient. Adaptive Perturbation Filtering (APF) algorithm is a lightweight and powerful first contact shield because it consumes only small computational overhead and allows real-time protection even on resource-scarce IoT devices. This shows that conventional wisdom that effective adversarial defense needs significant computational resources is wrong. With its prioritisation on frequency-domain patterns that effectively distinguish the properties of adversarial perturbations, APF can have high detection accuracies and reasonable processing times, which exhibits that lightweight algorithms can generate effective security even on low-spec machines when they are well-designed. Third, the experiments emphasize that the performance of detection is much better when the context-awareness is implemented. Contextual validation mechanism of TrustGuard provides domain-specific knowledge by checking the consistency between sensor readings and the larger picture of transportation context, allowing to improve the precision of detection with this approach. The above finding makes it essential to include application peculiarities and relations to the defensive strategies instead of the mere usage of generic security measures. In the case of transportation systems,

this context-awareness is very useful because there are physical and logical restrictions determining the direction and patterns of traffic and vehicle movements.

Fourth, dynamic resource allocation mechanism in TrustGuard shows that dynamic security contributes more towards protection and performance. TrustGuard balances defense strength and the level of threat along with the available resources enabling it to achieve high rates of detection whilst at the same time having a minimal effect on the operations. This observation implies that in the future, security systems need to be shifted more towards a dynamic rather than the fixed defensive layout to adapt to dynamic and fluctuating threat patterns and operational parameters. Lastly, the uniform TrustGuard effectiveness across the variety of attacks proves the idea of broad-based defense strategies, which take into consideration the wide range of threats instead of targeting particular attack methods. Such flexibility is also critical in the transportation systems where attackers can use different methods to accomplish their objectives.

### 6.2. Limitations and Future Work

Although it demonstrates good outcomes, TrustGuard is characterized by multiple limitations, which need to be eliminated in further research. A major constraint of research, in particular, was that TrustGuard was tested based on known types of attacks but expecting adversaries to come up with novel attack tactics that cannot be detected by the existing techniques. The potential areas of future investigation are: the superiority of TrustGuard to deal with the changing patterns of attacks, the possible integration of the unsupervised learning methods, allowing the identification of new types of anomalies without the need of labeled data. Such a strategy would enhance zero-day attack resilience of the system and minimize the necessity of regular updates leading to new attack vectors correction. The other weakness is that TrustGuard needs to be calibrated before the actual functioning so as to draw the initial patterns of the genuine transportation information. This may be a time-consuming activity and may have to be repeated as the transportation patterns may change over time e.g. at different times of the year or as the pattern of traveling changes. One of the potential future developments would be to investigate more effective ways to conduct the calibration procedure, which would be subject to change automatically without a need to continue the maintenance overhead of operating the system. Since the validity of the distributed framework is directly linked to its deployability in practice, self-adaptive methods which frequently refresh baseline models using known legitimate traffic might greatly enhance deployability.

System integration, too, becomes an issue because integrating TrustGuard into infrastructures currently involved in transportation systems might need changes to be applied on backward systems. Further study ought to be directed to investigate means of seamless integration that cause minimum interference with the present operations which could be in the modular dimension, whereby these modules could be integrated into various system structures without massive alterations. Interfaces and compatibility layers should be standardized, and it may help to connect with various transportation systems, which extends the range of the framework practical use. A fourth limitation that can be viewed in terms of privacy consideration is the need to collect and analyze transportation data to ensure security as this will bring about the security of the data that can be regarded as sensitive information. Adversity against privacy-preserving adversarial defense should be a future study that ensures security protection of sensitive data. Such methods as federated learning, different

privacy, and secure multi-party computation might allow joint defense of several transportation systems in terms of security and privacy, where raw data is not shared.

## 7. CONCLUSION

This paper presented TrustGuard, a novel multi-layered framework for ensuring adversarial robustness in IoT-based smart transportation systems. Our comprehensive experimental evaluation demonstrates that TrustGuard significantly outperforms state-of-the-art defense mechanisms in detection accuracy (92.6 percent vs 82.1 percent ) , computational efficiency (23ms vs. 96ms processing time), and resilience across various attack scenarios. The key contributions of this work include: (1) a multi-layered defense architecture that optimally distributes security functions across IoT infrastructure layers, (2) the Adaptive Perturbation Filtering algorithm providing lightweight yet effective edge-level defense, (3) context-aware detection mechanisms specifically optimized for transportation systems, and (4) dynamic resource allocation that adapts defensive intensity to threat levels. TrustGuard addresses critical limitations in existing approaches by enabling real-time protection on resource-constrained devices while maintaining comprehensive security coverage. Statistical analysis confirms that our performance improvements are significant ( p less than 0.01 ), validating TrustGuard's effectiveness as a practical solution for securing IoT-based transportation systems against adversarial attacks. Future research should focus on enhancing adaptability to emerging attack patterns, developing more efficient calibration methods, and incorporating privacy-preserving techniques to further improve deployability in real-world transportation infrastructure.

## REFERENCES

[1] H. HaddadPajouh, A. Dehghantanha, R. Parizi, M. Aledhari, H. Karimipour, "A survey on internet of things security: Requirements, challenges, and solutions," *Internet of Things*, vol. 14, p. 100129, 2019, doi: 10.1016/j.iot.2019.100129.

[2] H. Ahmed, A. Nasr, S. Abdel-Mageid, H. Aslan, "A survey of IoT security threats and defenses," *International Journal of Advanced Computer Research*, vol. 9, no. 45, pp. 325–350, 2019, doi: 10.19101/ijacr.2019.940116.

[3] M. Liu, Z. Zhang, Y. Chen, J. Ge, N. Zhao, "Adversarial attack and defense on deep learning for air transportation communication jamming," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 1, pp. 973–986, 2023, doi: 10.1109/tits.2023.3262347.

[4] Y. Saleem, N. Crespi, M. Rehmani, R. Copeland, "Internet of things-aided smart grid: technologies, architectures, applications, prototypes, and future research directions," *IEEE Access*, vol. 7, pp. 62962–63003, 2019, doi: 10.1109/access.2019.2913984.

[5] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal, B. Sikdar, "A survey on IoT security: application areas, security threats, and solution architectures," *IEEE Access*, vol. 7, pp. 82721–82743, 2019, doi: 10.1109/access.2019.2924045.

[6] I. Goodfellow, P. McDaniel, N. Papernot, "Making machine learning robust against adversarial inputs," *Communications of the ACM*, vol. 61, no. 7, pp. 56–66, 2018, doi: 10.1145/3134599.

[7] V. Azamfirei, F. Psarommatis, Y. Lagrosen, "Application of automation for in-line quality inspection, a zero-defect manufacturing approach," *Journal of Manufacturing Systems*, vol. 67, pp. 1–22, 2023, doi: 10.1016/j.jmsy.2022.12.010.

[8] J. Bhat, S. AlQahtani, M. Nekovee, "FinTech enablers, use cases, and role of future internet of things," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 1, pp. 87–101, 2022, doi: 10.1016/j.jksuci.2022.08.033.

[9] M. Usama et al., "Unsupervised machine learning for networking: techniques, applications and research challenges," *IEEE Access*, vol. 7, pp. 65579–65615, 2019, doi: 10.1109/access.2019.2916648.

[10] X. Liu et al., "Privacy and security issues in deep Learning: a survey," *IEEE Access*, vol. 9, pp. 4566–4593, 2020, doi: 10.1109/access.2020.3045078.

[11] A. Chatzimparmpas, R. Martins, I. Jusufi, K. Kucher, F. Rossi, A. Kerren, "the state of the art in enhancing trust in machine learning models with the use of visualizations," *Computer Graphics Forum*, vol. 39, no. 3, pp. 713–756, 2020, doi: 10.1111/cgf.14034.

[12] M. Shafiq, Z. Gu, "Deep residual learning for image recognition: a survey," *Applied Sciences*, vol. 12, no. 18, p. 8972, 2022, doi: 10.3390/app12188972.

[13] H. Dong et al., "AI-enhanced biomedical micro/nanorobots in microfluidics," *Lab on a Chip*, vol. 24, no. 5, pp. 1419–1440, 2024, doi: 10.1039/d3lc00909b.

[14] Z. Ahmad, A. Khan, C. Shiang, J. Abdullah, F. Ahmad, "Network intrusion detection system: a systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, 2020, doi: 10.1002/ett.4150.

[15] Z. Zhang, H. Hamadi, E. Damiani, C. Yeun, F. Taher, "Explainable artificial intelligence applications in cyber security: state-of-the-art in research," *IEEE Access*, vol. 10, pp. 93104–93139, 2022, doi: 10.1109/access.2022.3204051.

[16] L. Rueden et al., "Informed machine learning - a taxonomy and survey of integrating prior knowledge into learning systems," *IEEE Transactions on Knowledge and Data Engineering*, p. 1, 2021, doi: 10.1109/tkde.2021.3079836.

[17] C. Chen et al., "Toward a thousand lights: decentralized deep reinforcement learning for large-scale traffic signal control," Proceedings of the AAAI Conference on Artificial Intelligence, 2020, doi: 10.1609/aaai.v34i04.5744.

[18] F. Yu et al., "BDD100K: a diverse driving dataset for heterogeneous multitask learning," IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, doi: 10.1109/cvpr42600.2020.00271.

[19] Z. Lyu, M. Guo, T. Wu, G. Xu, K. Zhang, D. Lin, "Towards evaluating and training verifiably robust neural networks," IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, doi: 10.1109/cvpr46437.2021.00429.

[20] S. Asma, "The strangest sort of map: reply to commentaries," *Evolutionary Studies in Imaginative Culture*, vol. 5, no. 2, pp. 75–82, 2021, doi: 10.26613/esic.5.2.247.

[21] M. Ghassemi, T. Naumann, P. Schulam, A. Beam, I. Chen, R. Ranganath, "Practical guidance on artificial intelligence for health-care data," *The Lancet Digital Health*, vol. 1, no. 4, pp. e157–e159, 2019, doi: 10.1016/s2589-7500(19)30084-6.

[22] J. Gawlikowski et al., "A survey of uncertainty in deep neural networks," *Artificial Intelligence Review*, vol. 56, no. S1, pp. 1513–1589, 2023, doi: 10.1007/s10462-023-10562-9.