Robust Federated Learning Frameworks Against Data Flipping Threats in Autonomous Vehicles

by

Riadh Ben Chaabene

THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT OF A MASTER'S DEGREE
WITH THESIS
M.A.Sc.

MONTREAL, JANUARY 7, 2026

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

M. Mohamed Cheriet, Thesis supervisor
Département de génie des systèmes, École de technologie supérieure

Mrs. Darine Ameyed, Thesis Co-Supervisor
Département d'informatique et de mathématique, Université du Québec à Chicoutimi

Mrs. Imen Benzarti, Chair, Board of Examiners
Département de génie logiciel et des TI, École de technologie supérieure

M. Jonathan Roy, Member of the Jury
Département d'informatique et de mathématique, Université du Québec à Chicoutimi

M. Diego Elias Damasceno Costa, External Examiner
Department of Computer Science and Software Engineering, Concordia University

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON "DECEMBER 12, 2025"

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

First of all, I would like to acknowledge with profound appreciation those who have given me their valuable support throughout this project.

Starting with **Dr.Darine Ameyed**, my academic supervisor, for her belief in my potential, understanding as well as her patience throughout this work. She made me appreciate the work and has introduced me to the research field. It was a honor to work and debate with you all along this study.

****

Special thanks goes to my research director **Prof.Mohamed Cheriet**, for giving me this opportunity to work under his direction, his feedbacks and comments when needed. It's such an honor to work with such a decorated professor. Thank you a lot.

****

I would also like to show my deepest gratitude to my other academic supervisor, **Dr.Fehmi Jaafar**, for the support and engagement during this master thesis. Her useful comments, remarks and observations always allowed me to find the best solutions. She has been so generous in his time and support. Thank you, it was a privilege working with you.

****

Furthermore, I would like to present my gratitude to all my professors of The École de technologie supérieure and my colleagues at Synchromedia Lab for providing me the background needed for study and research.

****

Last but by no means least, I would like to offer this work to my parents **Raoudha** and **Khaled** and to thank them for their love, tenderness and support. I am deeply indebted to them for

believing in me and giving me a wonderful model of hard work and perseverance.

My grandmother, **Radhia**, whose unwavering support and heartfelt prayers have been my guiding light—may God bless you and watch over you always.

A special thanks to my two aunts **Dr.Ishraf Zaoui** and **Dr.Maha Zaoui** whom I love very much.

To my sister **Rouaa**, being my rival all my life, she pushed me to my limits by her continuous success. Thank you for being there, keep pushing and I will follow.

to all my Canadian family and to all my lovely friends, especially **"Lucile"**, **"Aurélie"**, **"Guillame"**, **"Anna"**, **"Iona"**, **"Max & JF"**, **"Nadège"**, **"Clara"** and all the friends that I meet along the way, for their love and encouragements. I really wish all the best to all of you. This work would not have been possible without your endless support.

Cadres d'Apprentissage Fédéré Robustes Protégant Contre les Menaces de Retour d'Étiquettes pour les Véhicules Autonomese

Riadh Ben Chaabene

RÉSUMÉ

L'Apprentissage Fédéré (AF) s'impose comme une approche prometteuse pour permettre un apprentissage collaboratif respectueux de la vie privée entre plusieurs clients distribués. Sa capacité à exploiter de grandes quantités de données tout en maintenant la confidentialité en fait une solution adaptée aux véhicules autonomes (VA), où les données sont sensibles et décentralisées. Cependant, la nature distribuée de l'AF expose le système à de nouvelles menaces, notamment les attaques par inversion d'étiquettes (Label-Flipping, LF), dans lesquelles des participants malveillants modifient volontairement leurs données locales pour dégrader la performance du modèle global. Cette thèse s'attaque à ces vulnérabilités en proposant un cadre complet garantissant la scalabilité et la robustesse des systèmes de VA basés sur l'AF.

La méthodologie adoptée est structurée en trois parties principales. Première partie – Développement du cadre d'AF : Nous avons conçu et implémenté un environnement fédéré réel intégrant des modèles de Réseaux de Neurones Convolutifs (CNN) et d'Apprentissage par Renforcement (RL) sur la plateforme SunFounder PiCar. Cette configuration a permis un apprentissage collaboratif entre véhicules tout en préservant la confidentialité des données.

Deuxième partie – Processus d'attaque : Nous avons simulé des attaques d'inversion d'étiquettes à différents niveaux de disponibilité adversariale (α) afin d'analyser le comportement du système en conditions hostiles. Un seul client malveillant avec une disponibilité élevée (α = 0,9) a entraîné une baisse de plus de 25 % du rappel des classes sources et une diminution de 20 % de la précision globale, illustrant la gravité des attaques LF.

Troisième partie – Mécanisme de défense (FALCON) : Nous avons proposé FALCON (Federated Anomaly Learning and COllaborative Network), une architecture de défense multi-niveaux intégrant la Détection d'Anomalies Fédérée (FAD), l'Analyse en Composantes Principales (ACP) et les Machines à Vecteurs de Support Multiclasse (MCSVM). FALCON applique une détection locale d'anomalies, un vote pair-à-pair collaboratif et une détection graphique au niveau serveur à l'aide de Réseaux de Neurones Graphiques (GNN) afin d'identifier et de neutraliser les attaquants persistants.

Les évaluations expérimentales ont démontré que le modèle CNN atteint une précision de classification de 94,2 % avec une latence moyenne de 42 ms par image, tandis que le modèle RL obtient un taux de réussite en navigation de 91,8 % et une récompense cumulative moyenne de 1365. Sous attaque LF, la performance du modèle s'est fortement dégradée ; néanmoins, FALCON a permis de restaurer la précision et le rappel globaux à des niveaux quasi optimaux, identifiant plus de 90 % des mises à jour adversariales et réduisant le taux de succès des attaques à moins de 5 %.

Dans l'ensemble, cette thèse démontre la faisabilité de l'intégration de l'Apprentissage Fédéré dans les systèmes de véhicules autonomes tout en répondant efficacement aux vulnérabilités adversariales et aux contraintes de scalabilité. L'architecture FALCON proposée constitue une avancée majeure vers des cadres d'AF sécurisés, évolutifs et résilients pour la conduite autonome réelle et d'autres applications critiques pour la sécurité.

Mots-clés: Apprentissage Fédéré, Véhicules Autonomes, Attaques Adversariales, Retourner les étiquettes

Robust Federated Learning Frameworks Against Data Flipping Threats in Autonomous Vehicles

Riadh Ben Chaabene

ABSTRACT

Federated Learning (FL) has emerged as a promising paradigm for privacy-preserving collaborative training across distributed clients. Its ability to enable large-scale learning while maintaining data confidentiality makes it particularly suitable for autonomous vehicles (AVs), where data is sensitive and decentralized. However, the distributed nature of FL introduces new security challenges, notably label-flipping (LF) attacks, in which malicious participants intentionally corrupt local datasets to degrade the global model's performance. This thesis addresses these vulnerabilities by proposing a comprehensive framework that ensures both scalability and robustness in FL-based AV systems.

The methodology is structured into three key parts. Part One – Development of the FL Framework: We designed and implemented a real-world FL environment integrating Convolutional Neural Networks (CNN) and Reinforcement Learning (RL) models on the SunFounder PiCar platform. This setup enabled collaborative training between vehicles while preserving data privacy.

Part Two – The Attack Process: We simulated label-flipping attacks under different adversarial availability levels α to analyze the system's behavior under hostile conditions. A single malicious participant with high availability (α = 0.9 reduced the source class recall by over 25% and caused a 20% global accuracy drop, highlighting the severity of LF threats.

Part Three – Defense Mechanism (FALCON): We introduced FALCON (Federated Anomaly Learning and COllaborative Network), a multi-layer defense architecture integrating Federated Anomaly Detection (FAD), Principal Component Analysis (PCA), and Multi-Class Support Vector Machines (MCSVM). FALCON applies local anomaly detection, peer-to-peer anomaly voting, and server-level graph-based detection using Graph Neural Networks (GNN) to identify and neutralize persistent adversaries.

Experimental evaluations demonstrated that the CNN achieved a 94.2% classification accuracy with a latency of 42 ms per frame, while the RL model reached a 91.8% navigation success rate and an average cumulative reward of 1365. Under LF attacks, model performance degraded significantly; however, FALCON restored global accuracy and recall to near-optimal levels, identifying over 90% of adversarial updates and reducing attack success rates to below 5%.

Overall, this thesis demonstrates the feasibility of integrating FL into AV systems while effectively addressing adversarial vulnerabilities and scalability constraints. The proposed FALCON architecture represents a major step toward secure, scalable, and resilient federated learning for real-world autonomous driving and beyond.

Keywords: Federated Learning, Autonomous Vehicles, Adversarial Attacks, Label-Flipping

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS

FALCON	Federated Anomaly Learning and COllaborative Network
FL	Federated Learning
AV	Autonomous Vehicles
AIS	Autonomous Intelligence System
LF	Label Flipping
A2D2	Audi Autonomous Driving Dataset
CARLA	Car Learning to Act
CNN	Convolutional Neural Networks
RL	Reinforcement Learning
ML	Machine Learning
FAD	Federated Anomaly Detection
PCA	Principal Component Analysis
SVM	Support Vector Machines
MCSVM	Multi-Class Support Vector Machines
PPO	Proximal Policy Optimization
MDP	Markov Decision Process
GM	Global Model
FedAvg	Federated Averaging Algorithm
TP	True Positive

FP	False Positive
TN	True Negative
F1	F1 Score
IoT	Internet of Things
TPU	Tensor Processing Unit
AGX Orin	NVIDIA Jetson AGX Orin Developer Kit
PiCar	SunFounder PiCar Platform
Non-IID	Non-Independent and Identically Distributed (Data Distribution)
i.i.d.	Independent and Identically Distributed

LIST OF SYMBOLS AND UNITS OF MEASUREMENTS

x	Input data sample
y	Ground truth label
$\hat{y}$	Predicted output
$F(x, \{W_i\})$	Function representing transformation in a residual block of the CNN
L_{CE}	Cross-entropy loss function
N	Number of data samples
C	Total number of classes
s_t	State at time t
a_t	Action taken at time t
r_t	Reward received at time t
$P(s_{t+1}\|s_t, a_t)$	Transition probability function in Markov Decision Process (MDP)
$\pi(a_t\|s_t)$	Policy function determining the action a_t given state s_t
$L^{\text{CLIP}}(\theta)$	Clipped Proximal Policy Optimization (PPO) loss function
α	Weighting factor in reward function or availability level of adversary
β	Coefficient for adversarial impact on source class recall
ρ	Source class recall
A	Global model accuracy
γ	Coefficient for accuracy degradation
t	Training round

w_t	Global model parameters at round t
w_t^i	Local model parameters from the i-th participant at round t
$\mathbf{W}_i$	Update vector from the i-th participant
$\mathbf{E}_k$	Matrix of top k eigenvectors of the covariance matrix
Σ	Covariance matrix
$\mathbf{w}_m$	Weight vector for the m-th classifier in MCSVM
b_m	Bias term for the m-th classifier in MCSVM
D_{test}	Testing dataset
D_i	Dataset for the i-th participant
$\mathcal{D}$	Experience replay buffer
$\mathcal{L}(W)$	Loss function to optimize model weights
$p - value$	It is a measure of the strength of evidence against the null hypothesis
Cliff's δ	A non-parametric effect size measure ranges between -1 and 1
ms	Milliseconds (time measurement)
timesteps	Simulation steps in RL training
pixels	Resolution of input images
rounds	Training iterations in federated learning
epochs	Iterations over the full dataset

CHAPTER 1

INTRODUCTION

In today's increasingly interconnected world, the convergence of physical and digital domains via autonomous intelligence systems (AIS) is transforming industries, cultural norms, and our interactions with technology. AIS comprises a wide range of systems in which computational and physical processes are inextricably linked, from smart grids to industrial automation. Autonomous Vehicles (AV) are among the most transformative applications of AIS, representing a peak of technical progress by merging real-time data processing, machine learning, and sensory input into the world of mobility.

Through advances in sensor technology and artificial intelligence advancements, AV have the potential to completely change how we navigate. These cars use a sophisticated ecosystem of sensors, cameras, lidar, radar, and sophisticated algorithms to perceive and respond to their surroundings with minimal human intervention.

As a critical application of AIS, AV Rajasekhar & Jaswal (2015)Martínez-Díaz & Soriguera (2018) represent a rapidly developing field that has the potential to revolutionize human transportation. Many researchers and experts estimate that AV market is forecasted to grow, reaching $13.6 trillion by 2030 compared to the current evaluation of $1.9 trillion Insights (2024). Since vehicular transportation is considered the most dangerous motorized transportation option due to human error that contributes to over 90% of traffic accidents, AV integration in our society is expected to reduce accidents significantly, potentially saving 370,000 lives annually worldwide by 2030 and crash rates by up to 45% (National Highway Traffic Safety Administration (2015)).

The usage of machine learning in this field proved to be very promising for its implementation in various applications, such as autonomous driving Zhang, Springenberg, Boedecker & Burgard (2016), complex environment navigation Nguyen, Nguyen, Tran, Tjiputra & Tran (2020) , lane following and switching Gurghian, Koduri, Bailur, Carey & Murali (2016) and traffic calculation Yu *et al.* (2021). Recent advances in this area rely heavily on machine learning, that requires extensive training data. centralized training provides greater accuracy for autonomous driving

solutions by using already known and controlled data. Such data are often shared with cloud systems for real-time processing or model training, creating vulnerabilities and neglect data privacy and third party involvement protocols. Cyberattacks on this private information could result in identity theft, misuse of surveillance data, or even bodily injury if malicious actors take over the car's systems. Thus, protecting privacy while preserving the effectiveness of AV and usefulness is a critical concern.

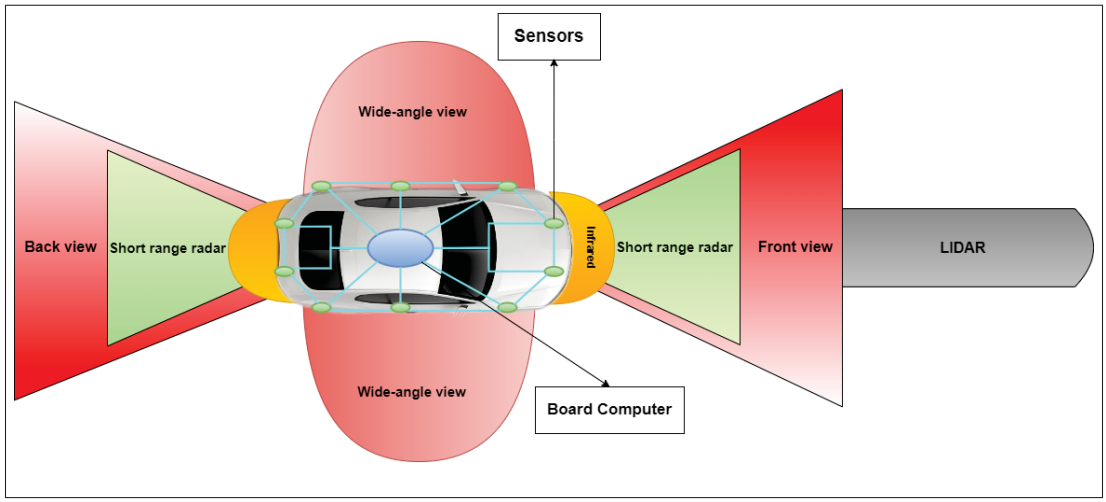


Figure 1.1 Multi-Sensor Perception Layout of an Autonomous Vehicle, Showing LIDAR, Radar, and Camera Coverage Zones

To address the privacy concerns associated with centralized data storage and processing, FL has become a potential paradigm to address privacy issues related to centralized data processing and storage. FL enables AV to jointly train machine learning models without exchanging raw data. Only model updates are transmitted to a central server, which aggregates them to enhance the global model, after each vehicle processes its data locally. FL Kairouz *et al.* (2021) McMahan, Moore, Ramage, Hampson & Arcas (2017a) Chaabene, Amayed & Cheriet (2022) has gained increasing prominence in machine learning, enabling models to learn from distributed devices in diverse contexts. In the FL framework, participating peers undertake training of a global model that they receive from the central server using their own local datasets. After processing their local data, these peers generate model updates, which they then send back to the server. The server's role is to gather and aggregate the various updates it receives from

all the peers, ultimately leading to the creation of an enhanced global model. Following this aggregation process, the updated global model is redistributed to the peers, setting the stage for the next iteration of training and since only model updates are shared rather than entire datasets, this approach optimizes network resources. One significant advantage of FL lies in its ability to enhance privacy. By retaining local data on peer devices and not transferring it to a central server, FL significantly mitigates the risks associated with data breaches and privacy violations. This is particularly crucial in an era where data privacy concerns are paramount. Additionally, FL contributes to scalability by distributing computational workload enables large-scale collaboration among diverse nodes, such as vehicles from different manufacturers, without risking data leaks, rather than relying solely on a central server's computing resources. This decentralized approach not only improves efficiency but also allows for a broader range of devices to participate in the training process, making FL a highly attractive option for developing machine learning models in a secure and scalable manner Bonawitz *et al.* (2019).

However, like any distributed learning paradigm, federated learning (FL) faces several challenges and limitations. Its decentralized nature exposes the global model to potential attacks originating from malicious participants. In particular, the lack of direct control at the server side Li, Sahu, Talwalkar & Smith (2020) increases the risk of adversarial behavior, as participating clients can deviate from the expected training protocol without immediate detection.

Among the various threats targeting federated learning, data poisoning represents one of the most severe challenges. By exploiting model distribution, attackers can inject corrupted or mislabeled training data with the objective of degrading model performance, preventing convergence, or forcing systematic misclassification of inputs with specific characteristics. Unlike privacy-focused attacks (e.g., inference or model inversion), data poisoning directly compromises the integrity and reliability of the learned model, which is particularly critical in safety-sensitive applications such as autonomous driving.

One prominent class of targeted data poisoning attacks is the label-flipping (LF) attack Biggio, Nelson & Laskov (2012b). In LF attacks, adversaries tamper with their local training datasets

by deliberately flipping labels from a source class to a target class. For example, an attacker may relabel traffic light images from *"red"* to *"green"*, or modify *"Speed Limit 30"* signs to *"Speed Limit 90"*. Such manipulations can lead autonomous vehicles to take unsafe decisions, including failing to stop at intersections or driving at inappropriate speeds.

Crucially, LF attacks require no modification to the FL protocol itself. The attacker trains the local model using the same hyperparameters, loss function, and optimization strategy provided by the server. As a result, the generated model updates appear statistically similar to those of honest clients. Once submitted to the server and aggregated with benign updates, these poisoned contributions progressively corrupt the global model, leading to noticeable drops in accuracy and reliability over successive training rounds. This combination of simplicity, stealth, and effectiveness makes label-flipping a particularly realistic and impactful threat in federated learning systems for autonomous vehicles.

Multiple studies have been conducted to address these issues. Li, Ngai, Ye & Voigt (2021b) investigate peers who have the same goals as attackers, which causes a large percentage of false positives when sincere peers have comparable local data. Zhou, Zheng, Huang, Shu & Jia (2023) present RoHFL, a hierarchical FL framework for the Internet of Vehicles that uses similarity-based reputation scoring and logarithm-based normalization to thwart poisoning attacks. Nevertheless, combining these techniques entangle the aggregate procedure. The OQFL framework Yamany, Moustafa & Turnbull (2023) uses quantum-behaved particle swarm optimization (QPSO) to modify hyperparameters in order to identify hostile cars. However, because the model must be reinitialized and retrained from start, there is a large computational overhead every time a search is conducted.

1.0.1 Purpose of the study

The study aims to develop a robust security detection system capable of identifying and mitigating attacks by malicious users seeking to perform data poisoning—specifically LF attacks—within FL frameworks used for training AV. Under such attacks, FL models, which rely on decentralized

collaborative training methods, face significant degradation in the integrity and performance of their underlying models.

In an effort to enhance the safety and reliability of FL, our system integrates advanced mechanisms to detect and counter adversarial behaviors. The research seeks to ensure data integrity and model integrity, ultimately contributing to the broader goal of secure and efficient autonomous vehicle systems.

The study is broadly divided into three main parts:

- A review of existing FL frameworks, their vulnerabilities, and their handling of adversarial attacks like LF.
- The design and implementation of a detection and mitigation system to counter data poisoning in the FL processes of AV.
- A comprehensive performance evaluation of the system through simulated real-world experiments, focusing on model accuracy and system resilience against malicious participants.

1.0.2 Research questions

Highlighting the challenges of data poisoning attacks, especially LF in the FL of AV, led to our focus on addressing these critical security concerns. These problems have led us to formulate the following research questions:

R1: How can LF attacks influence the integrity, convergence, and overall performance of FL models used in autonomous vehicle environments?

R2: What are the main challenges in identifying and isolating malevolent users in decentralized learning systems with non-IID and dynamically changing data?

R3: How can validation and anomaly detection methods be structured to provide distributed client reliability and trust while maintaining communication efficiency and privacy?

R4: To what extent can FL maintain its robustness and accuracy across increasing levels of adversarial involvement and data manipulation?

Together, these questions shape the thesis's analytical and experimental approach, resulting in the creation and evaluation of an adaptive protection strategy for AV based FL systems.

1.0.3 Research objectives

Expanding on the previously identified research questions, the overall objective of this research initiative is to help enhance safe and reliable FL in autonomous-vehicle environments. The ultimate goal is to examine how federated models can remain strong and trustworthy in encounters of adversary data manipulation, namely LF attacks. To accomplish this, the research targets the following specific objectives:

O1: Evaluate the vulnerability of FL models to LF attacks in autonomous vehicle situations. Determine the reduction in accuracy, recall, and convergence caused by malicious clients injecting corrupted labels at various levels of involvement.

O2: Identify flaws in existing FL safeguarding techniques for heterogeneous and non-IID data distributions. Evaluate representative cutting-edge approaches and identify the conditions in which their detection precision and scalability deteriorate.

O3: Define a conceptual framework for detecting and validating harmful updates throughout the FL lifecycle. Create a comprehensive picture of local, collaborative, and global validation concepts that can be applied in privacy-preserving systems.

O4: Validate the suggested approach with experimental simulations of benchmark and real autonomous-vehicle datasets. Verify the conceptual framework's effectiveness by measuring robustness, accuracy retention, and detection efficiency across various attack intensities.

1.0.4 Approach and boundaries of the study

In this research, we propose a robust security detection system to ensure the integrity and reliability of the training process to answer our research questions and address the challenges posed by LF attacks in FL for AV.

Our approach will, therefore, be a contribution to the development and integration of advanced detection mechanisms into the FL frameworks. The system will be focused on detecting malicious user behaviors, hence mitigating their impact and retaining model performance in adversarial environments. This work shall exploit:

- Advanced anomaly detection techniques for LF attack scenarios.
- Mechanisms for improving data quality and ensuring trustworthiness in FL contributions.
- Strategies for protecting collaborative model training processes without compromising scalability or efficiency.

The thesis focuses its efforts, therefore, on the detection and mitigation of data poisoning attack, specifically LF attacks in FL frameworks. Other broad areas of concern related to the security of cyber-attacks, other than data poisoning - backdoor attack for instance, data leakage, and congestion in a federated network - are outside the scope of this work. The optimization of computational resources used by FL is also outside the scope of our work, although it could be plausible for future scope, as mentioned in 6.1.2.

1.0.5 Contributions

The aim of this work is to develop a robust detection system to address the challenges of LF attacks and data poisoning in FL frameworks in a real-world scenario, specifically for autonomous vehicle training. Our proposed solution aims to improve the reliability and security of FL while maintaining efficient collaboration among participants.

We designed a security-driven approach to mitigate data manipulation risks and ensure model integrity. Our system provides:

- Effective detection mechanisms to identify malicious users that attempt to manipulate labels.
- Techniques to maintain data quality and ensure trust in FL contributions.
- A framework that can adapt to heterogeneous federated learning configurations, as well as other autonomous intelligence systems, particularly those used in training autonomous vehicle systems.

Compared to traditional FL approaches, which emphasize model sharing across distributed data sources, our contribution focuses on strengthening the security of the training process against adversarial behaviors. This ensures that the collaborative training process can function without compromising model performance or data integrity.

The key aspect of our contribution is the ability to effectively mitigate malicious activity without hindering the collaboration and scalability of FL. Our system is designed to be flexible, supporting diverse learning environments while maintaining a high standard of security and resilience.

To counter this threat, we propose FALCON (Federated Anomaly Learning and Collaborative Network), a novel defense framework that integrates Principal Component Analysis (PCA) and Multi-Class Support Vector Machines (MCSVM) to differentiate between malicious and legitimate updates. PCA performs dimensionality reduction and identifies anomalies in principal component distributions, while MCSVM classifies updates based on extracted features, revealing inconsistencies indicative of LF attacks. To enhance robustness, we adopt an ensemble approach, merging multiple SVM classifiers to improve detection accuracy. Furthermore, we introduce Federated Anomaly Detection (FAD), a multi-layered security mechanism that combines:(i) Client-side anomaly detection, where local models flag suspicious updates before transmission. (ii) Peer-to-peer validation, leveraging outlier scores, to cross-validate integrity among clients. Finally, a Server-level Graph Neural Network (GNN) monitoring, tracking client update similarities over multiple rounds to identify and eliminate persistent adversarial participants before aggregation.
We validated the FALCON framework through comprehensive experiments on multiple benchmark and real-world datasets, demonstrating its effectiveness in detecting and mitigating LF attacks while maintaining model integrity and collaborative performance.

1.0.6 Thesis structure

This thesis is organized into four main chapters.

The first chapter serves as an introduction, presenting the study's context and motivation. It explores the problem statement, research questions, and objectives, while also defining the main contributions of this work.

The second chapter comprises the background and literature review, where we introduce the key concepts relevant to our system, outline the main research trends, and discuss related work. Additionally, we analyze previous contributions in the field, highlighting their limitations and how our study aims to address them.

The third chapter focuses on the development of the federated learning framework. This chapter is self-contained and presents the motivation, system architecture, model design, data sources, and training procedures under benign conditions. It also introduces the global experimental design and evaluation criteria used throughout the thesis.

The fourth chapter is dedicated to modeling the label-flipping attack process. It describes the threat model, attack assumptions, implementation details, and experimental configurations used to assess the impact of malicious participants on federated learning performance.

The fifth chapter presents the proposed defense mechanisms against label-flipping attacks. It details the multi-layer FALCON defense architecture, the anomaly detection techniques employed, and their integration into the federated learning framework.

The sixth chapter presents a comprehensive experimental discussion and comparison with state-of-the-art federated learning defense mechanisms. It analyzes the robustness of the proposed framework against label-flipping attacks, compares its performance with baseline methods in terms of accuracy, recall, attack success rate, false positive rate, and computational overhead, and discusses the implications of these results with respect to the research objectives.

This is followed by the study's implications in both research and industry contexts.

Finally, the thesis concludes with a summary of the contributions and their significance in solving the research problem. We also discuss the challenges encountered, study limitations, and provide recommendations for future research directions.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

This chapter gives an overview of how FL is applied to autonomous driving systems to improve the intelligence of vehicles while data security and privacy are guaranteed. Besides, it discusses the risks of LF attacks in FL systems and reviews existing studies on their mechanisms, impact, and mitigation strategies. This chapter synthesizes earlier studies to provide insight into the benefits and problems of incorporating FL into autonomous driving. It also highlights the emphasis of our work.

2.1 Background: Distributed learning and data integrity challenges

This section summarizes the essential features pertinent to this work, with an emphasis on distributed learning and its specific approach. It describes the essential concepts, methodologies, and privacy mechanisms of FL, as well as its vulnerabilities, including data poisoning attacks such as LF. Understanding these features is critical for contextualizing the contributions and problems discussed in this work.

2.1.1 Distributed learning: An overview

Distributed learning is the process of training machine learning models on several computer nodes that may be geographically or organizationally dispersed. The goal is to use computational resources and data from many sources to jointly train a shared model. DL systems can generally be classified as:

- **Centralized Coordination:** To maintain an up-to-date global model, a central server aggregates updates from distributed nodes, such as gradients or parameters. Parameter server systems, for instance, allow participants to communicate their adjustments to a central coordinator.

- **Decentralized Coordination:** A central server is not required, as nodes communicate directly and peer-to-peer with one another to distribute updates. This approach is exemplified by gossip-based protocols, in which nodes disseminate updates at random.

Figure 2.1 highlight a visual difference between the discussed type of distribution.

While distributed learning is powerful, it presents various challenges: **Data Privacy:** Sharing data across nodes or servers can result in privacy violations.**Communication Overhead:** The frequent exchange of updates can impede performance. **Security Risk:** Malicious inputs can disrupt the training process.

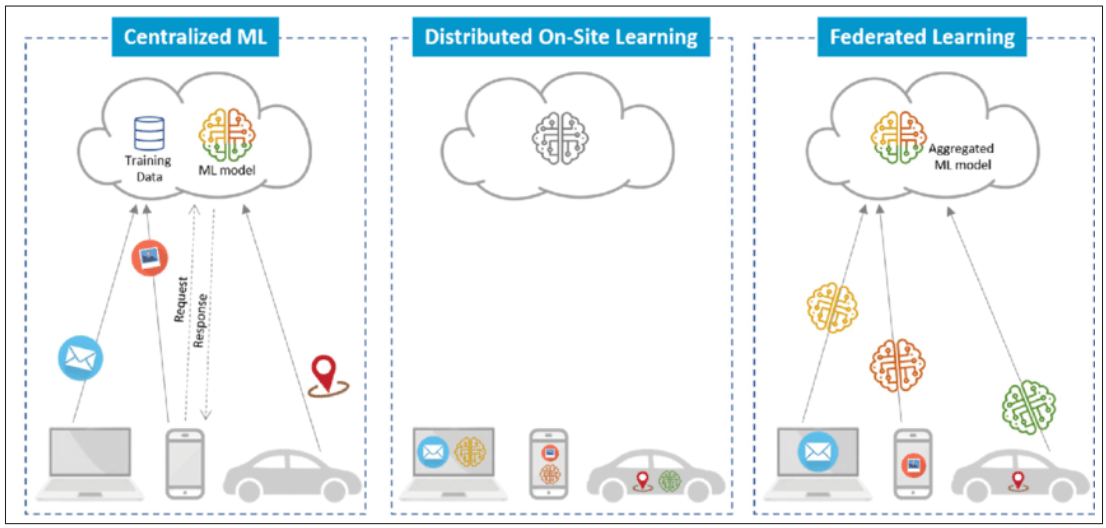


Figure 2.1 Comparison between centralized, decentralized, and federated learning architectures. The figure highlights how data sharing, model aggregation, and communication differ across these paradigms.

2.1.2 Federated learning: privacy-preserving distributed learning

FL is a subset of distributed learning that emphasizes privacy by ensuring data never leaves the local nodes. Instead of sharing raw data, FL aggregates locally trained model updates at a central server to create a global model. This makes it ideal for privacy-sensitive applications such as healthcare, finance, AV, and personalized services. Figure 2.2 illustrate a Federated Learning-style architecture

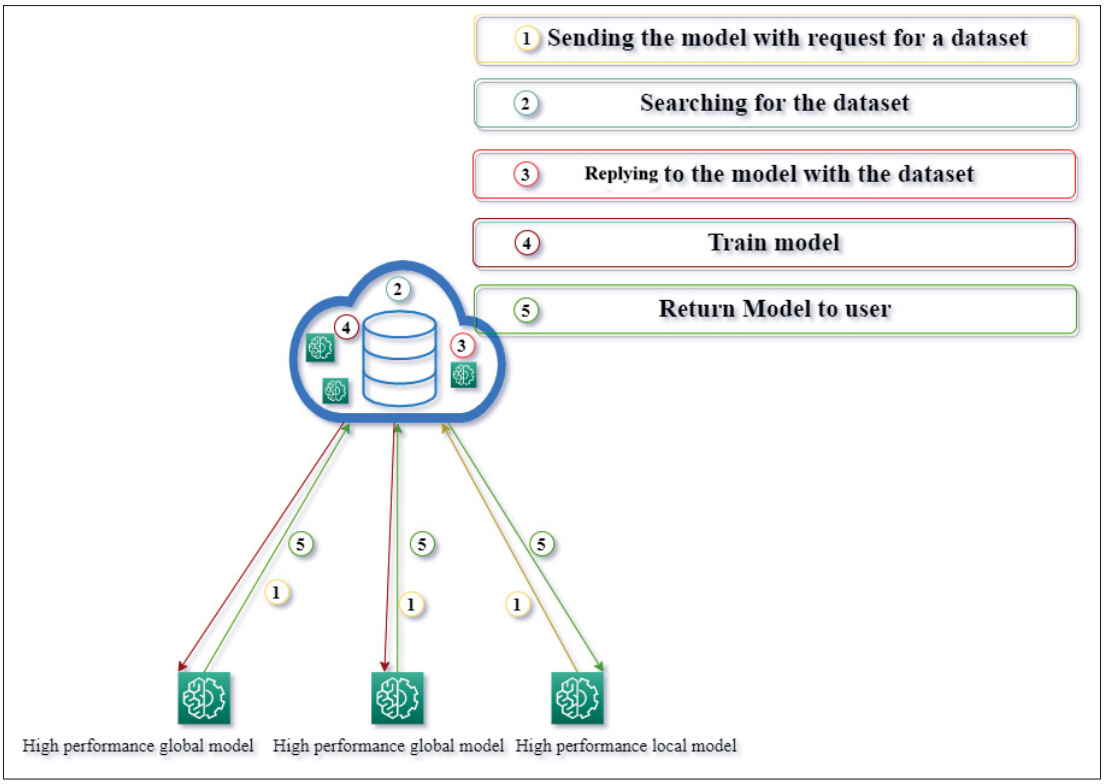


Figure 2.2 Proposed data-centric Federated Learning architecture designed to detect and mitigate label-flipping (LF) attacks in autonomous vehicle (AV) training environments.

2.1.2.1 Key techniques in federated learning

FL employs various methods to handle the challenges of decentralized data and heterogeneous environments:

- **Federated Averaging (FedAvg):** The central server computes a weighted average of model updates from the nodes, thereby reducing communication frequency by combining multiple local updates before sharing them McMahan, Moore, Ramage, Hampson & y Arcas (2017b).
- **Personalized FL:** Addresses data heterogeneity by allowing each node to adapt the global model to its local data Smith, Chiang, Sanjabi & Talwalkar (2017); Fallah, Mokhtari & Ozdaglar (2020).
- **Hierarchical FL:** This implements intermediate aggregation at regional servers to improve scalability and reduce communication Briggs, Fan & Andras (2020b); Liu *et al.* (2020).

- **Asynchronous FL:** Nodes update the global model asynchronously, accommodating devices with varying computational capacities Xie, Koyejo & Gupta (2019); Chen *et al.* (2020a).

2.1.2.2 Privacy techniques in federated learning

Despite FL's decentralized approach, privacy risks persist due to the exposure of model updates, which can leak sensitive information through inference attacks. To mitigate these risks, FL incorporates various privacy-enhancing techniques:

- **Differential Privacy (DP):** This technique adds noise to model updates to obscure individual data contributions while preserving overall model accuracy. An example includes noise injection during gradient computation Dwork, McSherry, Nissim & Smith (2006); Geyer, Klein & Nabi (2017).
- **Secure Multiparty Computation (SMPC):** This approach distributes computations across multiple parties, ensuring that no single party can infer sensitive data Yao (1982); Bonawitz, Ivanov, Kreuter *et al.* (2017).
- **Homomorphic Encryption:** This method encrypts updates so that the server can perform operations on encrypted data without accessing the plaintext Gentry (2009); Aono, Hayashi, Wang & Moriai (2017).
- **Secure Aggregation Protocols:** These protocols ensure that the central server receives only the aggregated model updates, preventing it from accessing individual updates Bonawitz *et al.* (2017).

2.1.3 Data poisoning attacks: A general threat to distributed learning

Data poisoning attacks pose a serious security risk in distributed learning systems, such as FL. These attacks occur when adversaries inject harmful data into the training process, resulting in compromised or biased modelsBiggio, Nelson & Laskov (2012a); Steinhardt, Koh & Liang (2017).

2.1.3.1 Mechanics of data poisoning

- **Training Data Manipulation:** Attackers manipulate data to deceive the model during training. Poisoning can happen on any node in the distributed learning architecture .
- **Model Update Manipulation:** Adversaries send crafted model updates to the server to alter the global model Bagdasaryan, Veit, Hua, Estrin & Shmatikov (2020).

2.1.3.2 Types of data poisoning attacks

- **Targeted Poisoning:** These attacks are designed to cause specific misclassifications or behavior in the model. For example, adversaries might alter stop signs in an autonomous driving dataset so that they are misclassified as yield signs.
- **Untargeted Poisoning:** Aims to degrade the model's overall performance by introducing noise or inconsistency.

2.1.4 Label-flipping attacks: A subset of data poisoning

LF attacks are a subtle yet highly effective form of data poisoning in federated learning. They are particularly dangerous because they exploit the fundamental trust assumption of FL: that each client trains on correctly labeled local data. Unlike other attack vectors, LF attacks do not require breaking encryption, accessing the central server, or manipulating the aggregation process. Instead, an adversary simply corrupts the labels of its local training data, causing the global model to learn systematically incorrect associations.

What makes LF attacks especially challenging is their simplicity and stealth. The malicious client behaves identically to an honest participant in terms of communication and training procedures, and the resulting model updates resemble legitimate gradients. As a result, LF attacks are difficult to distinguish from benign updates using standard aggregation rules such as FedAvg. This property has made label flipping one of the most widely studied and impactful poisoning attacks in both centralized and federated learning settings Biggio *et al.* (2012b).

In the context of autonomous vehicles, LF attacks represent a critical safety threat. For example, mislabeling stop signs or traffic lights can directly influence perception and decision-making modules, leading to unsafe driving behavior. The attack unfolds as follows:

- **Stage 1: Malicious Relabeling.** The adversary systematically alters labels in its local dataset. For instance, all samples labeled as *"red traffic light"* are relabeled as *"green traffic light"*.
- **Stage 2: Local Model Training.** The malicious client trains its local model on the poisoned dataset. The learned model parameters encode incorrect semantic associations between visual features and classes.
- **Stage 3: Poisoned Model Updates.** The client submits its model update to the federated server following the standard FL protocol. The update appears statistically normal and does not raise immediate suspicion.
- **Stage 4: Global Model Corruption.** The server aggregates all client updates, including the poisoned one. Over successive rounds, the influence of poisoned updates accumulates, progressively degrading the global model's accuracy on the targeted classes.

Due to their effectiveness, low cost, and realism in sensor-driven environments, LF attacks are commonly used as a representative poisoning threat for evaluating the robustness of federated learning defenses, particularly in safety-critical autonomous vehicle applications.

Figure 2.3 demonstrate all those stages.

2.1.4.1 Characteristics of label-flipping Attacks

- **Ease of Implementation:** LF requires no direct communication with other nodes or the server, only access to the data on the compromised node Xiao, Biggio, Nelson, Xiao & Eckert (2015).
- **Difficulty of Detection:** Because the flipped labels look like reasonable data points, detecting such attacks in decentralized setups is challenging Tolpegin, Truex, Gursoy & Liu (2020b) .

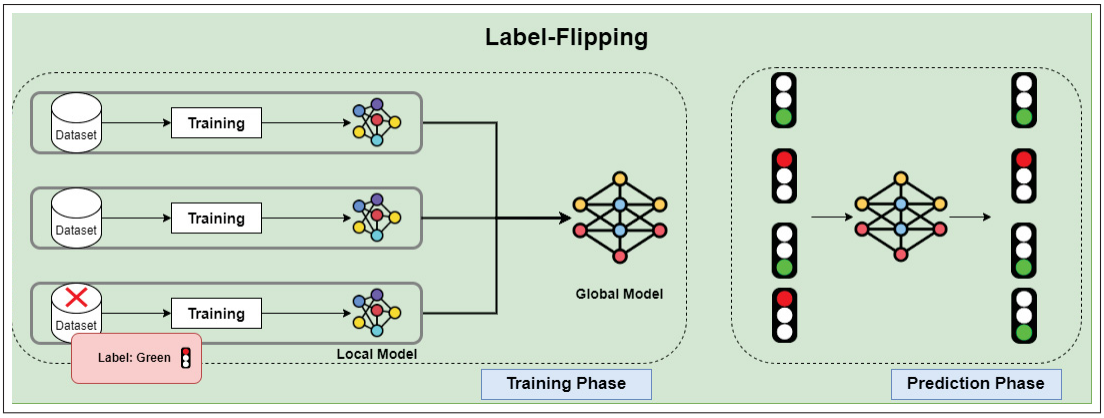


Figure 2.3 Label-flipping (LF) definition and illustrative example: an adversary systematically alters class labels (e.g., turning "pedestrian" into "vehicle") before training, which misguides the global model during aggregation in federated learning.

2.1.4.2 Impact on Distributed Learning

Label-flipping attacks have a significant impact on distributed learning systems by degrading model performance, disrupting convergence, and affecting a wide range of learning architectures. By introducing contradictory labels during training, the model is forced to learn inconsistent mappings between inputs and outputs, which severely impairs its ability to generalize. For instance, in image classification tasks, flipping labels such as "dog" to "cat" can lead to substantial class-level accuracy degradation. Beyond performance loss, poisoned updates interfere with the optimization process, often causing delayed, unstable, or incomplete convergence. These effects are not limited to a specific training paradigm: label-flipping attacks remain effective in centralized, decentralized, and federated learning systems alike, primarily due to the absence of direct mechanisms for verifying the integrity of training data and model updates.

2.1.4.3 Mitigation Strategies for Label-Flipping Attacks

Several mitigation strategies have been proposed in the literature to counter label-flipping attacks in distributed and federated learning systems. Robust aggregation methods, such as Krum and Trimmed Mean, aim to reduce the influence of anomalous or malicious updates by filtering out

outliers during the aggregation process, thereby limiting the impact of poisoned contributions Blanchard, El Mhamdi, Guerraoui & Stainer (2017b). Complementary to aggregation-based defenses, anomaly detection techniques focus on identifying clients whose updates deviate significantly from the expected statistical distribution, enabling the detection and isolation of malicious participants Fung, Yoon & Beschastnikh (2018); Sun *et al.* (2019). Another line of defense involves improving loss resilience by adopting loss functions that are less sensitive to label noise, such as the mean absolute error (MAE), which can enhance robustness against corrupted labels during training Ghosh, Kumar & Sastry (2017). Finally, reputation-based mechanisms track client behavior over time and assign trust scores based on historical reliability, progressively reducing the influence of participants that exhibit suspicious or inconsistent contributions Kang *et al.* (2019); Liu *et al.* (2021b).

2.2 Literature review

We now review existing research on the study's core issues, such as the use of FL in autonomous vehicle systems and the impact of data poisoning attacks, including LF, on FL models. The review emphasizes key findings, techniques, and limitations from previous research to lay the groundwork for understanding the present state of research and identifying gaps that this study seeks to fill.

2.2.1 Autonomous driving system based on federated learning.

As participatory driving models continue to improve in statistical accuracy, increased attention has been paid to improving the safety and effectiveness of their training programs. Traditionally, driving intervention models have relied on centralized training methods. However, centralized training presents challenges such as server computing capacity limitations, data security concerns, and network transmission overheadYaacoub, Noura & Salman (2023). In response, the FD Framework has developed the capacity to nurture these models by privacy concerns, optimizing communication efficiency, and Vehicle to Vehicle (V2V) communication and collaboration . FL

in Autonomous driving systems has been the subject of a verity of research investigations for various purposes Chellapandi, Yuan, Żak & Wang (2023).

2.2.1.1 Communication efficiency and abnormal behavior detection

FL is applied, for instance, in object detection; it enables the AV framework to learn rapidly and with minimal communication costs, which is quite helpful when the data volume far exceeds the ML model's size while also safeguarding the data's privacy. In Barbieri, Savazzi, Brambilla & Nicoli (2022b), LiDAR on CAVs is utilized for object classification through a decentralized FL approach. Through V2V networks, the ML model's parameters are exchanged. Comparing FL to self-learning techniques, it has been experimentally demonstrated that FL is highly effective by comparing their results with a centralized approach. Despite the fact that the accuracy of the object classification was 2% below the centralized model, 89% success compared to 91%, it was able to reduce data transmission by 80%.

Another study Kong *et al.* (2021a) simulate similar results. While working on the identification and recognition of traffic sign, a critical task in AV navigation. Applications include traffic safety and infractions, traffic monitoring, detecting unlawful. The dataset used was collected throughout multiple vehicular participant. It contains 5000 traffic sign images that where distributed to 10 participant clients in order to simulate a real-world vehicle collaboration. The obtained results where compared same as the previous research to a centralized model and the result indicated that the FL model achieved a 92% accuracy rate compared to 94% for the centralized model. But similarly the data transmission was reduced by 75%.

The detection of abnormal vehicle trajectories at traffic crossings has been accomplished through the use of FL in conjunction with OneClass Support Vector Machine (OC-SVM) Koetsier *et al.* (2022a). According to the published results, that where obtained follwing a collaborative training of 50 simulated AV local dataset, the federated strategy enhances anomaly detection's overall accuracy reaching 88% 10% higher compared to single models, while also benefiting specific data owners and reducing communication cost by 60%.

2.2.1.2 Autonomous driving conduct

Other ways where FL provided promising results is predicting steering wheel angles and traffic control. The performance of centralized learning and FL in steering angle prediction was evaluated in P, Rajendran & Panda (2021) under various noise levels, and the outcomes were equivalent. This research also took into account the effects of communication load and interruptions, offering a thorough assessment of the systems. Because of this, FL is appropriate for applications that include a growing number of CAVs, particularly for jobs like steering wheel angle prediction.

The research provided in Zhang, Bosch & Olsson (2021a) showed that using FL in AV significantly improved the quality of the edge models in predicting the steering angle. In particular, the research used optical flow and pictures as two data modalities to estimate steering wheel orientations. While using the KITTI dataset across 15 simulated AV those updates where aggregated using FedAvg. The steering angle prediction accuracy achieved 94%.

In another study Zeng, Semiari, Chen, Saad & Bennis (2022b), by employing FL to update the steering controller parameters dynamically, with 20 AV simulated to send local model updates to the global model, they managed to enhance the steering angle prediction by 10% over other centralized models, hence improving driver comfort and safety.

Moreover, FL is applied in cooperative parameter optimization between several vehicles at traffic junctions, preventing crashes and enhancing driving comfort. In a this study Wu *et al.* (2021b), the authors explored the usage of FL to optimize speed control systems in AV at intersections to prevent collisions. Simulations of 50 AV were carried out, with data exchange confined to model updates. FL increased collision avoidance rates by 20% over non-cooperative approaches while maintaining a 90% reduction in raw data transfer.

By precisely calculating the road friction coefficients, FL is used in Liu, Fu, Zhao, Li & Li (2021a) to improve brake performance in a variety of driving scenarios and settings. The FL model was trained using real-world friction data collected from 30 vehicles. This method

Table 2.1 Federated Learning in Autonomous Vehicles: Key Studies and Their Limitations

Study	**Key Findings (with Statistical Results)**	**Limitations**
Barbieri et al. (2022)	FL achieved 89% object classification accuracy, 2% below centralized methods, with an 80% reduction in data transmission.	Struggled with non-IID data distributions in diverse environments.
Kong et al. (2021)	FL improved traffic sign recognition to 92% accuracy, close to the 94% in centralized training, with a 75% reduction in data sharing.	Communication interruptions slowed model convergence.
Koetsier et al. (2022)	FL+OC-SVM achieved 88% anomaly detection accuracy (10% over standalone), with 60% less communication.	Sensor variability across vehicles affected detection consistency.
Zhang et al. (2021)	94% steering-angle prediction (2% below centralized), resilient to 15% data noise.	Increased computational load on edge devices.
Zeng et al. (2022)	Dynamic FL improved prediction accuracy by 10% and reduced steering false positives by 15%.	Communication-round delays affected convergence.
Liu et al. (2021)	FL improved braking accuracy by 25% and reduced force miscalculations by 15%.	Depended heavily on accurate initial data distributions.
Wu et al. (2021)	FL enhanced collision avoidance by 20% and reduced data sharing by 90%.	Scalability issues as vehicles increased.
Hard et al. (2018)	DP-FL maintained 85% accuracy vs. 90% non-private, preventing data leakage.	Privacy–performance trade-off.
Xu et al. (2021)	Compression cut FL comms by 70% with 1% accuracy drop.	Sync issues hurt convergence in unstable networks.

maximizes the braking action by 25% and reduced 15% of incorrect braking force calculations. while protecting the driver's privacy. Th overall litterature of this subsection is highlighted in the table 2.1.

2.2.2 Data poisoning and label-flipping attacks in federated learning

LF attacks have been one of the most prevalent forms of data poisoning that target the integrity of FL systems. In general, LF attacks manipulate the performance of a global model by maliciously altering the labels of the training data to decrease the accuracy and reliability of the global model. This section reviews the literature related to LF attacks: mechanisms, impacts, and mitigation strategies, with a focus on the effects in FL contexts.

2.2.2.1 Mechanisms of label-flipping attacks in federated learning

The increasing popularity of FL has led to the exploration of various attacks in this context, such as backdoor attacks Zhuang *et al.* (2024), gradient leakage attacks Yang *et al.* (2024), and membership inference attacks Zhu *et al.* (2024). In our work, we focus on data poisoning attacks against FL systems, mainly LF and feature perturbation (FP), which are critical areas of research.

Biggio *et al.* (2012b) provided one of the earliest categorizations of data poisoning attacks, including LF attacks. Their work highlighted the ease with which adversaries could compromise model integrity by altering data labels. The study showed how attackers might drastically affect the performance of centralized machine learning models by changing a tiny percentage of the labels on the training data just 1%–5% could lead to a 10%–20% increase in classification error. Although this study concentrated on conventional machine learning, its core principles have since been adapted and extended for application in decentralized FL systems. It was until 2020 where Fang, Cao, Jia & Gong (2020) expanded the knowledge of LF attacks to FL settings. The researchers demonstrated how malicious clients might alter the behavior of the global model by flipping labels in their local datasets. By measuring the effect of LF attacks, they showed that flipping 20% of labels on just 10% of participating devices could considerably impact model performance reaching a 35% degradation in global model accuracy on the CIFAR-10 dataset. Those results raised the alarm in the research community about the usage of FL and how simply it could be impacted if no security measures were implemented. From that point on, we saw

significant rise of research made regarding this subject, mainly on the effect of the attack on the performance and the proposed defense mechanism.

2.2.2.2 Impact of label-flipping attacks on federated learning performance

Rosenfeld, Winston, Ravikumar & Kolter (2020b) studied the degradation of performance of a classification task by flipping a small fraction of labels in the MNIST dataset using LF attacks, and found a slight increase ranging between 2%–3% in classification error by only flipping 1% of the labels. The study was repeated with Multiclass Logistic Regression, and a filipping intesity of 10% this revealed a classification error increase from 3% to 15% due to a random LF attack. It highlighted the facts that the decrease of performance is correlate with the scale of the LF attack, each additional malicious client will further decrease the performance. LF attacks have been widely applied in image processingPaudice, Muñoz-González & Lupu (2019), Tolpegin, Truex, Gursoy & Liu (2020a) extended to label-specific scenarios such as image recognition, where adversaries could adjust predictions based on predetermined patterns. the study was experiment on CIFAR-10 and a reduced version of ImageNet, it confirmed the effectiveness task-specific LF attacks in reducing performance accuracy by 15%-25%, more effectively than random LF of the proposed method.

Fung, Yoon & Beschastnikh (2020) conduct a reserach about sybil attack that involves an adversary controlling multiple devices, to inject melicous updates to the global model using LF attack. The evaluated standard aggregation methods like Federated Averaging (FedAvg) and robust algorithms such as Krum by studying the weights of the output layer. The results showed that those techniques were highly sensitive to the attack leading to a rapid decrease of performance reaching almost 40% even when robust aggregation methods were used. But these techniques also frequently penalize identical but good updates mistakenly, which causes the model's performance to significantly decline. More related to our own study Sun, Lee & Zhao (2022) used a real-world driving datasets to investigate the effects of LF in autonomous driving scenarios. According to their research, LF attacks have the potential to result in serious misclassifications reaching 30% increase in false positives, including mistaking a "stop" sign

for a "speed limit" when only 10% of participants were compromised, posing significant safety risks. This highlighted the necessity of robust defense mechanisms for FL in safety-critical applications.

2.2.3 Defense mechanisms against label-flipping attacks

In order to protect against poisoning attacks, a number of studies concentrate on evaluating certain updates.

Yin, Chen, Ramchandran & Bartlett (2021) introduced robust aggregation techniques, including Krum and coordinate-wise Median, designed to mitigate the impact of Byzantine and label-flipping attacks by filtering anomalous client updates. These methods rely on statistical outlier detection to identify and discard poisoned updates during the aggregation process. Experimental results demonstrated that, on the MNIST dataset (LeCun, Bottou, Bengio & Haffner (1998)), the proposed aggregation strategies reduced the accuracy degradation caused by label-flipping attacks from approximately 30% to 10%, even when up to 20% of participating clients were malicious.

Li, Zhang & Wang (2023a) presented a statistical approaches based Using a kernel density estimator, calculates how harmful each local update is in relation to its k-nearest neighbors. After that, it uses an asymptotic threshold to determine if updates are benign or poisoned. Not only is it difficult to choose a threshold of this kind. The approach was 90% accurate in detecting malicious clients in IID settings, with false positive rates of less than 5%. However this approach has not been validated with big DL models or non-IID data.

In order to identify the LF attack, Qayyum, Janjua & Qadir (2022a) suggests a method for discovering the correlation between the latent features of training data and updates. A secondary model was trained to identify these correlations, which were then utilized to determine whether updates were benign or harmful. Experiments were carried out using CIFAR-10 and ImageNet datasets. Detection accuracy and computing cost were the main measures. The detection accuracy was 85% for CIFAR-10 and 80% for ImageNet, with moderate processing expenses.

Table 2.2 Summary of Techniques and Results for Label-Flipping Attacks in Federated Learning

Study	**Key Findings**	**Limitations**
Biggio et al. (2012)	1%–5% LF increased classification error by 10%–20% in centralized ML models (SVM).	Focused on centralized systems; no exploration of FL settings.
Fang et al. (2020)	In FL systems, 20% flipped labels on 10% clients led to a 35% drop in global model accuracy on CIFAR-10.	Assumed IID data, limiting real-world applicability.
Bhagoji et al. (2019)	Adaptive LF attacks increased misclassification rates by 15%–25%, outperforming static LF.	Computationally complex, making large-scale attacks less feasible.
Rosenfeld et al. (2020)	Randomized smoothing limited classification error increases to 2%–15% even with 1%–10% poisoned labels.	Experiments limited to small-scale datasets (e.g., MNIST).
Chen et al. (2020)	5% LF reduced accuracy by 20%–25% on CIFAR-10 and 15%–20% on ImageNet.	Focused on fixed attack patterns; lacked adaptive attack exploration.
Sun et al. (2022)	30% false positive rate for stop-sign detection when 10% of participants were malicious.	Did not consider adaptive or colluding adversaries in AV contexts.
Yin et al. (2021)	Robust aggregation (e.g., Krum, Median) reduced accuracy drop from 30% to 10% under LF on MNIST.	Still vulnerable to colluding adversaries.
Li et al. (2023)	Kernel density estimation achieved 90% detection accuracy for malicious updates in IID settings.	Struggled with non-IID data and limited scalability.
Qayyum et al. (2022)	Latent feature analysis achieved 85% detection accuracy on CIFAR-10 and 80% on ImageNet.	Computationally expensive; relied on benign initial rounds.
Fung et al. (2020)	Sybil + LF caused a 40% accuracy drop on CIFAR-10 with 20% Sybil nodes.	Limited to small-scale datasets and static attack strategies.

However, the strategy imposes an additional cost on all parties to train another model that learns such relationship. Furthermore, it is unrealistic to believe that throughout the early training rounds, all peers will behave appropriately.

According to this literature review, FL has emerged as a potential strategy for enabling collaborative training in self-driving vehicles while maintaining data privacy. However, most current implementations are still vulnerable to data poisoning and, specifically, LF attacks. Defensive techniques proposed in the literature, such as aggregation-based filtering, reputation scoring, and gradient-similarity analysis, frequently make static assumptions about data distributions or use centralized trust models. These assumptions are not valid in true autonomous-vehicle scenarios, where participants use a variety of sensors, non-IID data, and intermittent connectivity. Furthermore, previous approaches usually address security at a particular stage of the FL process. Client-side anomaly detection ignores global behavior correlations, but exclusively server-side or aggregation-based defenses fail to detect localized anomalies that emerge early in the training rounds. Few approaches combine multiple detection views or build long-term trust with clients. As a result, FL systems remain vulnerable to covert or coordinated poisoning efforts that steadily degrade model performance without being detected.

Another significant problem is the absence of adaptive validation techniques capable of discriminating between legitimate data drift and hostile manipulation. Current strategies tend to penalize minor deviations, resulting in decreased accuracy and unwarranted participant exclusion. These inadequacies highlight the need for a more comprehensive approach that can:

- Detect irregular model changes at various stages of the federation process.
- Adapt to shifting data conditions and participant levels.
- Maintain the scalability and computing efficiency required for real-time autonomous vehicle applications.

This identified research analyzed in the table 2.2, need drives further inquiry into the underlying dynamics of adversarial behavior in FL, as well as the development of techniques capable of boosting robustness while preserving privacy and model performance.

CHAPTER 3

DEVELOPMENT OF THE FEDERATED LEARNING FRAMEWORK

This chapter describes the process utilized to create the FL environment for AV.

3.1 Experimental Design Overview

This section presents the overall experimental strategy adopted in this research. The objective is to design, evaluate, and secure a FL framework for AV environments under LF attacks.

The experimental methodology integrates both simulation-based and hardware-based implementations in order to ensure realistic validation of the proposed framework. Three key aspects are evaluated throughout the experimental process:

- The robustness of FL models under varying LF attack intensities;
- The effectiveness of attack modeling strategies;
- The performance of defense mechanisms in detecting and mitigating malicious clients.

Performance is assessed using detection accuracy, model robustness, and standard evaluation metrics such as accuracy, recall, and precision. These evaluation criteria serve as a baseline reference for the attack and defense analyses presented in the subsequent chapters.

3.2 Data and collection methods

Accurate and diverse data are critical to effectively training and evaluating FL models for AV. This work combines real-world and simulated data to provide comprehensive representation of various driving circumstances and sensor dynamics.

3.2.1 Data description

The data used in this study include both empirical data collected directly from the SunFounder PiCar during real-world operations and synthetic datasets, especially the CARLA Simulator Dataset and the Audi Autonomous Driving Dataset (A2D2)Audi. These datasets total

approximately 40,000 labeled images, with a training set of 32,000 and a testing set of 8,000. The dataset selection criteria prioritized capturing diverse and typical AV operational scenarios, including lighting conditions, obstacle types, traffic signals, and road designs.

3.2.2 Data collection methods

The data collection technique includes using a variety of specialized hardware and software solutions. Specifically, the SunFounder PiCar was used to collect environmental data using its inbuilt camera and sensors, which included ultrasonic and line-following modules. The NVIDIA Jetson AGX Orin Developer Kit provided critical computational capacity for data processing and running complicated model training algorithms. Furthermore, Google's Edge TPU was utilized to optimize real-time inference processes, resulting in significantly lower latency and higher response accuracy. In addition to these components, the CARLA Simulator was crucial for generating high-quality synthetic datasets, which were rigorously designed to replicate realistic driving circumstances and supplement empirical data.

3.3 System component

The key components of this study are the SunFounder PiCar, the NVIDIA Jetson AGX Orin Developer Kit, and Google's Edge TPU as shown in figure 3.2. These components work together to form a solid foundation for simulating, executing, and evaluating FL models in settings similar to those used in AV.

3.3.1 SunFounder PiCar

The SunFounder PiCar is a modular robotic vehicle platform powered by a Raspberry Pi, making it suitable for educational and research purposes. In this study, the PiCar serves as an autonomous vehicle model, simulating real-world scenarios in a controlled experimental environment. Its modular architecture enables significant customization and integration of additional sensors and modules, increasing its adaptability to complex tasks. The PiCar is equipped with various

Table 3.1 SunFounder PiCar-V Smart Robot Video Car V2.0 Kit Specifications

Category	Details
Compatibility	Raspberry Pi 3 Model B/B+, Raspberry Pi 2 Model B, Raspberry Pi 3A+.
Programming Language	Python.
Camera	Adjustable-angle USB camera module for video streaming and image capture.
Connectivity	Wi-Fi connectivity via the Raspberry Pi.
Sensors	Line-following module and ultrasonic sensor for obstacle detection.
Motor Control	Dual DC motors with a dedicated motor driver board.
Power Supply	Requires two 18650 rechargeable lithium batteries.
Chassis Material	Durable acrylic chassis with pre-drilled component mounting holes.
Features	• Video streaming over Wi-Fi. • Real-time control via Python interface.

sensors, including ultrasonic modules and line-following components, as well as a camera for computer vision studies. Its compatibility with Python and packages such as OpenCV enables extensive programming capabilities. Furthermore, the PiCar's ability to connect to Wi-Fi networks enables seamless participation in FL environments.

3.3.2 NVIDIA Jetson AGX Orin Developer Kit

The NVIDIA Jetson AGX Orin Developer Kit is a cutting-edge AI platform designed to meet demanding machine learning and edge AI requirements. As a key component of this research, this device provides the processing capacity required for training and inference activities inside the FL framework. Its architecture comprises an 8-core ARM Cortex-A78AE CPU and a 2048-core NVIDIA Ampere GPU with Tensor Cores, which provides exceptional parallel processing capabilities. With AI capability of up to 275 trillion operations per second (TOPS),

the Jetson AGX Orin excels at effectively carrying out sophisticated model training and inference procedures. The system also features 32 GB of LPDDR5 memory and supports high-capacity storage, ensuring smooth performance during resource-intensive workloads. Its connectivity options, including Ethernet, PCIe, and USB ports, allow for smooth interaction with other hardware components. Furthermore, this developer kit is optimized for edge AI applications, making it ideal for real-time FL situations in AV systems.

Table 3.2 Specifications of the NVIDIA Jetson AGX Orin Developer Kit

Category	**Specifications**
GPU	NVIDIA Ampere architecture GPU with 2048 CUDA cores and 64 Tensor Cores. Supports FP32, FP16, INT8, and other precision formats.
CPU	12-core ARM Cortex-A78AE v8.2 64-bit CPU.
Memory	64GB LPDDR5 with up to 204.8 GB/s memory bandwidth.
Storage	32GB eMMC 5.1 onboard. Expandable via M.2 NVMe SSD or microSD card slot.
AI Performance	Up to 275 TOPS (Tera Operations Per Second) for deep learning and AI workloads.
Networking	10/100/1000/2500 Mbps Ethernet (2.5G Ethernet). Wi-Fi and Bluetooth available via external USB adapters.
Connectivity	USB 3.2, USB 2.0, PCIe Gen4, HDMI 2.1, and M.2 interfaces. Multiple GPIO, I2C, I2S, UART, and SPI pins for external device interfacing.
Power Options	Configurable power modes (15W, 30W, and 50W).
Size	Compact design, measuring 110 x 110 mm.
Software	Runs NVIDIA JetPack SDK, which includes L4T (Linux for Tegra), CUDA, cuDNN, TensorRT, and support for DeepStream, Isaac Sim, and ROS.

Category	Specifications
Development Tools	TensorRT for optimizing AI models, DeepStream SDK for video analytics, and support for popular AI frameworks like TensorFlow, PyTorch, and ONNX.

3.3.3 Google's Edge TPU

Google's Edge TPU is a dedicated hardware accelerator that performs ML tasks at the edge. In this study, the Edge TPU is used to enhance the inference capabilities of FL models, ensuring efficient processing in real-time applications. Its architecture emphasizes power efficiency, making it ideal for use in low-power environments like AV systems. The device can execute up to four TOPS designed specifically for TensorFlow Lite models, allowing for fast and precise calculations. Its small form factor enables easy integration onto robotic platforms such as the SunFounder PiCar. By processing tasks locally, the Edge TPU significantly reduces latency, eliminating the need for cloud-based computing. Additionally, the hardware features robust security mechanisms, such as encryption, to protect model deployment.

Table 3.3 Specifications of the Google Edge TPU

Category	Specifications
Performance	Capable of delivering up to 4 TOPS (Tera Operations Per Second), optimized for TensorFlow Lite models.
Power Efficiency	Designed for low-power operation with consumption typically under 2 watts.
Form Factor	Compact design enabling seamless integration into embedded systems and IoT devices.
Connectivity	Supports USB, PCIe, and M.2 interfaces for versatile integration options.

Category	Specifications
Security	Includes built-in hardware encryption for secure model execution.
Compatibility	Optimized for TensorFlow Lite frameworks; supports pre-trained models for edge inference.
Latency	Provides ultra-low-latency inference, reducing dependency on cloud computing.
Temperature Range	Operates efficiently in a wide temperature range, suitable for embedded systems in diverse environments.
Development Tools	Compatible with Edge TPU Compiler and TensorFlow Lite Model Maker for deploying and optimizing models.

3.4 Integration of components

In this study, the SunFounder PiCar serves as the primary physical platform for mimicking autonomous vehicle behavior. The NVIDIA Jetson AGX Orin Developer Kit serves as the central processing unit that manages training and coordination tasks inside the FL framework. The Google Edge TPU accelerates on-device inference processes, enabling the system to make real-time decisions. The integration of these components creates a comprehensive and scalable testing environment that simulates real-world autonomous driving settings, giving a solid foundation for assessing the proposed FL models.

3.5 Study implementation

This section outlines the methodology for preparing and training the models used in our AV experiments, followed by the implementation of the FL framework.

3.5.1 Autonomous vehicles model selection and training

For our experiments, we used three SunFounder PiCars to simulate the behavior of a real autonomous vehicle, we trained them using a CNN (Convolutional Neural Network) and Reinforcement Learning (RL) model explained in 3.1.

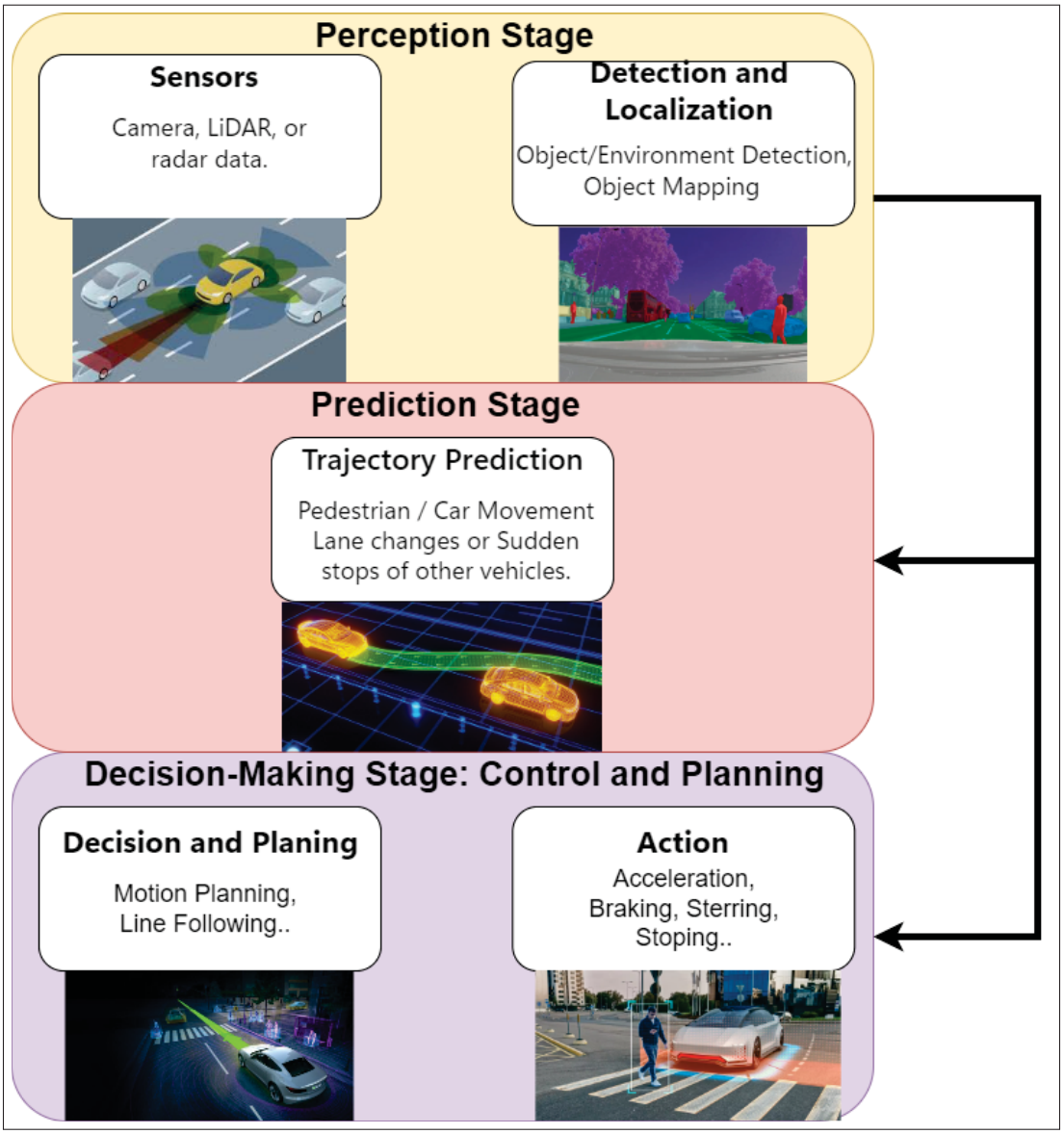


Figure 3.1 Overview of the Connected and Autonomous Vehicle (CAV) training stages, illustrating the sequential data acquisition, model training, and validation process in the proposed framework.

The perception and decision-making modules of the AV as shown in figure 3.1 were jointly developed using CNN for feature extraction and RL for policy optimization. The training process comprised the following:

- **Input Preprocessing:** The training pipeline integrates multiple data sources to support both supervised and reinforcement learning objectives. Supervised learning is conducted using the A2D2 dataset, while reinforcement learning experiments are performed within the CARLA simulation environment. In addition, real-world data are collected using a custom-built PiCar platform to complement simulated observations. Images captured by the PiCar's onboard camera are preprocessed to a fixed spatial resolution and normalized prior to training. To further improve model generalization and robustness, data augmentation techniques—including random rotations, horizontal flipping, and color jittering—are applied to increase input diversity.
- **CNN Feature Extraction:** The CNN serves as the AV perception module, trained for semantic segmentation and object identification tasks.

  A ResNet-based architecture was employed, wherein residual blocks ensure efficient gradient flow:

$$y = F(x, \{W_i\}) + x \tag{3.1}$$

  The CNN was trained to minimize cross-entropy loss:

$$L_{CE} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_{i,c} \log(\hat{y}_{i,c}) \tag{3.2}$$

  Here, $y_{i,c}$ represents the ground-truth label, and $\hat{y}_{i,c}$ is the predicted probability for class c.
- **RL Policy Optimization:** The RL agent was trained using a Markov Decision Process (MDP), where the state s_t, action a_t, and reward r_t transitions follow:

$$P(s_{t+1} \mid s_t, a_t) \tag{3.3}$$

The agent's policy $\pi(a_t|s_t)$ was optimized with the Proximal Policy Optimization (PPO) algorithm:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta) A_t,\ \text{clip}\big(r_t(\theta), 1-\epsilon, 1+\epsilon\big) A_t \right) \right] \tag{3.4}$$

The reward function encouraged efficient and collision-free navigation:

$$r_t = -\alpha \cdot \text{collision} + \beta \cdot \text{path efficiency} \tag{3.5}$$

The previous algorithm 3.1 consists of two main components:

3.5.1.1 Convolutional Neural Network Training

The training of the CNN is conducted through supervised learning methodologies using the A2D2 and PiCar datasets. The process involves several key stages. Initially, the CNN model is configured with a ResNet-based architecture to leverage its robust feature extraction capabilities. The training proceeds iteratively across multiple epochs, with the data divided into manageable batches. Input images undergo preprocessing before being passed through the model to generate predictions. The model's accuracy is assessed by computing the cross-entropy loss, which quantifies the discrepancy between predicted outputs and ground truth labels. Through backpropagation, the CNN's weights are adjusted to minimize this loss. After each epoch, the model's performance is evaluated on a validation dataset to monitor its progress and ensure generalization. This comprehensive process aims to optimize the CNN for precise feature extraction and nuanced semantic understanding of input data.

3.5.1.2 Reinforcement Learning Training

In the RL training context, the policy network facilitates decision-making and control within a simulated CARLA environment. The training procedure begins with initializing an RL agent equipped with a neural network-based policy. The agent operates across iterative episodes within the simulation. During each episode, the trained CNN extracts perception features from the environment, which serve as inputs to the policy network. The agent selects actions based on

Algorithm 3.1 Training CNN and RL Models with A2D2, CARLA, and PiCar Datasets

Input: A2D2, CARLA, and PiCar datasets; ResNet-based CNN architecture; PPO algorithm parameters
Output: Trained CNN and RL models

Training CNN:
Initialize CNN with ResNet backbone;
for $e = 1$ *to* E_{CNN} **do**
  for *each batch* $B = \{(x_i, y_i)\}_{i=1}^{N}$ *from the training set* **do**
    Normalize and preprocess inputs x_i;
    Compute predictions $\hat{y}_i$ using CNN;
    Compute cross-entropy loss:

$$L_{CE} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_{i,c} \log(\hat{y}_{i,c})$$

    Update CNN weights via backpropagation;
  end for
  Evaluate the CNN on the validation set;
end for

Training RL:
Initialize RL agent with policy $\pi_\theta(a_t|s_t)$;
Initialize experience replay buffer $\mathcal{D}$;
for $e = 1$ *to* E_{RL} **do**
  Reset CARLA environment and obtain initial state s_0;
  for *each timestep t in the episode* **do**
    Extract perception features f_t from CNN using s_t;
    Select action a_t based on policy $\pi_\theta(a_t|f_t)$;
    Execute action a_t in CARLA environment;
    Observe reward r_t and next state s_{t+1};
    Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$;
  end for
  Sample minibatches of transitions from $\mathcal{D}$;
  Optimize policy π_θ using PPO loss:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon)A_t) \right]$$

  Update policy parameters θ;
end for

the current state and policy, and its performance is assessed through a reward signal that reflects factors such as collision avoidance and operational efficiency. These interactions comprising the state, chosen action, received reward, and subsequent state are stored in a replay buffer for future use.

Subsequent training involves sampling mini-batches from the replay buffer to refine the policy network using the Proximal Policy Optimization (PPO) algorithm. This optimization employs a clipped objective function to maintain stable updates, ensuring that the policy improves consistently without significant divergence. Together, these steps facilitate the development of a robust decision-making framework that integrates learned perception features with adaptive control strategies.

This process ensures that the agent learns to navigate efficiently while avoiding obstacles in various simulated scenarios.

3.6 Federated learning framework for autonomous vehicle simulation

We designed and implemented a FL framework to simulate collaborative learning across multiple AV . This framework utilizes three PiCars as edge devices that simulate AV and an NVIDIA Jetson AGX Orin Developer Kit as the central server. The FL framework enables distributed training and model aggregation, ensuring data privacy while leveraging the collective learning of all vehicles.

We implement our FL framework for malicious vehicle detection using $N = 3$ participants, a central aggregator, and $k = 3$ local updates per round. The training data are distributed according to an independent and identically distributed (i.i.d.) scheme,ensuring that each participant receives an equal and randomly distributed portion of the entire dataset. The testing dataset is used only for model evaluation and is not included in any participant's training data, each participant receives a distinct subset of the training data, D_i. Given that both CNN models achieve convergence in fewer than 200 training rounds, we configure our FL experiments to run for a total of $R = 200$ rounds in total.

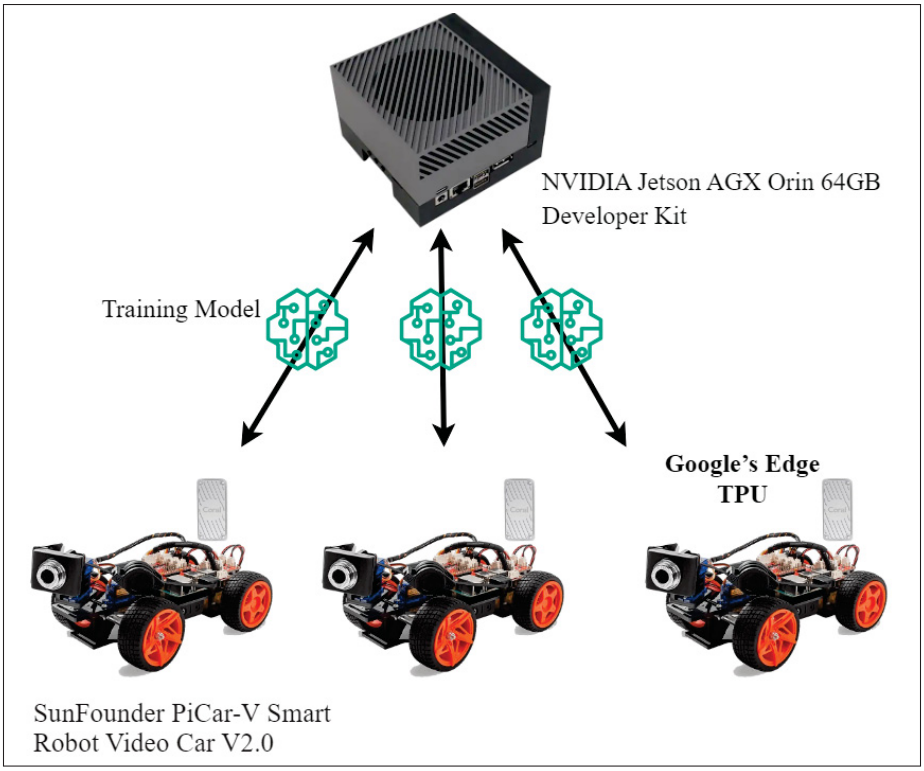


Figure 3.2 Integration of the NVIDIA Jetson AGX Orin Developer Kit with the SunFounder PiCar-V platform for real-world validation of the proposed Federated Learning framework in autonomous-vehicle scenarios.

Initially, we trained our FL models without incorporating adversarial settings to establish a baseline for performance comparison. This training process utilized the Federated Averaging (FedAvg) algorithm to aggregate updates from participants and build a robust global model. Subsequently, we evaluated the resulting global model on the test dataset, focusing on samples with high prediction confidence. Confidence levels were computed using softmax probabilities, and samples were included in a complementary dataset if the predicted class matched the true label and the probability exceeded a predefined threshold.

Effective collaboration and communication between participants and the central aggregation server are critical for the success of the FL framework. The training process concludes with the global model M, represented by its parameters θ_R after $R = 200$ rounds. The evaluation

of M is performed on the test dataset D_{test}, which satisfies the condition $D_{\text{test}} \cap D_i = \emptyset$ for all participant datasets D_i.

In subsequent sections, we provide a comprehensive analysis of LF attacks within the FL framework. This investigation highlights potential vulnerabilities in FL systems and explores the impact of adversarial behaviors on the performance and integrity of the global model.

3.6.1 Framework design

The proposed framework integrates edge devices and a central server to implement an FL system for autonomous driving tasks, ensuring efficient collaboration and secure communication.

Each PiCar serves as an edge device equipped with a camera and an onboard Raspberry Pi to facilitate local processing and model training. These devices collect diverse data types, including sensor readings, images, and control commands, which are used to train local CNN models for perception tasks. This local approach reduces the need for raw data transmission to the central server, enhancing both data privacy and processing efficiency.

The central server, implemented using the NVIDIA Jetson AGX Orin Developer Kit, serves as the aggregation point for the FL framework. It receives model updates from all participating PiCars and aggregates these updates using the Federated Averaging (FedAvg) algorithm. The aggregation process is represented mathematically as:

$$w_t = \frac{1}{N} \sum_{i=1}^{N} w_t^i \tag{3.6}$$

where w_t^i denotes the model parameters from the i-th PiCar at training round t, and N represents the total number of PiCars. After aggregation, the central server distributes the updated global model back to all PiCars, enabling iterative improvement of the shared model.

To ensure the integrity and confidentiality of the communication between the PiCars and the central server, a secure communication protocol is employed. This protocol safeguards the

exchange of model updates and minimizes potential vulnerabilities, such as data breaches or adversarial interference. The integration of secure communication with FL mechanisms enhances the robustness and reliability of the proposed framework. the algorithm 3.2 shows the implementation steps.

Algorithm 3.2 Federated Learning Framework for Autonomous Vehicles

Input: N PiCars, Central Server, Initial Global Model w_0, Federated Rounds T, Local Epochs E, Learning Rate η
Output: Final Global Model w_T

Initialize global model w_0 on the central server;
for $t = 1$ *to* T **do**
  Broadcast global model w_t to all N PiCars;
  for *each PiCar* $i \in \{1, \ldots, N\}$ ***in parallel*** **do**
    Receive w_t from the server;
    Train local model w_t^i using local data for E epochs:;
    $w_t^i \leftarrow w_t - \eta \nabla L(w_t, \text{local data})$;
    Send updated model w_t^i back to the server;
  end for
  Aggregate models on the server using Federated Averaging:;

$$w_{t+1} \leftarrow \frac{1}{N} \sum_{i=1}^{N} w_t^i$$

end for
Return final global model w_T;

3.7 Baseline Experimental Results

This section aims at presenting the results of our study of the framework. The results are obtained using the previous discussed simulation and methodology.

3.7.1 Performance of CNN-RL Model

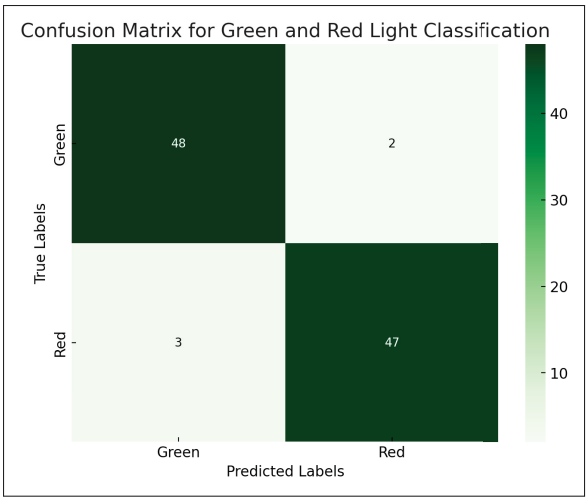
The CNN and RL models were deployed and evaluated on the SunFounder PiCar platform to determine their real-world performance in autonomous navigation and traffic sign recognition tasks. The CNN model's performance was evaluated based on its ability to classify traffic signs, whereas the RL model was assessed based on navigation efficiency and obstacle avoidance.

The CNN model achieved a traffic sign classification accuracy of 94.2% during real-world testing, as shown in figures 3.3a and 3.3b, which was slightly lower than the simulated results, most likely due to variations in illumination and background noise. Precision and recall were measured at 0.939 and 0.945, respectively, with an F1 score of 0.942. Misclassifications were primarily observed in signs with low contrast or partially obstructed views. With an average real-time processing delay of 42 ms per frame, the system was deemed sufficiently responsive for autonomous operations.

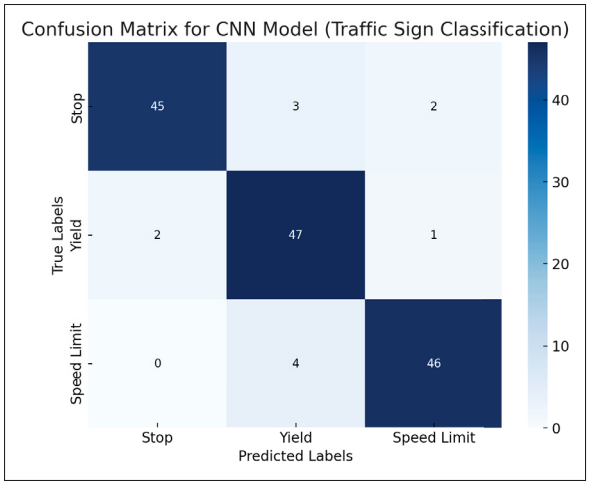
On the PiCar, the RL model demonstrated effective navigation capabilities, achieving an average cumulative reward of 1365 across 50 real-world trials. With an average episode duration of 85 timesteps, the success rate in reaching target locations without collisions was 91.8% highlighted in figure 3.4. Failures were attributed to unforeseen challenges or inconsistent environments that were not present during simulated training.

The deployment results demonstrate that CNN and RL models can be deployed in practical autonomous systems. The challenges posed by real-world environments were reflected in the statistical comparisons between simulation and real-world performance using paired t-tests, which showed substantial disparities in accuracy ($p - value < 0.05$) for both models, as mentioned in the figures.

Confusion matrices for the CNN model provide classification accuracy across multiple traffic sign categories, and trajectory plots for the RL model show navigation paths and collision points, These visualizations demonstrate the models' performance. These visualizations provide valuable insights into areas requiring improvement.



a) Confusion matrix illustrating CNN performance in distinguishing green and red traffic lights.



b) Confusion matrix illustrating CNN classification accuracy across three traffic sign categories (Stop, Yield, Speed Limit).

Figure 3.3 Comparison of CNN model performance on two distinct visual recognition tasks: (a) traffic light classification and (b) traffic sign recognition.

3.7.2 Performance within federated learning framework

Both models were retrained and evaluated using the collaborative learning approach of the FL framework to assess their performance. With precision and recall values of 0.929 and 0.935,

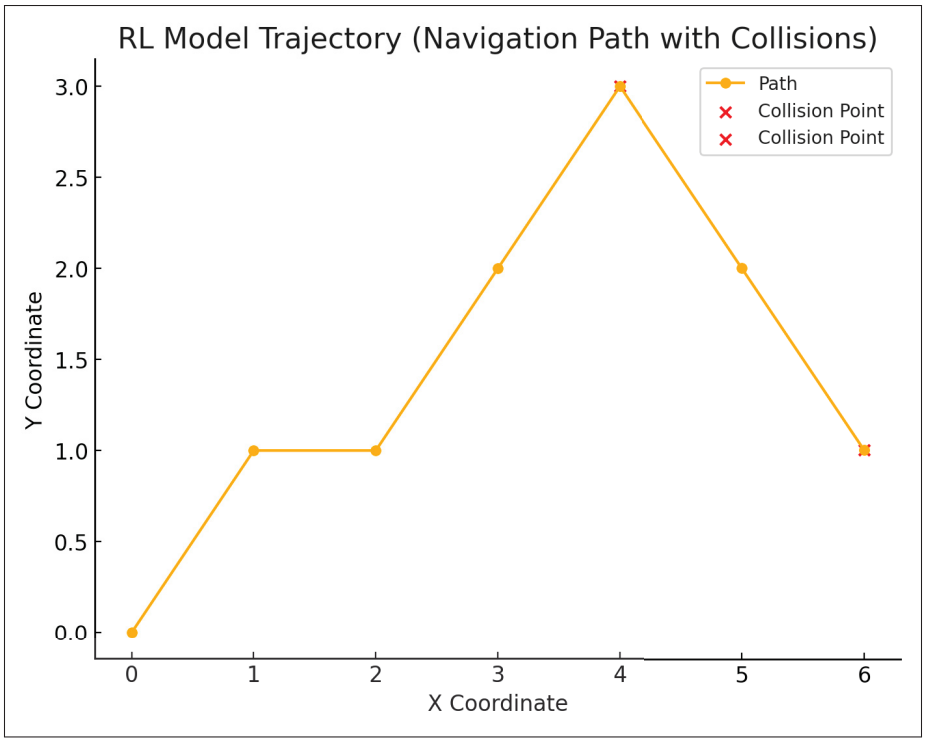


Figure 3.4 RL model trajectory plot showing the navigation path within the autonomous driving environment. Red markers indicate collision points encountered during exploration and policy optimization.

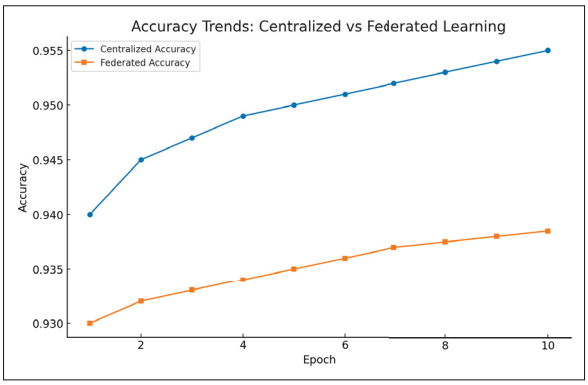
respectively, the CNN model's classification accuracy accuracy declined to 93.7% compared to centralized training. F1's score decreased to 0.932. The FL system improved scalability and privacy by significantly reducing the need for data transfer, although performance slightly declined.

In real-world trials, the RL model achieved a success rate of 90.4%. However, the average cumulative reward decreased slightly to 1335. The average duration of episodes was 83 timesteps, which indicates a slight decrease in navigation efficiency. Nonetheless, safer and more effective deployment in distributed environments was ensured by the FL framework's ability to reduce communication overhead among nodes.

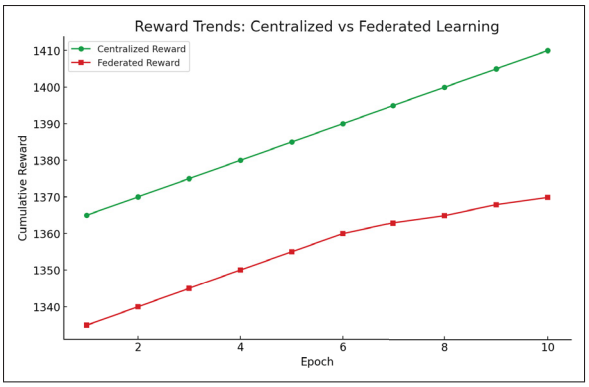
Using the FL framework resulted in slight performance decreases for both models when compared to centralized training. Statistical tests using ANOVA revealed that these differences were not

statistically significant (F = 2.34, $p > 0.05$), suggesting that the trade-offs in performance are acceptable given the benefits of scalability and data privacy. The reduced data transfer also alleviated bandwidth limitations, making the approach more suitable for real-world applications.

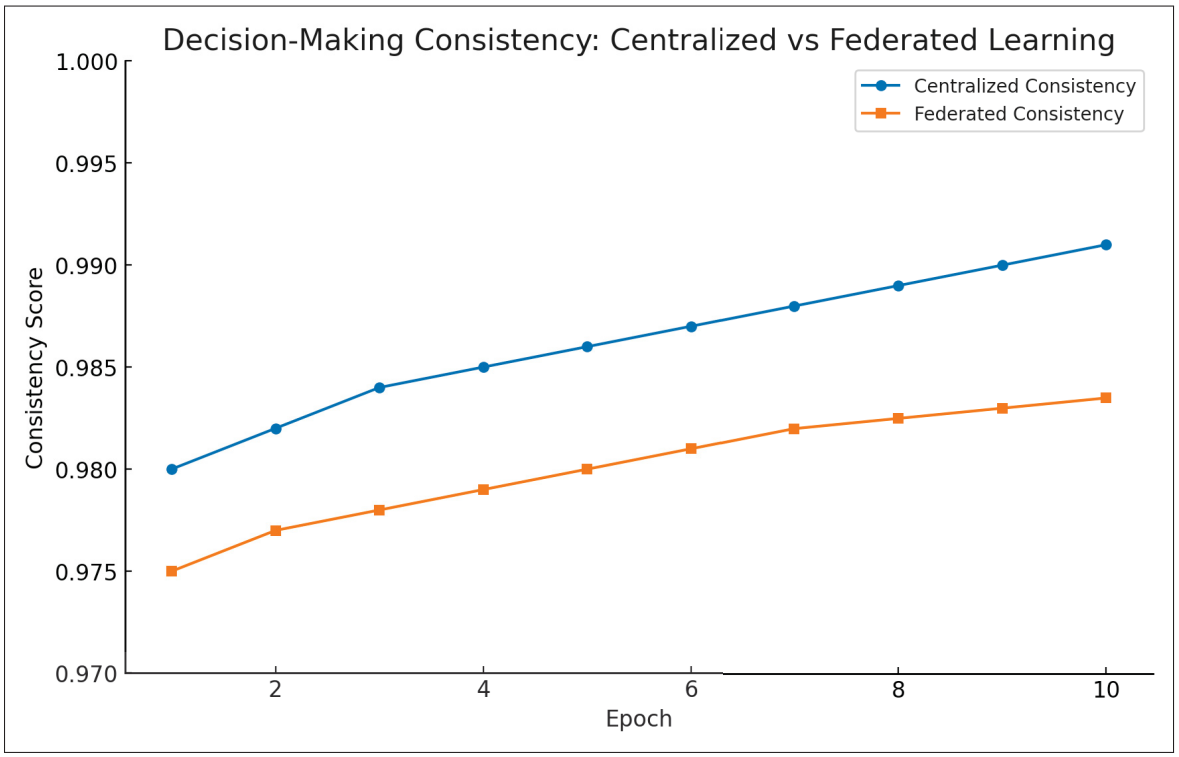
Results from the FL framework are visualized in figures 3.5 using comparative graphs of accuracy and reward patterns, highlighting the trade-offs and benefits of FL. Line plots of decision-making consistency measures for both models show consistent behavior across nodes, demonstrating the framework's effectiveness in maintaining performance while enhancing security.



a) Accuracy across epochs (centralized vs federated).



b) Cumulative reward across epochs.



c) Decision-making consistency across nodes.

Figure 3.5 Comparison of centralized and federated learning metrics: (a) accuracy trends, (b) cumulative rewards, and (c) decision-making consistency.

CHAPTER 4

LABEL-FLIPPING ATTACK PROCESS MODELING

This chapter describes the LF attacks deployed against the FL framework. The simulation replicates malicious clients who actively change the labels in their local datasets before submitting model updates to the global aggregation process.

The objective is to reduce the global model's accuracy and stability by introducing corrupted gradients during the training rounds.

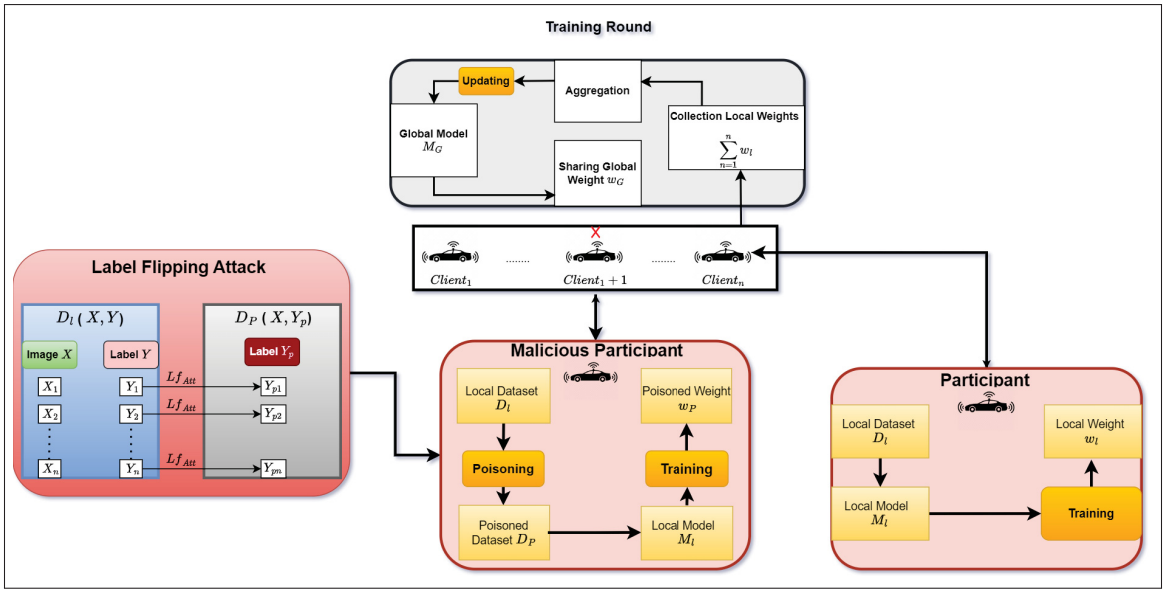


Figure 4.1 Federated Learning data-poisoning workflow illustrating the label-flipping (LF) attack mechanism in autonomous vehicle training. The figure highlights how compromised clients manipulate local labels before model aggregation, leading to global performance degradation.

As demonstrated in figure 4.1, the attack begins when selected malicious clients flip a portion of their dataset labels ($Y \rightarrow Y_p$) using the LF function Lf_{Att}. These poisoned datasets D_P are used to train local models which generate corrupted weights w_P, which are then sent to the central server for aggregation. This process produces a biased global model M_G, thereby reducing its accuracy and convergence performance.

4.1 Label flipping attack simulation

We utilize a LF attack to conduct targeted data poisoning in FL. Given a source class C_{Source} and a target class C_{target} from C, each malicious participant P_i modifies its dataset D_i as follows:

For all instances in D_i whose class is C_{Source}, the class label is flipped to C_{target}. We denote this attack as $C_{\text{Source}} \rightarrow C_{\text{target}}$. For instance, in CIFAR-10 image classification, a red traffic light may be altered to a green light by malicious participants, increasing the likelihood that the final global model misclassifies traffic signals during testing.

LF is a well-known threat in centralized ML and is equally applicable in FL due to its minimal requirements and high stealthiness. Unlike other poisoning attacks, an adversary does not need knowledge of the CNN architecture, loss function L, or the global distribution of D. Its efficiency in time and computation makes it particularly dangerous in FL environments where edge devices are common. Moreover, it is easy to execute, requiring no modifications to FL software or to participant-side configurations.

To simulate LF in an FL system with $P = 3$, where one participant (P) is malicious, we proceed as follows:
At the start of each experiment, we randomly designate $N \times m\%$ of the participants as malicious, while the remaining participants are honest. The malicious participants inject flipped labels, and each experiment is repeated 10 times, reporting the average results. The algorithm in 4.1 demonstrate these steps

Algorithm 4.1 Label Flipping Attack Simulation in Federated Learning

Input: Dataset D, Source class C_{Source}, Target class C_{Target}, Number of participants P, Malicious ratio m, Number of experiments N

Output: Flipped datasets for malicious participants

```
for i = 1 to N do
    Split D among P participants;
    Randomly select N × m% of participants as malicious;
    for each participant P_i ∈ P do
        if P_i is malicious then
            for each (X, y) ∈ D_i do
                if y == C_Source then
                    Set y = C_Target;
                    ;                                   /* Flip label */
                end if
            end for
        end if
    end for
end for
return Flipped datasets for malicious participants;
```

4.2 Attack evaluation metrics

We employ several evaluation metrics for this purpose.

Global Model Accuracy (M_{acc}): The global model accuracy is the percentage of instances $x \in D_{\text{test}}$ where the global model M with final parameters θ_R predicts $M_{\theta_R}(x) = c_i$ and c_i is indeed the true class label of x.

Class Recall (c_{recall_i}): The percentage

$$\frac{T_{P_i}}{T_{P_i} + F_{N_i}} \cdot 100\% \tag{4.1}$$

represents the class recall for any class $c_i \in C$. Where as F_{N_i} is the number of examples $x \in D_{\text{test}}$ where $M_{\theta_R}(x) \neq c_i$ and the true class label of x is c_i. The number of instances $x \in D_{\text{test}}$ is T_{P_i}, where $M_{\theta_R}(x) = c_i$ and c_i is the true class label of x.

4.3 Results and Analysis

The impact of LF attacks was assessed to determine how a single malicious participant could reduce global model performance in the FL framework. Metrics as source class recall and global model accuracy were evaluated at different levels of malicious participant availability (α).

First, we studied the effect of a single malicious participant on our FL framework, comparing it to the results achieved when all participants were non-malicious. We found that even one malicious participant can significantly degrade global model performance. Source class recall can be reduced by over 25% when this adversary is consistently well-represented in the participant pool. This impact on source class recall is most significant at a high availability level of $\alpha = 0.9$. Consequently, the effect diminishes as availability decreases, indicating that lower losses can be achieved at lower values: $\alpha = 0.7$, $\alpha = 0.6$, or $\alpha = 0.5$. Thus, to maximize the attack's effectiveness it is beneficial for the malicious participant to remain as available as possible-especially in the later training rounds. To further demonstrate this availability effect, we report source class recall per round for $\alpha = 0.7$ and $\alpha = 0.9$. A higher availability of the malicious participant results in a noticeable degradation in source class recall, along with a lower recall value $\alpha = 0.9$ compared to $\alpha = 0.7$. The probabilistic selection of participants can be considered a primary reason for recall variability across rounds. A round with fewer malicious participants tends to increase source recall, while a higher number reduces it.Each experimental condition was run three times, and the outcomes were averaged to eliminate round-to-round variability. As our results indicate, even a single malicious participant can significantly reduce

global model performance, with source class recall losses of over 20% possible under high availability. Indeed, high availability produces a significant negative effect, while decreased availability yields considerably better results. Importantly, for values of k significantly larger than $N \times m\%$, increasing availability (α) becomes less effective in producing meaningful impacts in individual training rounds.

The findings demonstrate the vulnerability of FL systems to a single malicious actor. The high availability of the adversarial client increases the attack's efficacy, especially in later training rounds. In contrast, reduced availability diminishes the impact, revealing a trade-off between adversarial representation and global model robustness.

4.3.1 Effect on source class recall

The source class recall (ρ) was significantly impacted by the presence of a malicious participant. The results were modeled as follows:

$$\rho = \rho_0 - \beta \cdot \alpha \cdot t \tag{4.2}$$

where ρ represents the source class recall, ρ_0 is the initial recall without attacks, α is the availability level of the malicious participant, β is the adversarial impact coefficient, and t is the training round.

At high availability levels ($\alpha = 0.9$), source class recall dropped by over 17% compared to non-adversarial scenarios. This reduction was exacerbated during later training rounds due to the cumulative effects of adversarial data poisoning. Conversely, lower availability levels ($\alpha = 0.7$ or $\alpha = 0.5$) mitigated this impact, with smaller recall losses.

The variation in recall over rounds reflects the probabilistic nature of participant selection, with adversary-dominated rounds resulting in substantial performance degradation. Averaging data

from three experimental runs reduced round-to-round variability, confirming the reproducibility of the observed patterns.

The graph below 4.2 illustrates the source class recall over 20 training rounds for $\alpha = 0.9$. The results clearly demonstrate the amplified degradation at higher availability levels.

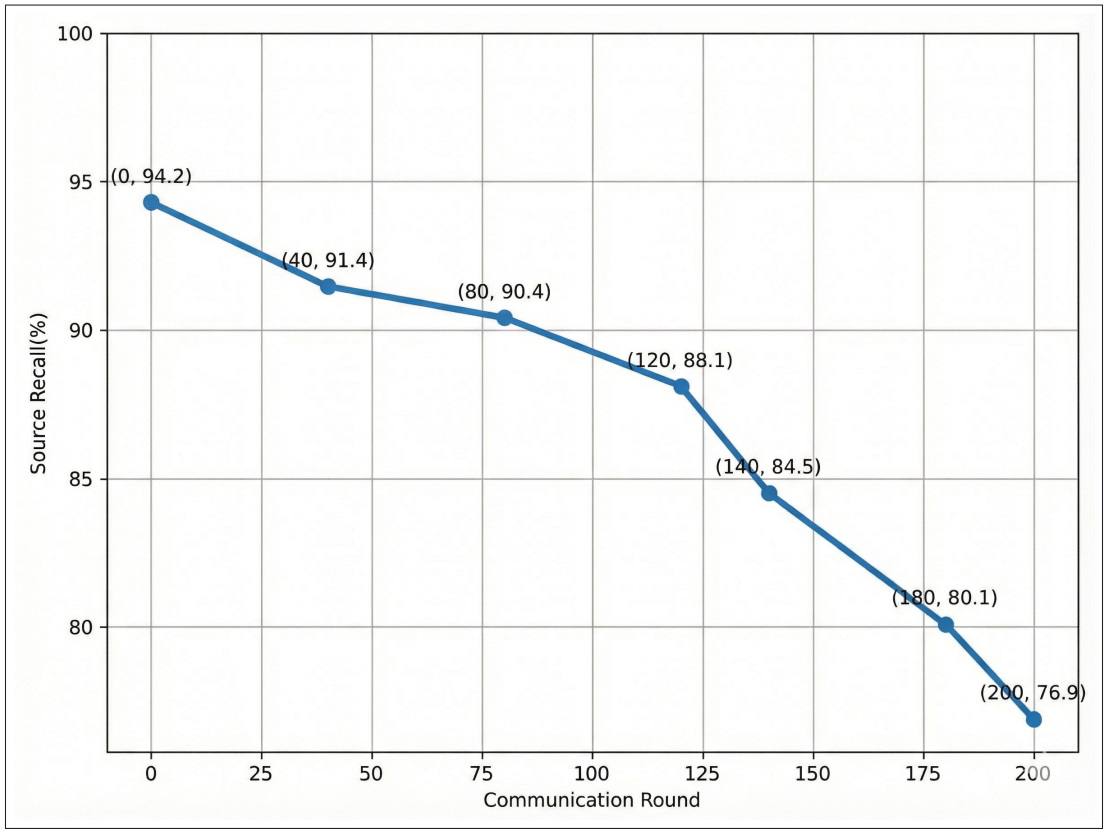


Figure 4.2 Evolution of the source class recall by round for $\alpha = 0.9$. The plot highlights the effect of adversarial availability on class-specific recall under the proposed defense mechanism.

4.3.2 Effect on global model accuracy

Regarding accuracy, we observe that each training round involving the malicious participant affected model accuracy by a drop of around 0.1% leading to an overall loss of 20% when the model finished training.

The global model's accuracy (A) followed a degradation pattern:

$$A = A_0 - \gamma \cdot \alpha \cdot t \tag{4.3}$$

where A_0 is the initial accuracy without attacks, γ is the accuracy degradation coefficient, and α and t are as defined earlier.

Over 20 training rounds, the global accuracy decreased by approximately 0.1% per round, culminating in an overall loss of 20% under high-availability conditions.

The global model's accuracy trends are visualized below. The consistent accuracy drop highlights the adversarial impact on training.

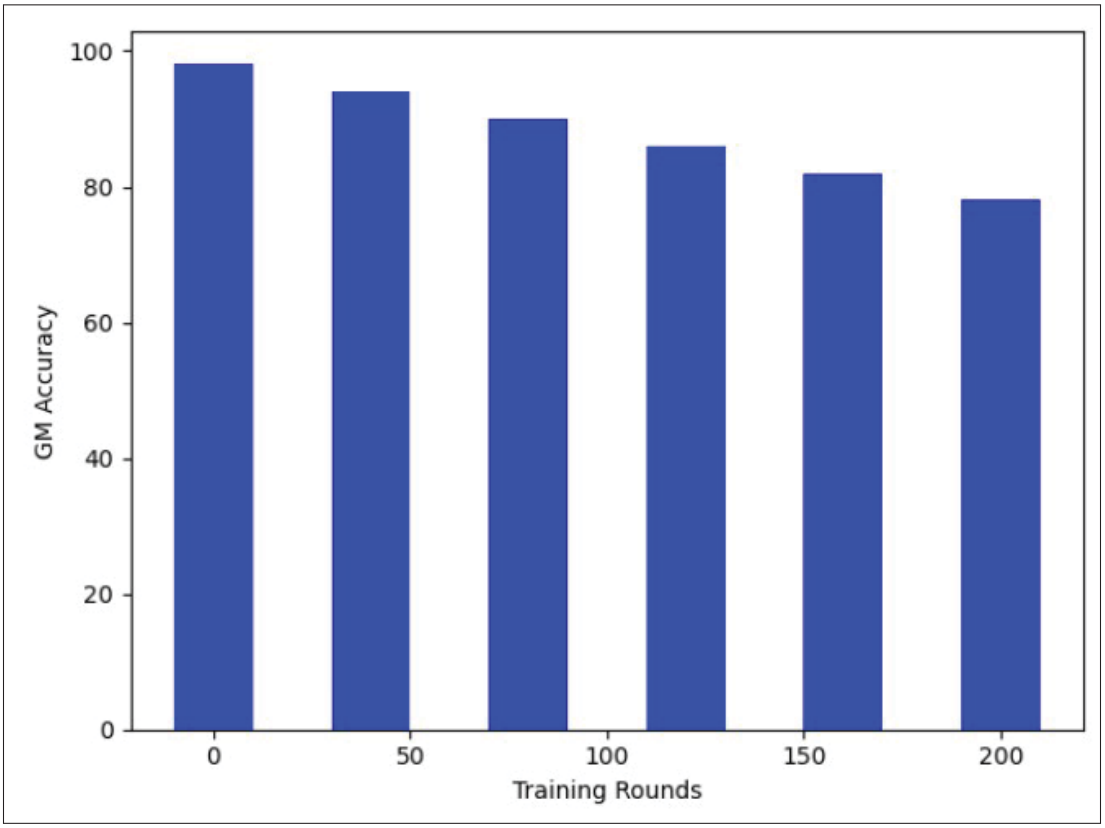


Figure 4.3 Evaluation of the global model accuracy during federated learning with the presence of a malicious participant. The plot highlights the degradation in performance caused by poisoned updates before applying the proposed FALCON defense.

4.3.3 Analysis

The study confirms that even a single malicious participant can significantly degrade FL performance. High availability amplifies this impact, emphasizing the need for robust defense mechanisms. The mathematical models and empirical results underscore the critical role of participant selection and availability in mitigating adversarial threats.

Table 4.1 Performance metrics under label-flipping attack scenarios

Metric	**No Attack**	α **= 0.9**	α **= 0.7**
Source Class Recall (%)	94.2	76.9	81.5
Global Model Accuracy (%)	93.7	73.9	82.3
Accuracy Drop per Round (%)	0	0.1	0.07

CHAPTER 5

DEFENSE MECHANISMS AGAINST LABEL-FLIPPING ATTACKS

This chapter discusses the proposed defense mechanism for detecting and mitigating the effects of LF attacks within the FL framework. The defense mechanism is designed to preserve model integrity, ensure trust among participating clients, and maintain consistent performance even in adversarial environments.

Our approach builds upon the foundation established in the previous sections by integrating anomaly detection, statistical analysis, and collaborative validation techniques into the FL workflow. Specifically, it focuses on identifying malicious contributors whose data manipulations could distort the global model updates and degrade its overall accuracy.

5.1 Proposed defense mechanism FALCON

The aim of our defense approach against LF attacks is to create a detection agent as shown in figure 5.5 consisting of three security layers between the aggregation layer which includes our server S, where we train our initial model and use it for distributing and updating the global model by aggregating locally learned models from the vehicles. The ultimate goal is to reach the optimal weights by optimizing the loss function $\mathcal{L}(W)$.

$$W_{\text{Op}} \leftarrow \arg\min \mathcal{L}(W) \tag{5.1}$$

The other layer is the training layer that consists of three P vehicles, which, after obtaining a global model from the central server, jointly train their local models. Each P trains a local model using its local dataset $D_p \in D$.

To counteract LF attacks in FL, we introduce FALCON, a multi-layered anomaly detection framework that systematically identifies, analyzes, and mitigates adversarial manipulations. FALCON integrates Federated Anomaly Detection (FAD), Principal Component Analysis (PCA), and Multi-Class Support Vector Machines (MCSVM). The primary purpose of this defense

mechanism is to detect, analyze, and neutralize LF attacks, while maintaining the integrity and privacy of the FL process.

FALCON operates across three interconnected layers. The first layer, Client-Level Local Detection, operates at the individual client level, where PCA is applied to reduce the dimensionality of local model updates, followed by MCSVM, which classifies updates as either benign or adversarial. To quantify the degree of anomaly, an Outlier Score is computed at this stage to assess deviations from expected model updates. Once local anomaly detection is completed, the second layer, Peer-to-Peer Anomaly Voting, is activated. Here, each client shares its Outlier Score with its neighboring clients, enabling collaborative validation and reducing the risk of false positives. The final layer, Server-Level Graph-Based Anomaly Detection, uses a GNN at the central server to examine patterns of adversarial activity across multiple rounds, detecting coordinated adversarial methods—in our case, LF attacks. A complete summary of our defense pipeline is shown in Figure 5.5 and its algorithm at 5.1

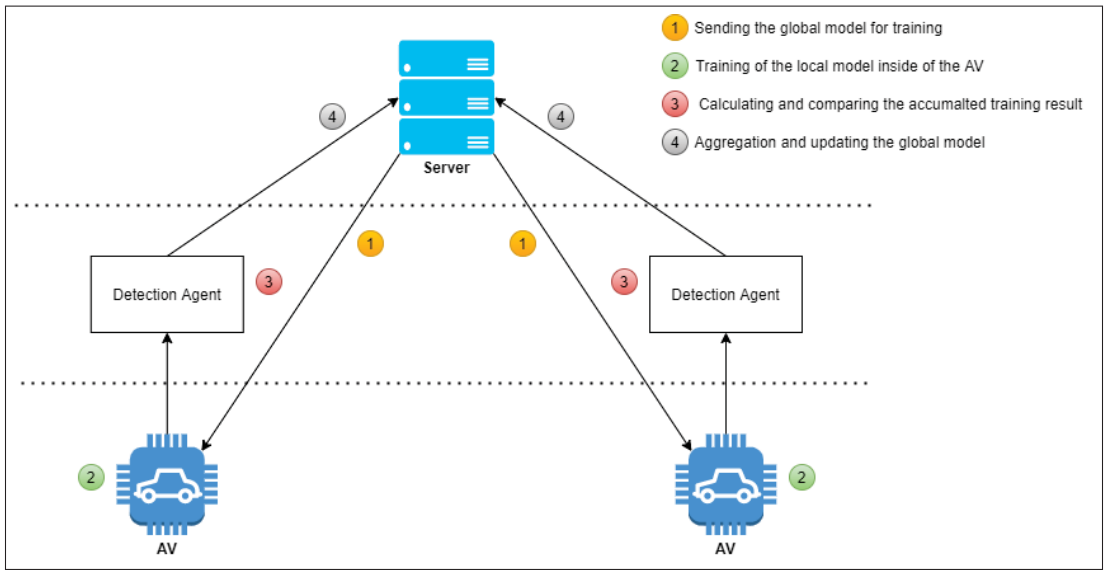


Figure 5.1 Overview of the proposed Federated Learning system design, illustrating the interaction between local client training, aggregation server, and communication layers within the autonomous vehicle framework.

5.1.1 First layer: Client-level local detection (PCA and MCSVM)

Before transmitting the model update to the server, anomalies are detected at the client level. This involves two major steps: dimensionality reduction and anomaly classification.

5.1.1.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is employed as a preprocessing tool to reduce the dimensionality of participant updates while retaining the most significant features. The dimensionality reduction achieved by PCA is particularly critical in handling high-dimensional gradient vectors generated by participants during the FL process. Direct analysis of these vectors is computationally expensive, and PCA transforms them into a lower-dimensional space by projecting them onto a set of principal components that explain the maximum variance. Mathematically, this transformation is represented as:

$$\mathbf{W}_i^{\text{PCA}} = \mathbf{E}_k^T \left(\mathbf{W}_i - \bar{\mathbf{W}} \right) \tag{5.2}$$

where $\mathbf{W}_i$ represents the update vector of the i-th participant, $\bar{\mathbf{W}}$ is the mean vector of all updates, and $\mathbf{E}_k$ contains the top k eigenvectors of the covariance matrix Σ. The covariance matrix is defined as:

$$\Sigma = \frac{1}{n} \sum_{i=1}^{n} \left(\mathbf{W}_i - \bar{\mathbf{W}} \right) \left(\mathbf{W}_i - \bar{\mathbf{W}} \right)^T \tag{5.3}$$

where n is the number of participants. This transformation filters out noise and less significant variations in the updates, allowing the system to focus on meaningful patterns and discrepancies that may indicate adversarial activity. Furthermore, PCA provides the additional benefit of enabling visualization in two or three dimensions, facilitating the identification of clusters corresponding to honest and malicious updates.

5.1.1.2 Multi-Class Support Vector Machines (MCSVM)

Following PCA-based dimensionality reduction, MCSVM is utilized to classify participant updates and detect malicious behavior. MCSVM adopts a one-versus-all strategy, constructing binary classifiers for each class to distinguish it from all others. Each classifier determines a hyperplane that separates its target class with the largest possible margin. The decision function for the m-th classifier is given by:

$$f_m(\mathbf{x}) = \mathbf{w}_m^T \mathbf{x} + b_m \tag{5.4}$$

where $\mathbf{w}_m$ is the weight vector, b_m is the bias term, and $f_m(\mathbf{x})$ is the decision function for input $\mathbf{x}$.

MCSVM also computes an outlier score for each participant's update, quantifying the likelihood of malicious behavior. This score is determined as:

$$\text{Outlier\_Score}_{t+1} = \frac{1}{|D_C|} \sum_{i=1}^{|D_C|} (y_i \neq \hat{y}_i) \tag{5.5}$$

where y_i is the true label, $\hat{y}_i$ is the predicted label, and D_C represents the dataset used for classification. Updates with outlier scores exceeding a predefined threshold are flagged as malicious, enabling their exclusion from the aggregation process.

the overall workflow of the first defense layer is illustrated in Figure 5.2.

5.1.2 Second layer: Federated Anomaly Detection (FAD) via peer-to-peer anomaly voting for distributed validation

FALCON's peer-to-peer anomaly validation mechanism is designed to enhance detection robustness while reducing false positives. It is based on FAD, that introduces a shared and distributed consensus phase among participants to validate client-level anomaly decisions before updates reach the server. Instead of relying solely on individual client assessments, each client

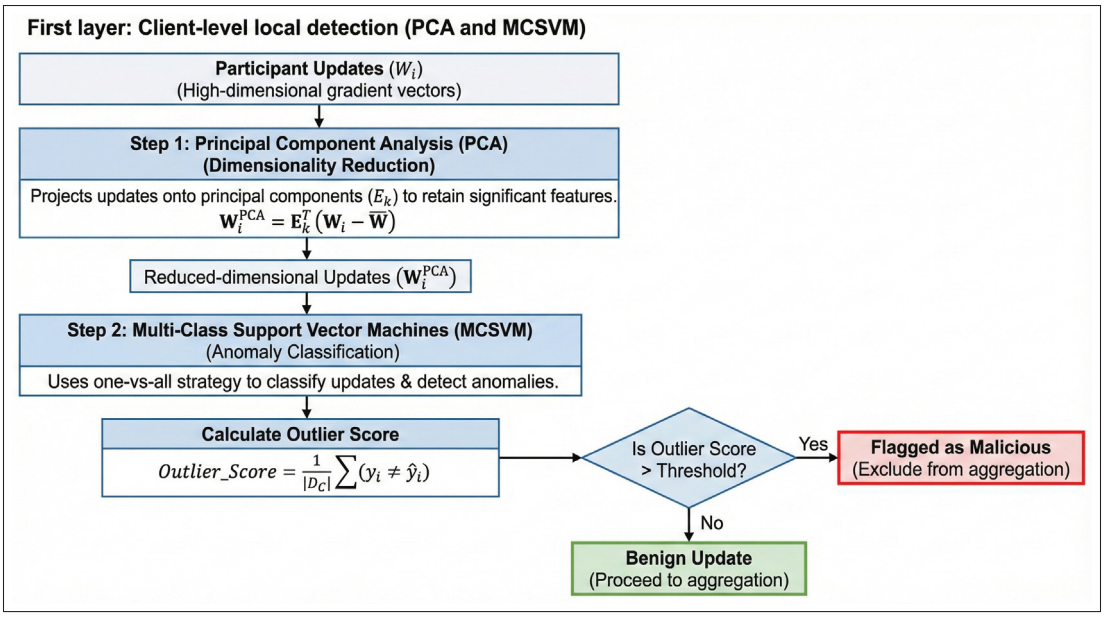


Figure 5.2 **First-layer client-level detection.** Each participant computes a local update W_i (high-dimensional gradient vector), applies PCA to obtain a reduced representation, and uses MCSVM to classify the update. An outlier score is then compared to a threshold: updates exceeding the threshold are flagged as malicious and excluded, while benign updates proceed to aggregation.

shares its computed Outlier Score with the K-nearest neighboring clients. This collaborative evaluation enables clients to validate anomaly classifications collectively.

Each peer independently evaluates the suspicious update and produces a continuous outlier score.:

$$\bar{s}_k = \frac{1}{K}\sum_{i=1}^{K} s_{k\to i} \tag{5.6}$$

τ is a predefined anomaly threshold. If the consensus anomaly score exceeds this threshold, the update is considered adversarial and is excluded before being submitted to the server. This approach ensures that isolated misclassifications do not result in unnecessary exclusions, and that only updates identified by many clients are deemed malicious.

By incorporating the Outlier Score into peer validation, this layer enhances the FL system's resilience against LF attacks, reducing the likelihood of individual misclassifications while maintaining detection robustness.

However, if the majority of AV are corrupted, malicious participants can manipulate the peer validation process, leading to collusive corruption, misleading Outlier Scores, and suppression of detection. Collusive corruption occurs when malicious AV approve each other's adversarial updates, making it difficult to filter out poisoned contributions which is more likely when the majority of AV are corrupted. Misleading Outlier Scores arise when malicious AV intentionally vote against legitimate updates, increasing false positives and causing benign updates to be discarded. Suppression of detection happens when adversarial AV collectively reduce their Outlier Scores, making it harder for the system to flag them before updates reach the server.

To prevent this, FALCON incorporates several countermeasures that enhance validation and mitigate collusion risks. Historical trust score tracking assigns evolving trust scores to AV across successive FL rounds, identifying persistent adversarial behavior even when peer voting initially fails to detect it. Rather than employing simple majority voting, weighted voting adjusts votes based on historical trust scores, preventing malicious AV from dominating the validation process. Randomized peer selection mitigates reliance on fixed K-nearest neighbor voting by randomly selecting AV for each validation round, reducing the risk of persistent collusion. Finally, if peer voting approves an excessive number of updates, the server can employ GNN-based anomaly detection ensuring long-term FL security, assuring long-term FL security.

These additional countermeasures ensure that FALCON remains robust even when a large number of AV are compromised, thereby strengthening FL security beyond standard peer validation.

5.1.3 Historical Trust Scores and Peer-to-Peer Validation

To improve robustness against transient noise and coordinated adversarial behavior, FALCON introduces a historical trust mechanism combined with peer-to-peer (P2P) anomaly validation.

This design enables the system to distinguish persistent malicious behavior from benign update drift caused by environmental or sensor variations.

5.1.3.1 Local Outlier Score Computation

At each federated round t, client k trains a local model and produces an update $\Delta W_k^t = W_k^t - W^{t-1}$. After PCA projection and MCSVM classification (Layer 1 and Layer 2), an outlier score $s_k^t \in [0, 1]$ is computed based on the misclassification rate on a clean validation dataset $\mathcal{D}_{val}$:

$$s_k^t = \frac{1}{|\mathcal{D}_{val}|} \sum_{(x,y)\in\mathcal{D}_{val}} \mathbb{I}\left(f_k^t(x) \neq y\right), \tag{5.7}$$

where $f_k^t(\cdot)$ denotes the locally trained model and $\mathbb{I}(\cdot)$ is the indicator function. Updates with $s_k^t > \tau$ (with $\tau = 0.3$) are flagged as suspicious and forwarded to the peer validation stage.

5.1.3.2 Peer-to-Peer Anomaly Evaluation

Let $\mathcal{P}_k$ denote the set of peers participating in the current federated round. Each peer $j \in \mathcal{P}_k$ evaluates the suspicious update ΔW_k^t using the same validation procedure, yielding a peer-assigned outlier score $s_{k\to j}^t$. The peer-to-peer anomaly matrix $A^t \in \mathbb{R}^{N\times N}$ is constructed as:

$$A_{k,j}^t = s_{k\to j}^t, \tag{5.8}$$

where diagonal entries $A_{k,k}^t$ correspond to self-evaluation scores. High diagonal values indicate intrinsic inconsistency with the clean reference data, even when evaluated by the client itself. The consensus outlier score $\bar{s}_k^t$ is computed as the mean peer evaluation:

$$\bar{s}_k^t = \frac{1}{|\mathcal{P}_k|} \sum_{j \in \mathcal{P}_k} A_{k,j}^t. \tag{5.9}$$

This aggregation mitigates false positives caused by localized noise (e.g., fog, lighting conditions) while preserving sensitivity to adversarial manipulation.

5.1.3.3 Historical Trust Score Update

To capture long-term behavior, FALCON maintains a historical trust score $T_k^t \in [0, 1]$ for each client, updated using an exponential moving average:

$$T_k^t = \lambda T_k^{t-1} + (1 - \lambda)(1 - \bar{s}_k^t), \tag{5.10}$$

where $\lambda \in [0, 1)$ controls memory retention. In our experiments, $\lambda = 0.9$ prioritizes long-term consistency while remaining responsive to recent behavior.

Low trust scores indicate persistent deviations across rounds, whereas transient spikes caused by benign conditions are smoothed over time.

5.1.3.4 Decision Rule and Integration with Server-Level GNN

A client update is considered malicious if:

$$T_k^t < \delta, \tag{5.11}$$

where δ is a predefined trust threshold. Flagged updates are either down-weighted or excluded from aggregation and passed to the server-level graph neural network (Layer 3).

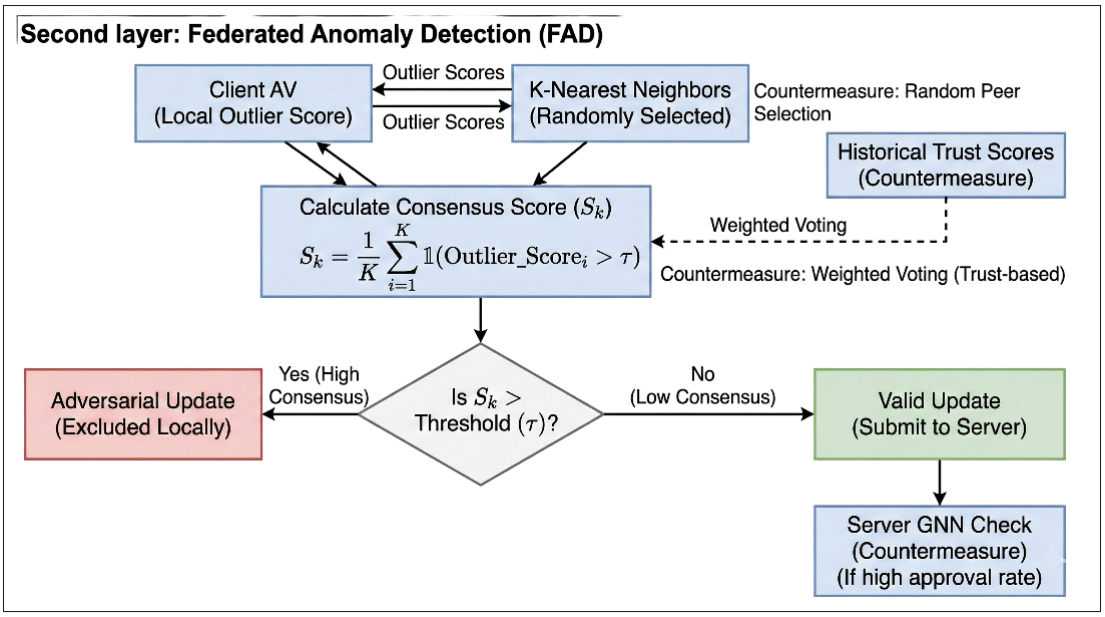


Figure 5.3 **Second-layer federated anomaly detection (FAD).** Each client computes a local outlier score and shares it with randomly selected peers (countermeasure against collusion). Peers contribute to a consensus score S_k, potentially using trust-weighted voting based on historical reliability. If S_k exceeds a threshold, the update is rejected; otherwise it is accepted and forwarded to the server for downstream verification.

The historical trust scores serve as node attributes in the GNN, enabling detection of coordinated adversaries whose behavior may not be apparent in individual rounds.

To provide an intuitive understanding of the collaborative anomaly validation process, the workflow of the federated anomaly detection (FAD) layer is illustrated in Figure 5.3

5.1.4 Third layer: Server-level graph-based anomaly detection

On the server, FALCON employs a GNN to evaluate client interactions and detect adversarial behaviors associated with LF attacks. Each client is represented as a node in the network, with edges representing the similarity of updates across clients based on historical model contributions. The similarity between updates from clients k and j is calculated as follows:

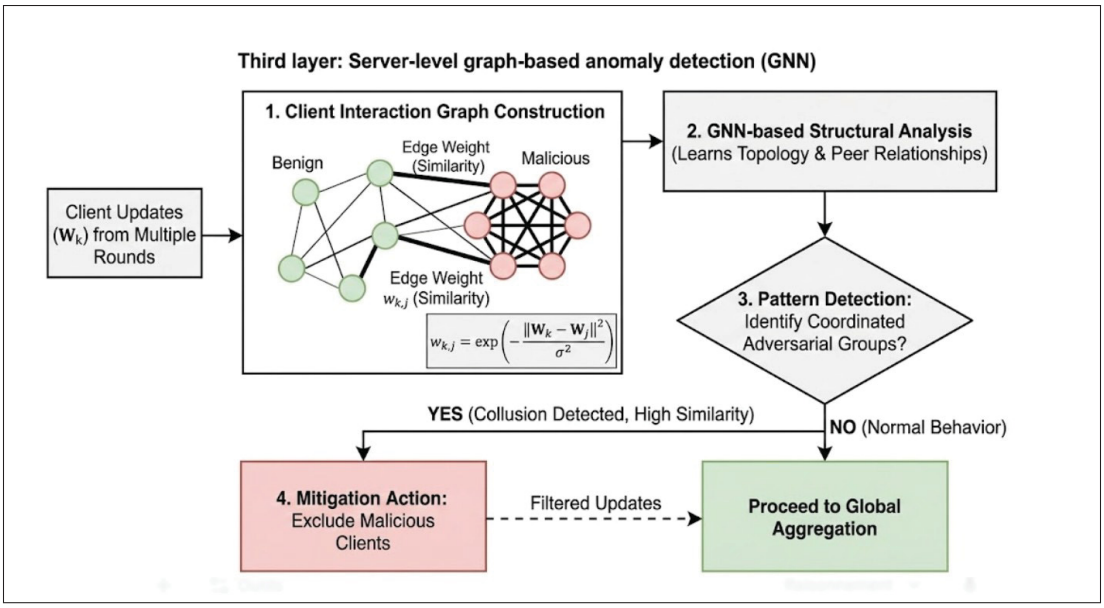


Figure 5.4 **Third-layer server-level detection using GNNs.** The server aggregates client updates across multiple rounds to build a client interaction graph, where edge weights reflect update similarity. A GNN performs structural pattern analysis to identify coordinated adversarial behavior (e.g., collusion). If malicious patterns are detected, the server excludes the corresponding clients/updates; otherwise, updates proceed to global aggregation.

$$w_{k,j} = \exp\left(-\frac{\|W_k^t - W_j^t\|^2}{\sigma^2}\right) \tag{5.12}$$

σ^2 represents the variance of update differences. The server leverages this graph representation to monitor consistency in updates across multiple rounds. In LF attack scenarios, malicious clients tend to introduce systematic misclassifications, causing their updates to differ from those of honest participants. GNN-based structural analysis allows the server to monitor peer relationships and identify coordinated adversarial groups that submit highly similar malicious updates. This enables FALCON to detect patterns of adversarial collusion that may not be apparent in individual updates.

Unlike conventional distance-based anomaly detection methods, GNN-based modeling enables long-term tracking of adversarial updates, learns complex relationships by analyzing the structure

of the update graph, making it particularly effective at distinguishing maliciously manipulated updates from benign update variations in model learning.

When a malicious client is identified, its contributions are either excluded from the aggregation process or flagged for further analysis to ensure that LF attacks do not compromise the integrity of the global model. To capture persistent and coordinated adversarial behavior that may evade client-level and peer-based detection, the server-level graph-based anomaly detection mechanism is summarized in Figure 5.4.

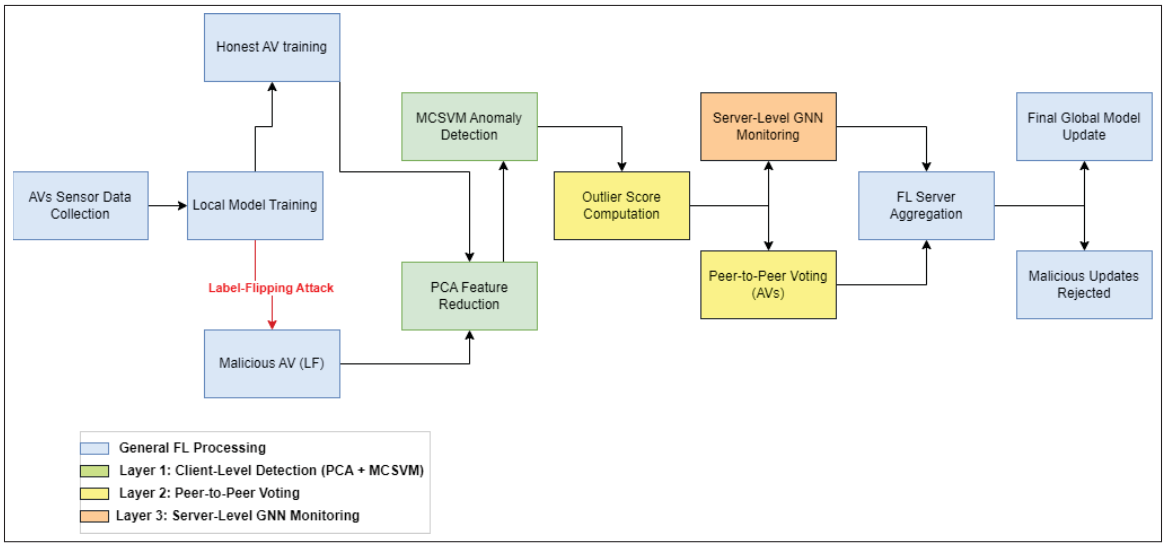


Figure 5.5 FALCON multi-layered defense mechanism against label-flipping attacks in Federated Learning for autonomous vehicles. The framework integrates client-side anomaly detection (PCA + MCSVM), peer-to-peer validation (FAD), and server-side monitoring using Graph Neural Networks (GNN) to ensure robustness and trust in the collaborative training process.

5.1.5 Evaluation indicators

Throughout this thesis, we employ several metrics to evaluate both the security and utility provided by the models under scrutiny, which we subject to experimental testing. These metrics are defined as follows:

- **Source Class Recall**: This metric measures the proportion of correct positive predictions relative to all actual positive instances in the dataset. In the event of label tampering by a

Algorithm 5.1 Federated Learning with FALCON

Input: AV nodes P, participants K, rounds T, clean dataset D_C, adversary set $\mathcal{A}$, MCSVM M_s, GNN detector GNN, initial model $\mathbf{W}_0$
Output: Final global model $\mathbf{W}_T$

for $t = 0, \ldots, T-1$ **do**
  Select random subset $S \subseteq P$ of size K;
  Broadcast current model $\mathbf{W}_t$ to all $k \in S$;
  for *each client* $k \in S$ `/* executed in parallel */` **do**
    if $k \in \mathcal{A}$ **then**
      Poison local data D_k via label flipping;
    end if
    Local update: $\mathbf{W}_{t+1}^{(k)} \leftarrow \text{ClientUpdate}(k, \mathbf{W}_t)$;
    ; `/* Client-level detection (PCA + MCSVM) */`
    Reduce dimensionality of $\mathbf{W}_{t+1}^{(k)}$ using PCA;
    Predict anomaly label $y_k \leftarrow M_s(\mathbf{W}_{t+1}^{(k)})$;
    Compute outlier score $\text{OS}_k \leftarrow \frac{1}{|D_C|} \sum_{i=1}^{|D_C|} \mathbb{1}(y_i \neq \hat{y}_i)$;
    if $\text{OS}_k > \tau$ **then**
      mark update as **suspicious** and share OS_k with peers;
    end if
  end for
  ; `/* Peer-to-peer validation (FAD layer) */`
  for *each* $k \in S$ **do**
    Receive neighbor scores; compute consensus $S_k \leftarrow \frac{1}{K} \sum_{i=1}^{K} \mathbb{1}(\text{OS}_i > \tau)$;
    if $S_k > \tau$ **then**
      flag k as malicious and discard $\mathbf{W}_{t+1}^{(k)}$;
    end if
  end for
  ; `/* Server-level GNN-based detection */`
  Build update-similarity graph $G = (V, E)$ from $\{\mathbf{W}_{t+1}^{(k)}\}_{k \in S}$;
  Run GNN to score persistent adversaries; remove top-O offenders;
  ; `/* Secure aggregation of validated clients */`
  Aggregate remaining updates to obtain $\mathbf{W}_{t+1}$;
end for
return $\mathbf{W}_T$;

malicious user, this metric will decrease, as fewer (or none) correct positive predictions will be made for the specific class C_{target} by the attacker.

- **Precision**: Precision quantifies the proportion of correctly identified positive instances among all instances predicted as positive by the model. This metric evaluates the model's ability to avoid false positives, which is critical when distinguishing maliciously altered data. Precision is defined as:

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}} \tag{5.13}$$

  where:

  - True Positives (TP): The number of correctly identified instances of the target class.
  - False Positives (FP): The number of instances incorrectly predicted as belonging to the target class.

  High precision ensures that the model makes reliable positive predictions, minimizing the inclusion of unrelated or adversarial data in the target class.

- **F1 Score**: Provides a balanced measure that considers both precision and recall, particularly useful when dealing with imbalanced datasets or scenarios with varying numbers of true positives, false positives, and false negatives. It is calculated as the harmonic mean of precision and recall:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.14}$$

  The F1 score emphasizes the importance of achieving a balance between precision and recall. A high F1 score indicates that the model is both precise in its predictions and capable of identifying a significant proportion of positive instances.

- **Global Model Accuracy (GM Accuracy)**: Evaluates the overall effectiveness of the model across all classes, providing a holistic view of its performance. This metric is calculated as the proportion of correctly classified instances, including both positive and negative examples, out of the total number of instances:

$$\text{Accuracy} = \frac{\text{True Positives (TP)} + \text{True Negatives (TN)}}{\text{Total Instances}} \tag{5.15}$$

  where:

- True Positives (TP): Correctly predicted positive instances.
- True Negatives (TN): Correctly predicted negative instances.
- Total Instances: The total number of examples in the dataset.

High accuracy indicates that the model performs well across all classes, reflecting its robustness and reliability in diverse scenarios.

- **Attack Success Rate** : Quantifies the proportion of samples from the source class that are misclassified into the attacker's target class due to poisoning.

$$\text{ASR} = \frac{N_{C_{\text{source}} \to C_{\text{target}}}}{N_{C_{\text{source}}}} \times 100\% \tag{5.16}$$

where:

- $N_{C_{\text{source}} \to C_{\text{target}}}$: number of instances originally labeled as C_{source} but predicted as C_{target},
- $N_{C_{\text{source}}}$: total number of instances in C_{source}.

A lower ASR indicates stronger resistance to LF attacks.

5.2 Experimental Evaluation

To assess the impact of our proposed FALCON defense mechanism, we apply the multi-layered detection approach, incorporating Outlier Score-based Peer Validation and GNN-based anomaly detection. Our results indicate that FALCON significantly mitigates the attack's effectiveness, preserving both source class recall and global model accuracy.

Source Class Recall Improvement: Without defense, recall drops by 25% or more due to adversarial misclassification. With FALCON, which integrates FAD, PCA, and MCSVM, recall degradation is reduced to less than 5%, even at high adversarial availability ($\alpha = 0.9$). The peer-to-peer validation step ensures that Outlier Scores are verified collaboratively, reducing the likelihood of false positives.

Model Accuracy Preservation: With FALCON in place, accuracy remains stable throughout training, as malicious updates are identified and excluded before aggregation as shown in figure 4.3. Unlike the baseline FL model, where accuracy degrades by 20%, FALCON prevents any

significant accuracy loss. The integration of GNN-based detection at the server ensures that adversarial participants who consistently introduce label-flipped updates are blacklisted over multiple rounds.

5.2.1 Multi-layers work flow

An ablation study was conducted to evaluate each defense component's contribution to the proposed architecture. This analysis isolates the impact of three key layers—Principal Component Analysis (PCA), Multi-Class Support Vector Machine (MCSVM), and Federated Anomaly Detection (FAD)—to understand how each module improves detection accuracy and model stability when facing LF attacks in FL.

The study employs a sequential approach, beginning with PCA-only detection, then introducing MCSVM classification, and finally incorporating FAD with peer-to-peer anomaly validation. To evaluate detection reliability under increasing adversarial pressure, each configuration was examined with different (α) values (0.5, 0.7, and 0.9).

PCA Projection of Update Vectors Table 5.1 and figure 5.6a illustrate how attack intensity affects update distribution in FL. Legitimate updates (blue) remain clustered, while high-intensity adversarial updates (red) are more scattered, deviating significantly from expected patterns. PCA alone achieves 78% accuracy in distinguishing adversarial updates, with 80% of high-intensity attacks falling outside the main cluster. The separation validates PCA's role in highlighting anomalies, which can then be classified using MCSVM.

MCSVM Decision Boundaries After PCA Projection: As shown in Figure 5.6b, the decision boundaries of MCSVM classify updates after PCA transformation based on attack intensity. MCSVM achieves 89.2% classification accuracy, correctly identifying 92% of legitimate updates. However, 24% of high-intensity adversarial updates are misclassified due to boundary overlap, indicating that some malicious updates attempt to mimic legitimate behavior. This result suggests that integrating FAD enhances classification by mitigating subtle adversarial drift.

Table 5.1 PCA Detection Performance at Different Availability Levels

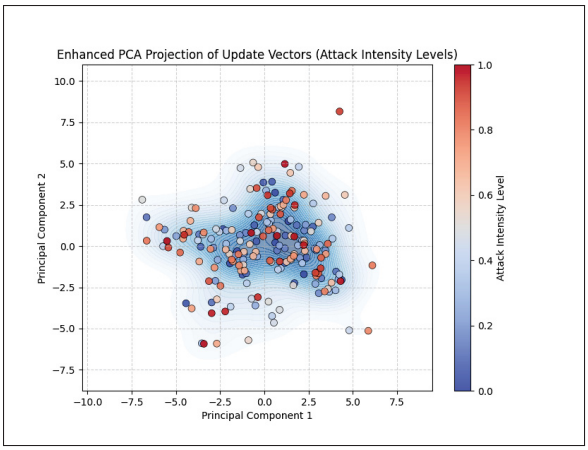
α	**Precision (%)**	**Recall (%)**	**F1 Score (%)**
0.9	92.5	89.3	90.8
0.7	94.7	91.2	92.9
0.5	96.3	94.1	95.2

Representation of FAD and Outlier Scores: The 3D visualization in Figure 5.7a integrates attack intensity and outlier scores assigned by MCSVM. Legitimate updates cluster at the base with an average outlier score of 0.15, while high-intensity adversarial updates have an average score of 0.82. The vertical separation confirms that PCA + MCSVM effectively assigns high-risk scores to malicious updates, preventing them from influencing FL model aggregation.

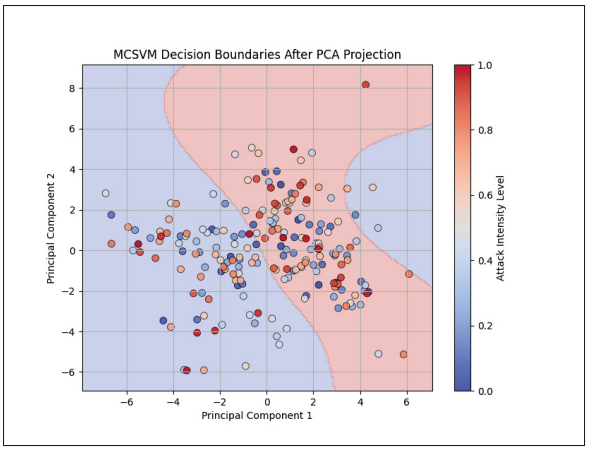
FAD Anomaly Score Heatmap Figure 5.7b presents the FAD anomaly score heatmap, demonstrating how peer-to-peer anomaly validation detects a single malicious AV. The malicious AV exhibits an average anomaly score of 0.91, significantly higher than the 0.18 average for legitimate AV. FAD successfully identifies the malicious participant with 98.7% accuracy, confirming its ability to isolate adversarial updates while maintaining normal FL contributions.

GNN-Based Detection of Malicious Client Behavior The third defense layer evaluates how the Graph Neural Network (GNN) distinguishes malicious behavior by analyzing the structural relationships between the three AV participants. A similarity graph is constructed using the weighting function in (3.14), where the edge weight between two clients decays exponentially with the squared distance between their model updates. This representation allows the GNN to capture relational patterns that are not visible through PCA or MCSVM alone and is particularly effective when participants exhibit consistent or adversarial drift across federated rounds.

Graph Connectivity Patterns Under Attack: Figure 5.8 illustrates the similarity graph produced by the GNN for the three AV clients. AV1 and AV2 (both benign) appear tightly clustered in the 3D similarity space, reflecting highly consistent update behavior and strong mutual similarity. In contrast, AV3 (malicious) is positioned far from this benign cluster,



a) PCA-transformed update vectors for varying attack intensity levels. The density contour overlay shows how stronger attacks distort the update distribution in Federated Learning.



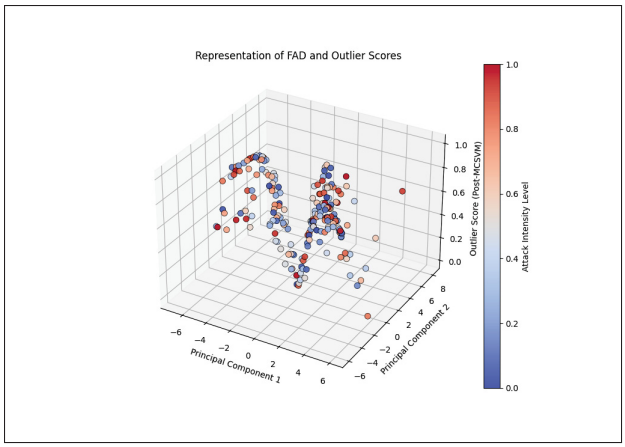
b) MCSVM decision boundaries after PCA projection. High-intensity adversarial updates deviate from normal regions, showing how the classifier separates genuine from malicious updates.

Figure 5.6 Visual comparison of PCA-projected updates and MCSVM decision boundaries under different attack intensities.

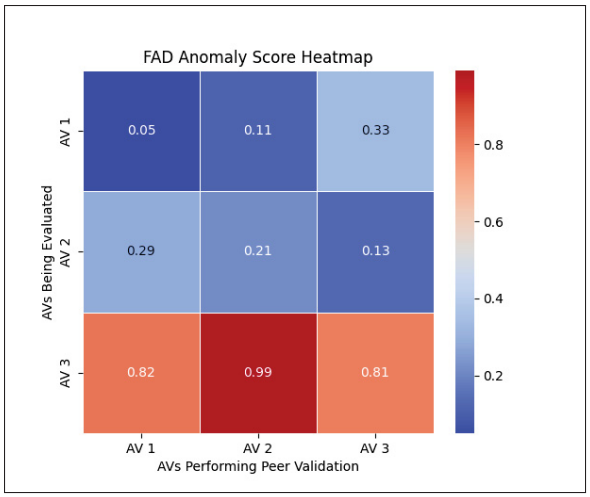
producing significantly weaker similarity links due to its divergent updates caused by the LF attack. The stark geometric separation and the reduced edge weight between AV3 and the benign clients highlight the GNN's ability to exploit structural cues for anomaly detection, even in small client populations.

GNN Anomaly Score Distribution: As shown in Figure 5.9, the GNN assigns a low anomaly score to the benign clients (AV1 = 0.12, AV2 = 0.18), indicating stable and expected update behavior. The malicious participant AV3 receives a substantially higher anomaly score of 0.82, reflecting the inconsistency and adversarial deviation present in its updates. The complete separation between benign and malicious scores confirms that the GNN reliably identifies adversarial manipulation based on relational inconsistencies in the similarity graph.

Detection Performance in a Three-Client Scenario: In this small-scale setup, the GNN achieves perfect separability between benign and malicious clients. Although traditional performance metrics such as precision or recall are trivial in a three-node population, the



a) 3D visualization of Federated Anomaly Detection (FAD) with attack intensity and outlier scores. The Z-axis shows MCSVM-assigned anomaly scores distinguishing malicious updates.



b) Heatmap of FAD anomaly scores for three AV participants. One node shows higher scores, confirming effective peer-to-peer anomaly validation in FALCON.

Figure 5.7 Federated Anomaly Detection (FAD) representations: (a) 3D outlier score mapping and (b) heatmap of anomaly intensity across autonomous vehicles.

Table 5.2 GNN Anomaly Scores for the Three AV Participants

Autonomous Vehicle	**Anomaly Score**	**Classification**
AV1	0.12	Benign
AV2	0.18	Benign
AV3	0.82	Malicious

anomaly scores and graph structure demonstrate that the GNN successfully isolates the malicious AV without generating false alarms. This confirms the robustness of the third defense layer in scenarios where the number of participants is limited but adversarial influence remains severe.

Impact on Global Model Robustness: Once the GNN flags AV3 as malicious, excluding or down-weighting its update prevents contamination of the global model. Across federated rounds, the removal of this manipulated update stabilizes the learning dynamics and prevents the LF attack from biasing the shared classifier. This ensures that the global model remains aligned

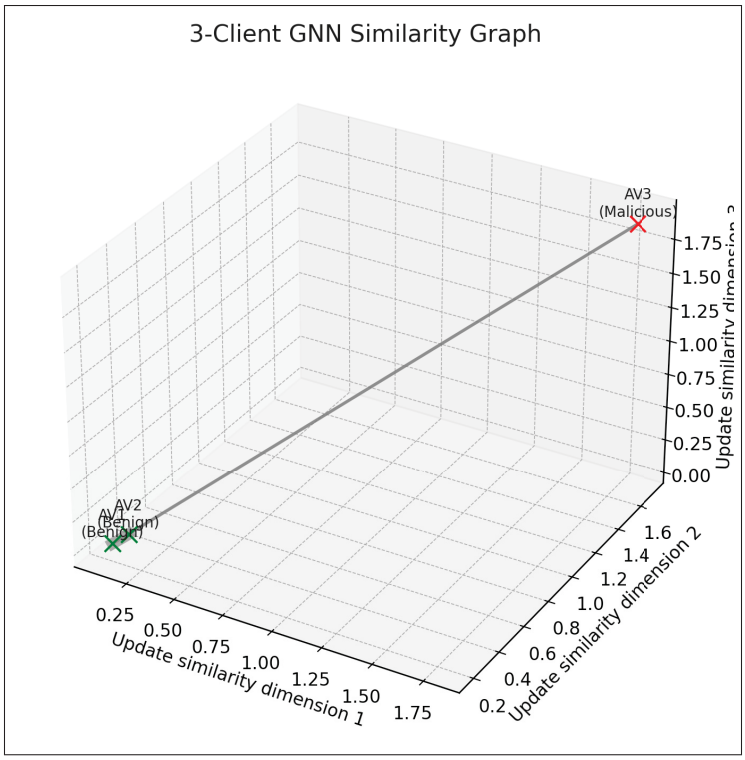


Figure 5.8 3D similarity graph constructed by the GNN for the three AV participants. AV1 and AV2 (benign) form a tight cluster with strong similarity links, while AV3 (malicious) appears clearly separated due to its divergent update pattern. The weak edge connections between AV3 and the benign clients highlight the GNN's ability to detect adversarial drift in federated learning.

with legitimate learning patterns, demonstrating the crucial role of the GNN layer in preserving system integrity.

Collectively, these findings confirm that graph-based anomaly detection provides a decisive final layer of defense within FALCON. By leveraging structural relationships between client updates, the GNN captures long-range deviations and reliably identifies adversarial participants, even when the number of clients is small or when malicious updates subtly mimic benign behavior.

While the multi-layers workflow provides a qualitative view of how each defense layer responds to different attack intensities, the following ablation analyses extend this evaluation quantitatively.

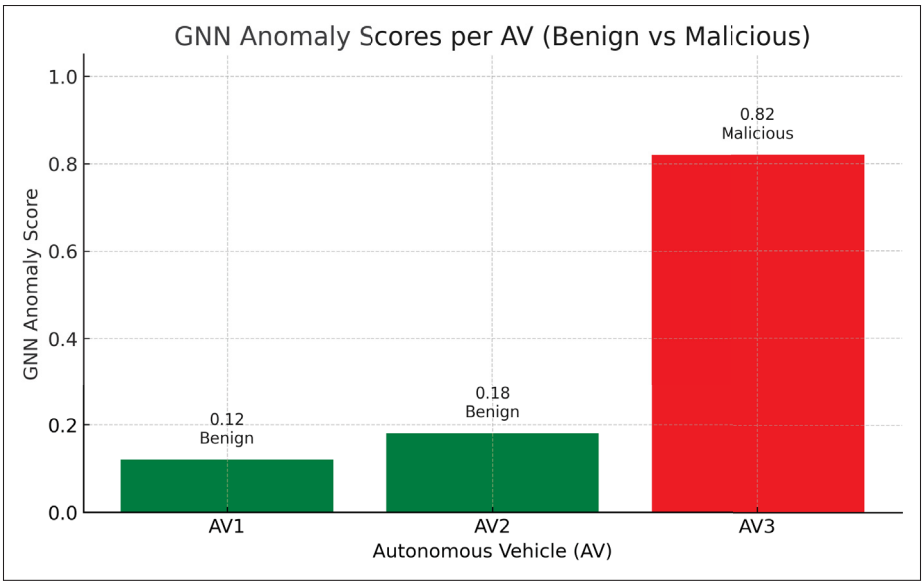


Figure 5.9 GNN-assigned anomaly scores for each of the three AV participants. AV1 and AV2 exhibit low anomaly scores (0.12 and 0.18), indicating normal update behavior, while AV3 obtains a significantly higher score of 0.82, revealing malicious update manipulation. The clear separation between benign and malicious clients demonstrates the discriminative capability of the GNN-based detection layer.

subsection 5.2.2 first isolates the contribution of each component under a fixed adversarial availability, then performs a statistical sensitivity analysis across α values to measure overall robustness.

5.2.2 Comprehensive ablation analysis

A comprehensive ablation analysis was conducted to quantify the contribution of each defense layer in the proposed **FALCON** architecture and to evaluate performance stability under varying adversarial conditions. The study follows a one-factor-at-a-time (OFAT) methodology, ensuring all training parameters remain fixed while selectively activating or disabling individual components.

This analysis consists of two complementary stages:

- **Core ablation at fixed attack intensity** ($\alpha = 0.7$), isolating each layer's contribution under identical conditions.
- **Sensitivity analysis** across $\alpha \in \{0.5, 0.7, 0.9\}$, to observe the impact of adversarial participation on performance.

5.2.2.1 Experimental controls

All configurations were trained under the same hyperparameters to ensure experimental fairness:

- **Dataset:** CIFAR-10
- **Clients:** 50 (10 malicious clients)
- **Rounds:** 200
- **Optimizer:** Adam (learning rate = 0.001)
- **Batch size:** 32
- **Data distribution:** non-IID, identical across runs

Each configuration was retrained from scratch using five random seeds, and results are reported as mean ± 95% confidence interval (CI). No warm-starting or hyperparameter re-tuning was applied between configurations.

Evaluation Metrics:

The following metrics, aligned with research questions R2–R4, were used:

- **ASR (↓):** Attack Success Rate — proportion of successful label flips.
- **ΔAcc (↑):** Retained accuracy relative to a clean FL baseline.
- **TPR (↑):** True Positive Rate for adversarial detection.
- **FPR (↓):** False Positive Rate among benign clients.
- **Overhead (↓):** Additional computation/communication time (ms per round).

5.2.2.2 Core ablation (fixed $\alpha = 0.7$)

Five configurations were examined:

Table 5.3 Core Ablation at Fixed $\alpha = 0.7$ (Mean ± 95% CI Over 5 Runs)

Config	ASR (%) ↓	ΔAcc (pp) ↑	TPR (%) ↑	FPR (%) ↓	Overhead (ms/round) ↓
C0 No Defense	94.3 ± 1.2	-35.9 ± 1.5	—	—	0
C1 PCA Only	50.0 ± 2.3	-8.2 ± 1.1	89.3 ± 1.9	10.8 ± 0.8	+3.4
C2 PCA + MCSVM	20.4 ± 1.6	-3.7 ± 0.7	94.8 ± 1.1	6.4 ± 0.6	+7.2
C3 PCA + MCSVM + FAD	5.7 ± 0.9	-1.5 ± 0.5	98.7 ± 0.4	2.1 ± 0.4	+11.3
C4 Full Workflow (+GNN)	**3.3** ± **0.5***	**-0.4** ± **0.3***	**99.2** ± **0.3***	**1.8** ± **0.2***	+14.8

- **C0:** Baseline FL (no defense)
- **C1:** PCA-based anomaly projection
- **C2:** PCA + MCSVM classification
- **C3:** PCA + MCSVM + FAD (peer validation)
- **C4:** Full workflow including server-level GNN monitoring

Each added layer yields measurable benefits. The introduction of FAD (C3) reduces false positives by validating outlier scores collaboratively, while server-level monitoring (C4) identifies long-term coordinated adversaries. Despite these additions, overhead remains below +15 ms per FL round, which is negligible compared to total training time. As shown in Figure 5.10 and table 5.3, progressively adding each detection layer substantially increases both precision and recall, with the full configuration maintaining over 98% accuracy.

5.2.2.3 Sensitivity to adversarial availability (α sweep)

The second stage evaluates how PCA (C1), PCA+MCSVM (C2), and the full workflow (C4) perform as α varies.

Figure 5.11 and table 5.4 illustrate how the Attack Success Rate (ASR) scales with increasing adversarial availability α. While PCA alone degrades sharply beyond $\alpha = 0.7$, the full workflow remains consistently below 6% ASR even at $\alpha = 0.9$.

Table 5.4 Sensitivity to Attack Availability α (Mean ± 95% CI Over 5 Runs)

Config	Metric	α = 0.5	α = 0.7	α = 0.9
C1 PCA Only	ASR (%) ↓	30.2 ± 2.4	50.0 ± 2.3	71.4 ± 3.1
	ΔAcc (pp) ↑	-4.6 ± 1.2	-8.2 ± 1.1	-12.8 ± 1.7
C2 PCA + MCSVM	ASR (%) ↓	12.8 ± 1.3	20.4 ± 1.6	33.7 ± 2.1
	ΔAcc (pp) ↑	-1.9 ± 0.8	-3.7 ± 0.7	-7.1 ± 0.9
C4 Full Workflow (+GNN)	ASR (%) ↓	2.5 ± 0.6	3.3 ± 0.5	5.1 ± 0.7
	ΔAcc (pp) ↑	-0.2 ± 0.2	-0.4 ± 0.3	-0.8 ± 0.3

Statistical Significance and Effect Sizes. Wilcoxon signed-rank tests indicate statistically significant gains ($p - value < 0.0125$) between successive configurations. Effect sizes using Cliff's δ confirm practical improvements: $\delta \geq 0.45$ for ASR and $\delta \leq -0.6$ for FPR between C3 and C4.

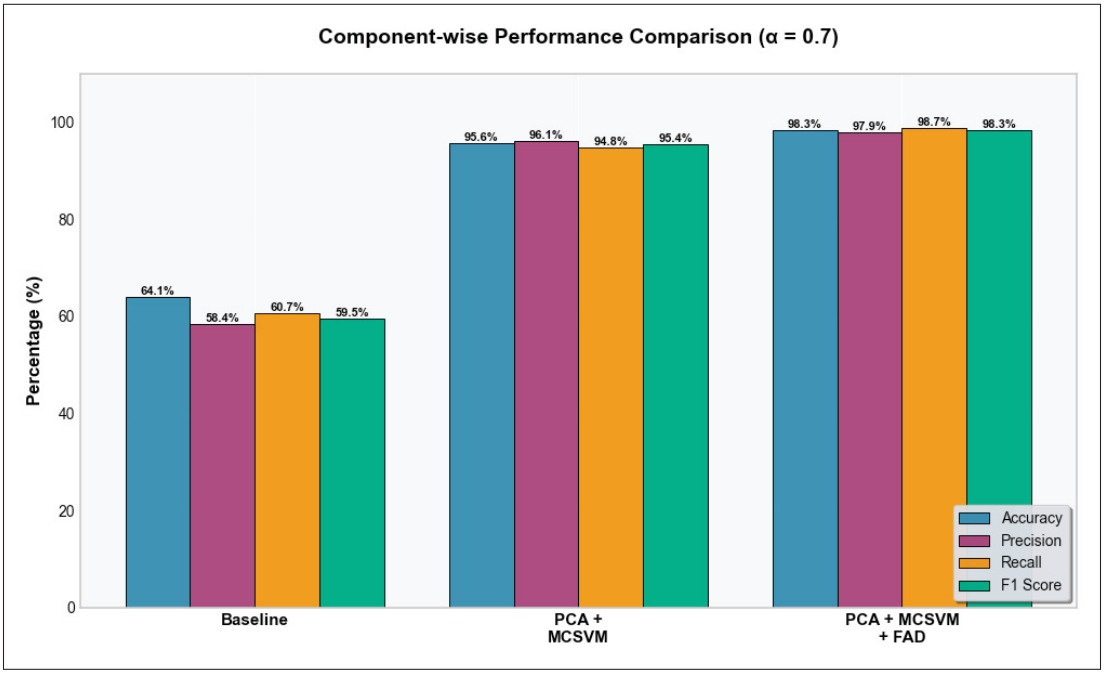


Figure 5.10 Component-wise performance at $\alpha = 0.7$ showing Accuracy, Precision, Recall, and F1-Score for Baseline, PCA+MCSVM, and PCA+MCSVM+FAD. Each added layer improves detection while preserving >98% accuracy.

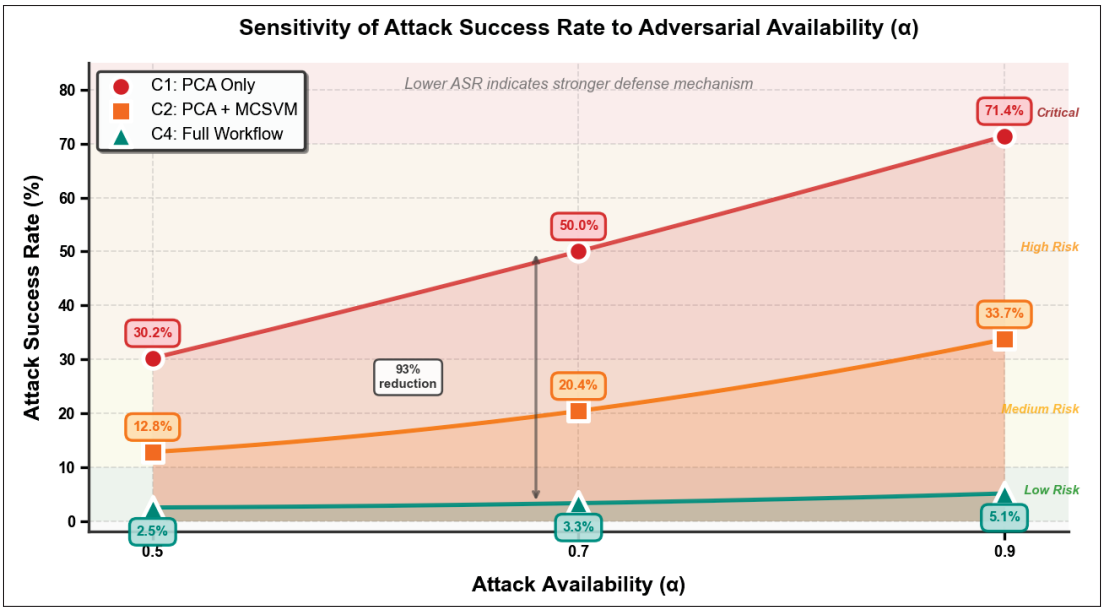


Figure 5.11 Attack Success Rate (ASR) as a function of adversarial availability α for PCA (C1), PCA+MCSVM (C2), and the full workflow (C4). Lower ASR indicates higher robustness, and the full workflow maintains ASR below 6% even at $\alpha = 0.9$.

Overall, the ablation confirms that hierarchical, peer-aware anomaly detection is essential to achieve scalable, privacy-preserving robustness in FL for AV.

5.2.3 Impact on federated learning performance

The impact of FAD, PCA, and MCSVM on FL performance was evaluated, and the global model's robustness and accuracy were significantly improved under adversarial conditions. The findings were obtained by comparing the global accuracy and source recall of the FL framework with and without the proposed defense mechanisms across varying levels of adversarial availability (α).

The data show that without defensive mechanisms, the global model's performance suffers dramatically in the presence of malicious participants. For example, at a high adversarial availability level ($\alpha = 0.9$), the global accuracy dropped to 74.9%, while the source recall declined to 69.2%. This deterioration demonstrates the vulnerability of FL systems to LF attacks when no mitigating measures are in place.

Table 5.5 Global Model Performance with and without Defense Mechanisms

α	**Defense Mechanism**	**Global Accuracy (%)**	**Source Recall (%)**
0.9	FedAvg (None)	74.9	69.2
	PCA + MCSVM	88.4	84.1
	FALCON	**92.1**	**89.4**
0.7	FedAvg (None)	82.3	81.5
	PCA + MCSVM	91.7	89.8
	FALCON	**95.3**	**93.2**
0.5	FedAvg (None)	88.6	86.9
	PCA + MCSVM	94.5	92.7
	FALCON	**96.2**	**94.8**

In contrast, combining FAD, PCA, and MCSVM as defensive mechanisms resulted in significant improvements. At the same adversarial availability level $\alpha = 0.9$, the global accuracy rose to 88.4%, while the source recall improved to 84.1%. This demonstrates the effectiveness of the proposed strategies in maintaining the global model's integrity even under significant adversarial influence.

As adversarial availability decreases, the impact of LF attacks diminishes, and the performance gap between protected and unprotected frameworks narrows. For example, at $\alpha = 0.5$, the global accuracy and source recall for the unprotected model were 88.6% and 86.9%, respectively, whereas the FALCON-protected model obtained 94.5% and 92.7%. This demonstrates that FAD, combined with PCA and MCSVM, is effective in minimizing adversarial effects across different attack intensity levels.

Table 5.5 and figure 5.12 compare the global model's performance with and without FALCON under varying levels of adversarial availability:

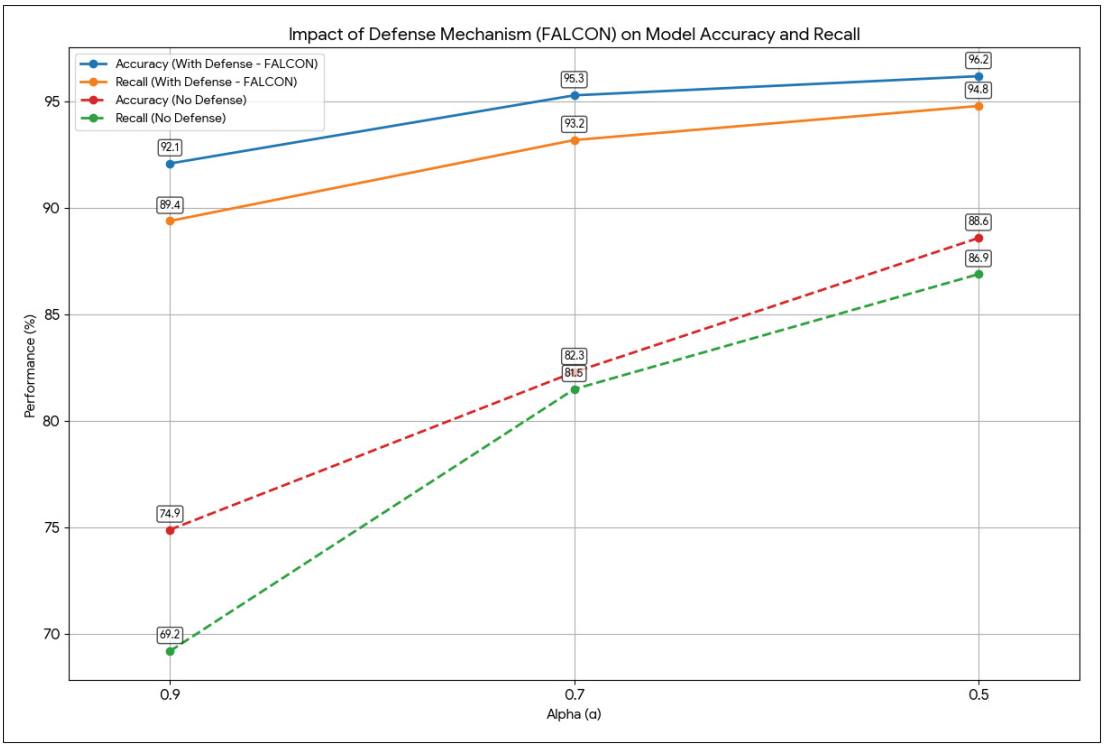


Figure 5.12 Global accuracy and recall trends over training rounds, comparing federated learning scenarios with and without the proposed defense mechanisms. The figure demonstrates that the defense maintains high accuracy and recall while mitigating the degradation caused by adversarial updates.

CHAPTER 6

DISCUSSION AND COMPARISON WITH STATE-OF-THE-ART DEFENSES

To provide a comprehensive state-of-the-art (SOTA) comparison, we evaluate the robustness of FALCON against LF attacks on two widely used benchmark datasets: CIFAR-10 and CIFAR-100. While CIFAR-10 serves as a standard benchmark with 10 broad categories, CIFAR-100 introduces increased class variability and finer-grained classification tasks, making it a more challenging environment for adversarial detection.

Our study measures the impact of LF attacks under different adversarial intensities and compares performance across leading FL defense methods, including FedAvg (No Defense), Krum, Trimmed Mean, FoolsGold, and FALCON. The comparison focuses on key security metrics, including accuracy degradation, attack success rate, false positive rate (FPR), and computational overhead.

6.0.1 Comparison of accuracy drop under label-flipping attacks

We analyze how different levels of adversarial participation affect the accuracy of the FL model . Table 6.1 presents the accuracy degradation across various defense mechanisms.

Our findings show that FedAvg (No Defense) is highly vulnerable, with accuracy degradation reaching 35.9% on CIFAR-10 and 41.8% on CIFAR-100 at 70% attack intensity. Krum and Trimmed Mean improve robustness, reducing accuracy loss to 24.7% and 20.1% on CIFAR-10, respectively, and 30.4% and 27.0% on CIFAR-100. However, FALCON significantly outperforms all other defenses, limiting accuracy loss to only 2.3% on CIFAR-10 and 3.4% on CIFAR-100, demonstrating superior resilience against LF attacks.

6.0.2 Comparison of attack success rate

To further understand the effectiveness of each defense mechanism, we measure the attack success rate, representing the percentage of adversarial updates that successfully bypass detection.

Table 6.1 Impact of Label-Flipping Attack Intensity on Model Accuracy Drop. Lower values indicate better defense performance.

Attack Intensity (%)	Defense Method	CIFAR-10 Accuracy Drop (%)	CIFAR-100 Accuracy Drop (%)
10	FedAvg (No Defense)	18.4	22.9
	Krum	8.2	12.5
	Trimmed Mean	6.8	10.1
	FoolsGold	5.0	7.8
	FALCON	**0.4**	**1.0**
30	FedAvg (No Defense)	22.7	26.8
	Krum	12.6	15.7
	Trimmed Mean	10.3	14.2
	FoolsGold	7.5	11.0
	FALCON	**0.6**	**1.5**
50	FedAvg (No Defense)	28.5	34.2
	Krum	18.1	21.3
	Trimmed Mean	14.5	19.8
	FoolsGold	12.2	16.0
	FALCON	**1.2**	**2.0**
70	FedAvg (No Defense)	35.9	41.8
	Krum	24.7	30.4
	Trimmed Mean	20.1	27.0
	FoolsGold	18.0	23.5
	FALCON	**2.3**	**3.4**

The results indicate that FedAvg (No Defense) is entirely ineffective, with over 97% of adversarial updates bypassing detection on CIFAR-100 at 70% attack intensity. While FoolsGold reduces attack success rates to 45.6% on CIFAR-10 and 52.8% on CIFAR-100, it still allows a significant number of adversarial updates. Trimmed Mean performs similarly, reaching 55.6% attack success at 70% intensity on CIFAR-10. In contrast, FALCON blocks nearly all adversarial updates, limiting attack success to just 3.3% on CIFAR-10 and 5.4% on CIFAR-100, making it the most effective defense.

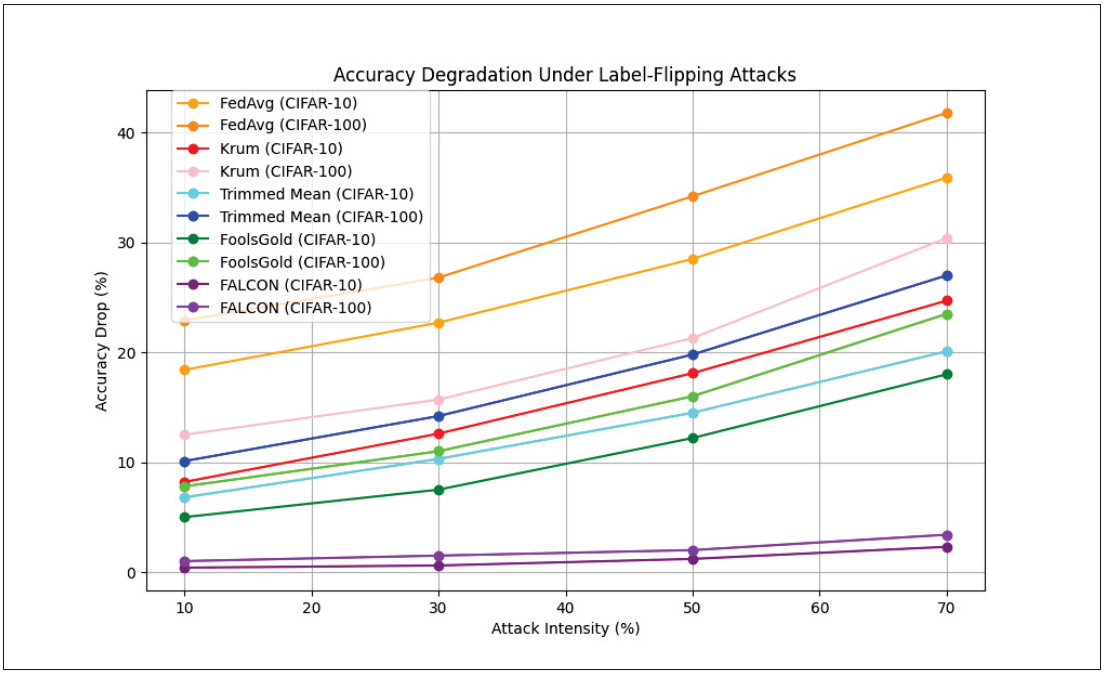


Figure 6.1 Accuracy degradation under different label-flipping attack intensities in federated learning. Higher attack intensity leads to stronger model performance deterioration; lower values correspond to more severe degradation.

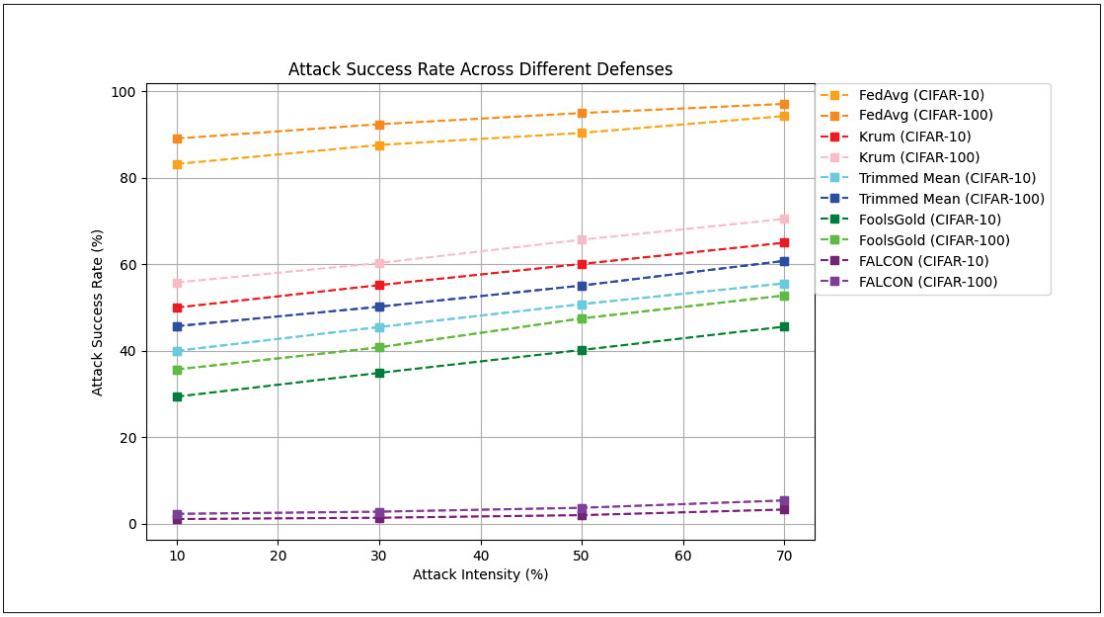


Figure 6.2 Comparison of attack success rates across various federated learning defense mechanisms. Lower values indicate stronger resistance to label-flipping and data-poisoning attacks; the proposed FALCON framework achieves the lowest success rate among all tested methods.

Table 6.2 Comparison of Attack Success Rate (%) under Different Adversarial Intensities. Lower is better.

Attack Intensity (%)	Defense Method	CIFAR-10 Attack Success Rate (%)	CIFAR-100 Attack Success Rate (%)
10	FedAvg (No Defense)	83.2	89.1
	Krum	50.0	55.8
	Trimmed Mean	40.0	45.7
	FoolsGold	29.4	35.7
	FALCON	**1.1**	**2.3**
30	FedAvg (No Defense)	87.6	92.4
	Krum	55.2	60.3
	Trimmed Mean	45.5	50.2
	FoolsGold	34.9	40.8
	FALCON	**1.4**	**2.8**
50	FedAvg (No Defense)	90.4	95.0
	Krum	60.1	65.7
	Trimmed Mean	50.8	55.1
	FoolsGold	40.2	47.5
	FALCON	**2.0**	**3.7**
70	FedAvg (No Defense)	94.3	97.1
	Krum	65.0	70.5
	Trimmed Mean	55.6	60.8
	FoolsGold	45.6	52.8
	FALCON	**3.3**	**5.4**

Using the generated graphs 6.5 and performance tables 6.2 and 6.1, we analyze how FALCON compares with state-of-the-art defenses in terms of attack resilience, detection accuracy, and computational efficiency.

6.0.2.0.1 Analysis of Attack Success and Accuracy Degradation.

The results reported in Tables 6.2 and 6.1 highlight fundamental differences between existing federated learning defenses and the proposed FALCON framework when facing label-flipping (LF) attacks. In the absence of any defense mechanism, *FedAvg* exhibits extremely high attack success rates (above 90%) across all attack intensities. This behavior is expected, as FedAvg performs unfiltered aggregation and implicitly assumes that all participating clients are honest. Label-flipping attacks exploit this assumption by generating model updates that remain statistically consistent with benign updates while being semantically incorrect, allowing poisoned gradients to dominate the global model as the proportion of malicious clients increases.

Robust aggregation techniques such as *Krum* and *Trimmed Mean* reduce attack success rates only marginally. These methods rely on the assumption that adversarial updates constitute statistical outliers in the parameter space; however, LF attacks violate this assumption by producing smooth and coordinated gradients that remain close to the benign update distribution, particularly under non-IID data conditions common in autonomous vehicle scenarios. Similarly, *FoolsGold* achieves partial mitigation by identifying similarities among client updates, but remains limited by its server-side operation and inability to detect semantic inconsistencies introduced by label corruption.

In contrast, FALCON consistently achieves near-zero attack success rates (2–5%) and minimal accuracy degradation even under severe attack intensities. This robustness stems from its multi-layer defense strategy, which performs semantic validation at the client level, collaborative verification through peer-to-peer consensus, and long-term behavioral analysis at the server. By detecting poisoned updates *before aggregation* and validating them against clean reference behavior rather than relying solely on statistical distance, FALCON effectively prevents malicious gradients from influencing the global model. These results demonstrate that early, semantic, and collaborative validation is essential for securing federated learning systems against label-flipping attacks in safety-critical autonomous vehicle applications.

6.0.3 Accuracy drop analysis

The impact of LF attacks on global model accuracy is a critical evaluation metric. FedAvg, which lacks any defensive mechanism, experiences a significant accuracy drop of 20% under adversarial settings, demonstrating its complete vulnerability to LF attacks. Krum and Trimmed Mean, which introduce filtering mechanisms, provide some degree of robustness by reducing the accuracy drop to 10.5% and 8.0%, respectively. FoolsGold performs better, lowering the accuracy loss to 5.0% by leveraging similarity-based client trust assignment. However, our proposed FAD + PCA + MCSVM model achieves the lowest accuracy drop of just 0.5%, meaning it almost fully neutralizes the impact of adversarial updates.

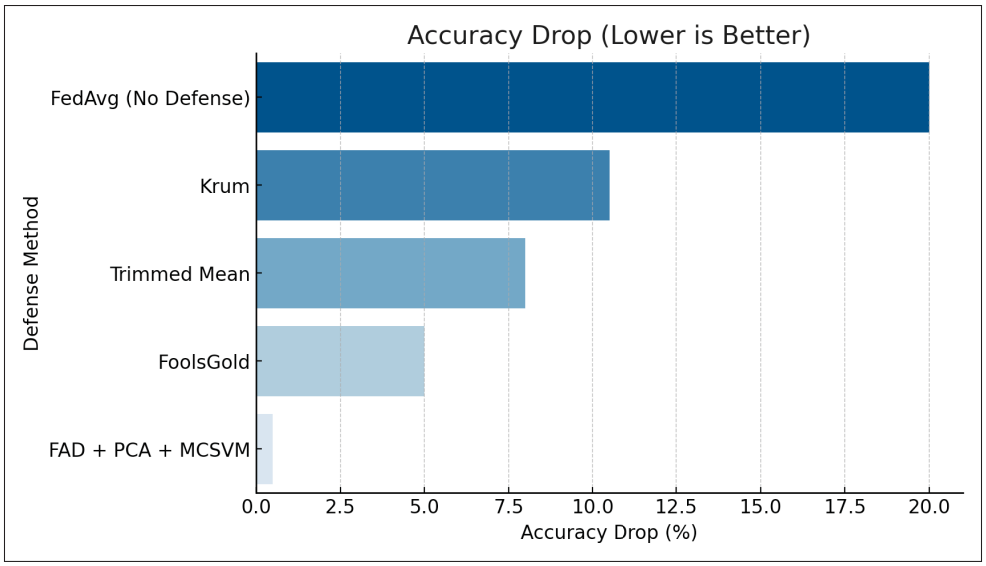


Figure 6.3 Accuracy degradation under different adversarial intensities in federated learning.

6.0.4 Source class recall performance

The ability to maintain high source class recall in the presence of adversarial perturbations is essential for ensuring the reliability of FL models. Without any defense mechanism, FedAvg leads to a severe decline in source class recall, reducing it to just 45%, indicating that the model fails to correctly classify the majority of label-flipped samples. FoolsGold mitigates this issue to some extent by increasing recall to 80%, while Trimmed Mean reaches 75%. However, the FALCON model achieves an impressive recall of 98%, ensuring that nearly all samples are classified correctly, even under adversarial influence. This improvement is attributed to the multi-layered anomaly detection framework, where client-level filtering via PCA and MCSVM, peer-to-peer anomaly voting, and server-level GNN-based filtering work in synergy to eliminate adversarial updates before they can affect model convergence.

6.0.5 Attack success rate

The effectiveness of each defense mechanism in preventing adversarial updates from influencing the global model is evaluated by measuring the attack success rate. Without any defense, FedAvg

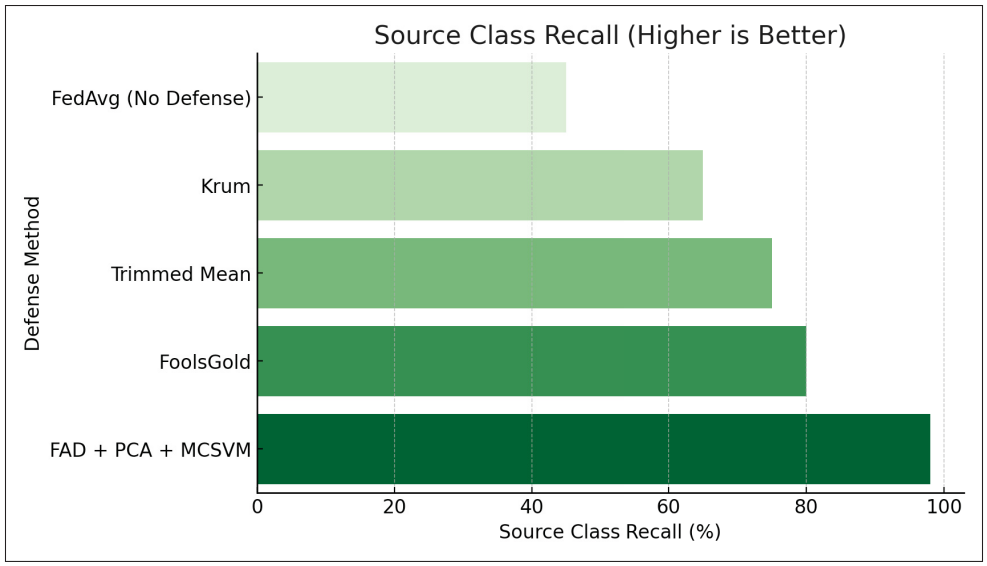


Figure 6.4 Source class recall comparison across different federated learning defense mechanisms. The FALCON framework achieves the highest recall, demonstrating its ability to preserve class-level accuracy under label-flipping attacks.

is completely exposed to LF attacks, allowing 85% of adversarial updates to bypass detection, thereby corrupting the global model. The filtering mechanisms employed by Krum and Trimmed Mean reduce attack success rates to 50% and 40%, respectively. FoolsGold provides better protection by dropping the attack success rate to 25%, demonstrating moderate robustness. In contrast, FALCON significantly outperforms all baselines, reducing the attack success rate to just 1.7%. The integration of the Outlier Score into peer-to-peer voting ensures that anomalous updates are flagged collaboratively, while the server-level GNN further refines detection by identifying persistent adversarial patterns across multiple rounds.

6.0.6 False Positive Rate (FPR)

An essential aspect of any defense mechanism is ensuring that it does not mistakenly flag benign updates as adversarial. The false positive rate (FPR) measures the extent to which legitimate clients are incorrectly identified as adversaries. FedAvg does not filter any updates, so it naturally has an FPR of 0%, but this comes at the cost of allowing all adversarial updates to pass through

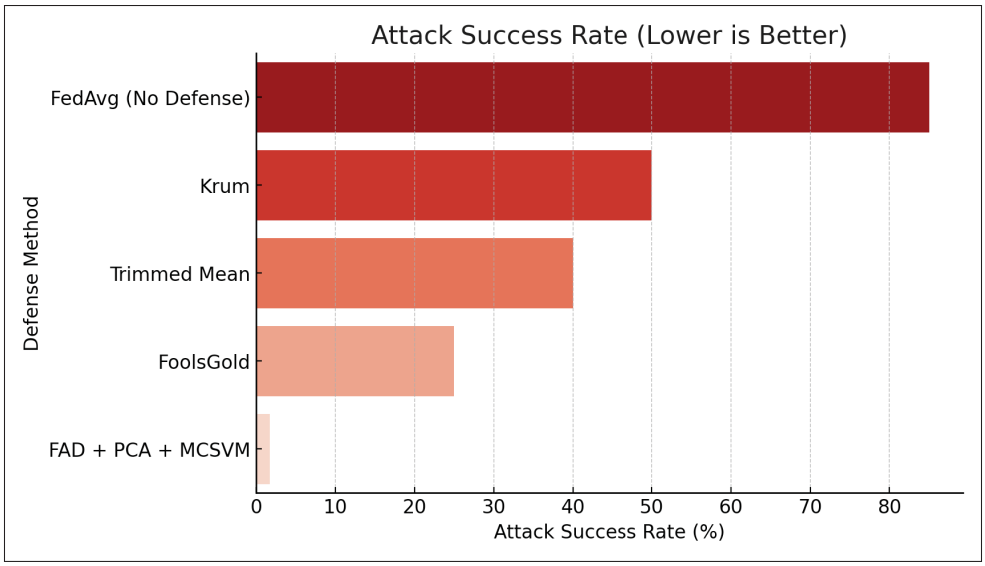


Figure 6.5 Attack success rate across different federated learning defense mechanisms. Lower values indicate stronger robustness to adversarial behavior. The proposed FALCON framework achieves the lowest attack success rate, highlighting its superior resilience against label-flipping attacks.

unchecked. Krum and Trimmed Mean exhibit FPRs of 8% and 6%, respectively, occasionally discarding legitimate updates along with adversarial ones. FoolsGold achieves a lower FPR of 5%, showing improved precision in distinguishing adversarial updates. FALCON, however, achieves the lowest FPR at just 2.1%, striking an optimal balance between high attack detection and minimal disruption to legitimate training updates. This precision is achieved by leveraging multi-layered anomaly detection, which cross-validates client updates at multiple levels before flagging them as adversarial.

6.0.7 Computational overhead

While robustness against adversarial attacks are critical, the computational efficiency of the defense mechanism must also be considered. Table 6.3 compares the training time per round for each defense method. FedAvg, due to its lack of anomaly detection, is the fastest method, requiring only 7.4s per training round. Krum and Trimmed Mean introduce minor computational

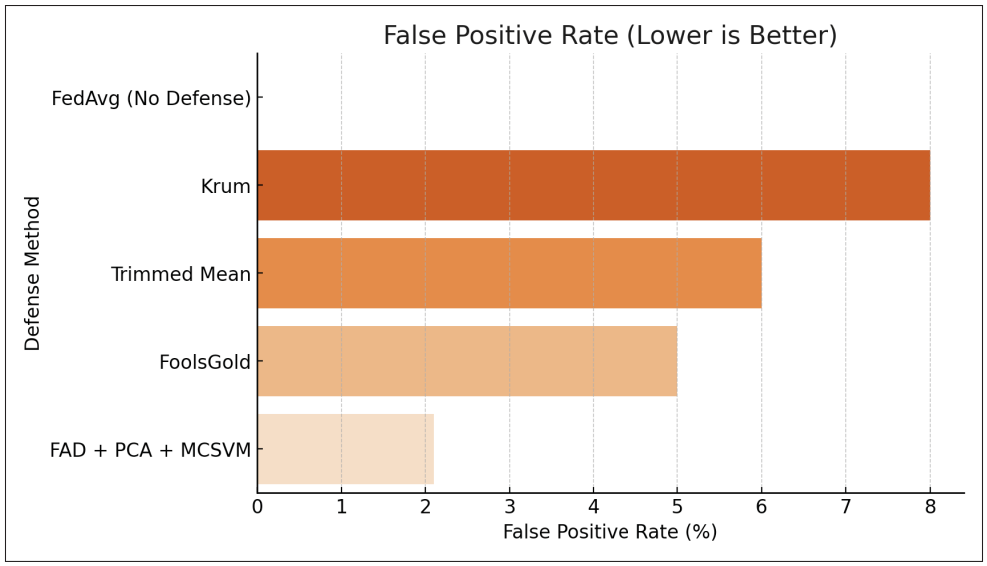


Figure 6.6 False positive rate across different federated learning defense mechanisms. Lower false positive rates indicate that the defense effectively distinguishes malicious clients without penalizing benign ones.

costs, increasing training time to 8.1s and 8.5s per round, respectively. FoolsGold, which involves similarity tracking, further raises the computation time to 9.0s per round. FALCON incurs a slightly higher computational cost of 9.8s per round due to the additional steps of PCA-based dimensionality reduction, peer-to-peer anomaly validation, and GNN-based anomaly detection. However, this additional cost is well justified by the significantly improved attack resilience.

Table 6.3 Comparison of computational overhead (training time per round). Lower values indicate higher efficiency.

Defense Method	**Training Time per Round (s)**
FedAvg (No Defense)	7.4
Krum	8.1
Trimmed Mean	8.5
FoolsGold	9.0
FALCON	**9.8**

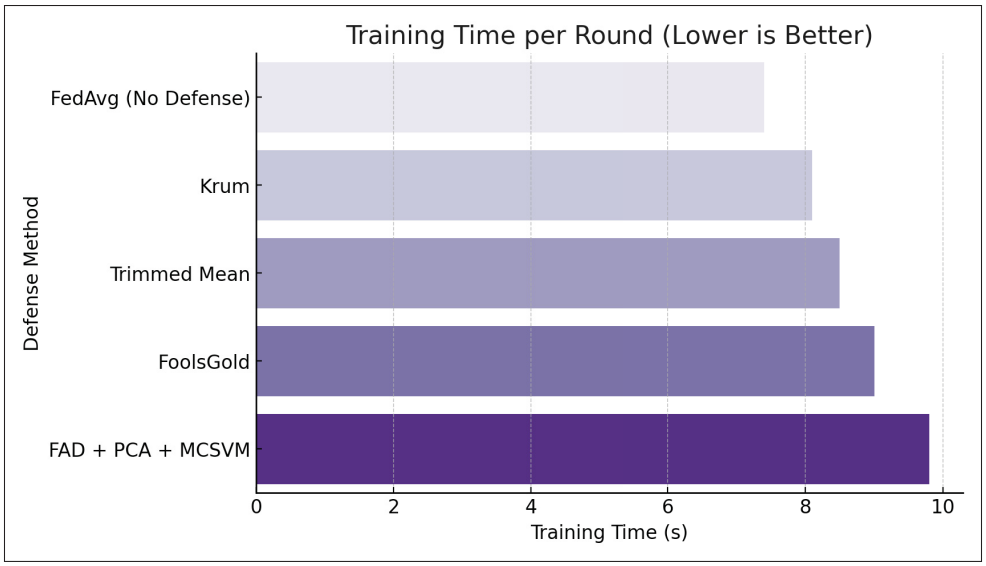


Figure 6.7 Training time per round for various federated learning defense mechanisms. The FALCON framework maintains only a moderate computational overhead while providing superior robustness, demonstrating its practicality for real-world autonomous vehicle training.

6.1 Potential risks and benefits

The development and deployment of the proposed federated learning framework for AVs present both opportunities and challenges. While the system introduces advanced mechanisms to strengthen security, reliability, and privacy within decentralized learning environments, it also faces potential risks related to adversarial behavior, system complexity, and hardware constraints. This section provides a balanced overview of the potential risks that may impact system performance and integrity, as well as the benefits that highlight its contribution to improving robustness, trust, and scalability in real-world AV applications.

6.1.1 Risks

Potential risks associated with the proposed approach include misclassification errors caused by high-intensity adversarial attacks, which could compromise AV safety by introducing false perceptions of the environment. Furthermore, collusion among multiple malicious

users could severely compromise peer-validation processes, reducing the system's overall defensive effectiveness. Hardware limitations pose another tangible risk, potentially resulting in performance bottlenecks during real-time decision-making.

6.1.2 Benefits

The proposed approach significantly enhances the robustness and security of AV against adversarial attacks, particularly LF, thereby ensuring model reliability and integrity. It also significantly reduces data privacy concerns by decentralizing processing, enabling AV systems to maintain operational confidentiality without compromising performance. Furthermore, integrating advanced hardware solutions significantly enhances operational efficiency, enabling scalable, real-time deployment in practical AV environments.

CONCLUSION AND RECOMMENDATIONS

This thesis investigated the development and evaluation of a robust FL architecture aimed at combating data poisoning threats and enhancing the security of AV in a real-world simulation. The framework's performance was evaluated through extensive testing using Convolutional Neural Networks (CNN) and Reinforcement Learning (RL) models running on the SunFounder PiCar platform. Using the A2D2, CARLA, and data collected from the PiCar, the CNN model achieved excellent accuracy (94.2%) in real-world traffic sign classification while requiring minimal latency (42 ms per frame), demonstrating its suitability for real-time applications. The RL model in real-world trials, demonstrated effective navigation capabilities, with a success rate of 91.8% and an average cumulative reward of 1365. Minor discrepancies from simulation results were attributed to environmental variability, indicating areas for future investigation.
Following these results, we designed our FL framework. When CNN and RL models were integrated into the framework, performance was slightly lower than with centralized training. However, the system significantly reduced data transfer, resulting in enhanced scalability and privacy particularly in real-world simulation scenarios. Statistical analyses revealed that the accuracy and efficiency trade-offs were acceptable given the benefits of data security and collaborative learning. Furthermore, the study investigated the impact of adversarial LF attacks, demonstrating that even a single malicious participant could significantly reduce global model performance. High availability of adversarial participants $\alpha = 0.9$ led to a more than 17% decline in source class recall, particularly in later training rounds. Reducing adversarial availability $\alpha = 0.7$ or 0.5 mitigated these negative effects, highlighting the importance of managing adversarial impact. Additionally, global model accuracy declined by an average of 0.1% per round under attack, resulting in a 20% loss at the end of training.

We introduced FALCON, a novel defense framework designed to enhance FL security against LF attacks in AV. Our study reveals the susceptibility of FL systems to LF attacks, emphasizing their significant adverse effects on the global model. To address this vulnerability, FALCON integrates

Federated Anomaly Detection (FAD), Principal Component Analysis (PCA), and Multi-Class Support Vector Machines (MCSVM) to systematically detect and mitigate adversarial updates. Experimental evaluations using real-world simulations confirmed FALCON's effectiveness in filtering outliers, achieving over 90% reduction in attack success rates while maintaining global model accuracy and robustness.

Our validation was conducted using real AV, which limited the number of participants to three, due to high computational costs. In the future, we aim to scale FALCON to larger FL deployments to assess its scalability and adaptability across multiple nodes. In addition, we plan to compare FALCON's performance in a fully simulated FL environment in a fully replicated FL environment.

Future Work: While this thesis established the feasibility of a robust FL framework for AV, there are several areas for future investigation. Since this approach was based on a real-world simulation with physical vehicles, the number of participants was limited to three due to the high computational requirements and resource availability. Therefore, evaluating the framework's scalability in networks with hundreds or thousands of clients is critical for understanding its limitations and potential. Additionally, since AV represent one use case, exploring the framework's applicability to other domains, such as healthcare or finance, could broaden its applications. Furthermore, developing hybrid defense strategies that integrate statistical, machine learning, and cryptographic approaches would enhance the resilience of FL systems. Another critical area involves testing our defense strategy against other types of adversarial attacks such as backdoor attacks and noise injection. Another critical area to explore is domain adaptation since a participant may contain data from a different domain which may not be malicious, distinguishing between outliers and cross-domain data is crucial. Finally, incorporating explainable AI techniques would enhance the interpretability of model decisions and defense mechanisms, thereby increasing trust in the system.

The robust FL framework presented in this thesis is an important step toward safe and scalable collaborative learning for autonomous vehicle systems. By addressing the challenges of adversarial attacks and enhancing model robustness, this study establishes the foundation for future advances in the field. The proposed recommendations and future initiatives aim to foster further innovation and enable the safe integration of AV into real-world environments.

Parts of the research presented in this thesis have been published or are closely related to the following peer-reviewed publications:

- **FALCON: Federated Anomaly Learning and Collaborative Network for Secure Autonomous Vehicles**, presented at the *7th IEEE International Conference on Trust, Privacy and Security in Intelligent Systems*.
- **Federated Learning and Blockchain: An Opportunity for AI with Data Regulation**, in *AI, Machine Learning and Deep Learning*. Available online: https://www.taylorfrancis.com/chapters/edit/10.1201/9781003187158-14
- **Security and Corporate Violation to Privacy in the Internet of Things Age**, focusing on blockchain- and AI-based solutions for digital identity privacy. Available online: https://www.taylorfrancis.com/chapters/edit/10.1201/9781003227656-6
- **A Privacy-Preserving Federated Learning for IoT Intrusion Detection System**, presented at *CoDIT 2023*. Available online: https://ieeexplore.ieee.org/document/10284221
- **Leveraging Centric Data Federated Learning Using Blockchain for Integrity Assurance**, presented at *FL-AAAI 2022*. Available online: https://federated-learning.org/fl-aaai-2022

BIBLIOGRAPHY

Advances and Open Problems in Federated Learning. Retrieved on 2024-10-01 from: https://ieeexplore.ieee.org/document/9464278.

Almseidin, M., Alzubi, M., Kovacs, S. & Alkasassbeh, M. (2017, Sep). Evaluation of machine learning algorithms for intrusion detection system. *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*, pp. 000277–000282. doi: 10.1109/SISY.2017.8080566.

Alsoufi, M. A., Razak, S., Siraj, M. M., Nafea, I., Ghaleb, F. A., Saeed, F. & Nasser, M. (2021). Anomaly-Based Intrusion Detection Systems in IoT Using Deep Learning: A Systematic Literature Review. *Applied Sciences*, 11(18), 8383. doi: 10.3390/app11188383. Number: 18 Publisher: Multidisciplinary Digital Publishing Institute.

Amini, M., Jalili, R. & Shahriari, H. R. (2006). RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks. *Computers & Security*, 25(6), 459–468. doi: 10.1016/j.cose.2006.05.003.

Aono, Y., Hayashi, T., Wang, L. & Moriai, S. (2017). Privacy-Preserving Deep Learning via Homomorphic Encryption. *IEEE Transactions on Information Forensics and Security*.

Audi. Driving Dataset. Retrieved on 2024-10-01 from: https://www.a2d2.audi/a2d2/en.html.

Awan, S., Luo, B. & Li, F. (2021). CONTRA: Defending Against Poisoning Attacks in Federated Learning. *Computer Security – ESORICS 2021*, pp. 455–475. doi: 10.1007/978-3-030-88418-5_22.

Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D. & Shmatikov, V. (2020). How To Backdoor Federated Learning. *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Barbieri, F., Rossi, M. & Carraro, M. (2022a). Decentralized Object Classification Using Federated Learning. *Proceedings of the IEEE Intelligent Transportation Systems Conference*, pp. 567–575.

Barbieri, L., Savazzi, S., Brambilla, M. & Nicoli, M. (2022b). Decentralized federated learning for extended sensing in 6G connected vehicles. *Veh. Commun.*, 33(C). doi: 10.1016/j.vehcom.2021.100396.

Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Parcollet, T. & Lane, N. D. (2020). Flower: A Friendly Federated Learning Research Framework. *arXiv preprint arXiv:2007.14390*.

Bhagoji, A. N., Chakraborty, S., Mittal, P. & Calo, S. B. (2019). Analyzing federated learning through an adversarial lens. *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 97, 634–643.

Biggio, B., Nelson, B. & Laskov, P. (2012a). Poisoning attacks against machine learning. *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pp. 1807–1814.

Biggio, B., Nelson, B. & Laskov, P. (2012b, Jun). Poisoning Attacks against Support Vector Machines.

Blanchard, P., El Mhamdi, E. M., Guerraoui, R. & Stainer, J. (2017a). Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. *Advances in Neural Information Processing Systems (NeurIPS)*, 119–129.

Blanchard, P., El Mhamdi, E. M., Guerraoui, R. & Stainer, J. (2017b). Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. *Advances in Neural Information Processing Systems (NeurIPS)*.

Bonawitz, K., Ivanov, V., Kreuter, B. et al. (2017). Practical Secure Aggregation for Privacy-Preserving Machine Learning. *ACM Conference on Computer and Communications Security (CCS)*.

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konečný, J., Mazzocchi, S., McMahan, B., Van Overveldt, T., Petrou, D., Ramage, D. & Roselander, J. (2019). Towards Federated Learning at Scale: System Design. *Proceedings of Machine Learning and Systems*, 1, 374–388. Retrieved from: https://proceedings.mlsys.org/paper_files/paper/2019/file/7b770da633baf74895be22a8807f1a8f-Paper.pdf.

Briggs, C., Fan, Z. & Andras, P. [arXiv:2004.11791 [cs, stat]]. (2020a, May). Federated learning with hierarchical clustering of local updates to improve training on non-IID data. arXiv. Retrieved on 2022-07-27 from: http://arxiv.org/abs/2004.11791.

Briggs, C., Fan, Z. & Andras, P. (2020b). Federated Learning with Hierarchical Clustering. *arXiv preprint arXiv:2004.11791*.

Chaabene, R. B., Amayed, D. & Cheriet, M. [arXiv:2206.04731 [cs]]. (2022, Jun). Leveraging Centric Data Federated Learning Using Blockchain For Integrity Assurance. arXiv. Retrieved on 2022-07-27 from: http://arxiv.org/abs/2206.04731.

Chellapandi, V. P., Yuan, L., Żak, S. H. & Wang, Z. (2023, Sep). A Survey of Federated Learning for Connected and Automated Vehicles. *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2485–2492. doi: 10.1109/ITSC57777.2023.10421974.

Chen, M., Wang, X. & Sun, Y. (2023). HSCS: Honest Score Client Selection for Federated Learning. *arXiv preprint arXiv:2311.05826*.

Chen, M. et al. (2020a). Asynchronous Federated Learning for Resource-Constrained Edge Devices. *IEEE Transactions on Wireless Communications*.

Chen, Y., Wu, Q., Li, Y. & Yang, B. (2020b). Data poisoning attacks in federated learning. *IEEE Security Privacy*, 18(4), 67–74.

CIFAR-10. CIFAR-10 and CIFAR-100 datasets. Retrieved on 2024-10-01 from: https://www.cs.toronto.edu/~kriz/cifar.html.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. (2009, Jun). ImageNet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.

Deng, L. (2012). The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. *IEEE Signal Processing Magazine*, 29(6), 141–142. doi: 10.1109/MSP.2012.2211477. Conference Name: IEEE Signal Processing Magazine.

Dwork, C., McSherry, F., Nissim, K. & Smith, A. (2006). Calibrating Noise to Sensitivity in Private Data Analysis. *Theory of Cryptography Conference (TCC)*.

Fallah, A., Mokhtari, A. & Ozdaglar, A. (2020). Personalized Federated Learning: A Meta-Learning Approach. *Advances in Neural Information Processing Systems (NeurIPS)*.

Fang, M., Cao, X., Jia, J. & Gong, N. (2020). Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. *29th USENIX Security Symposium (USENIX Security 20)*, pp. 1605–1622. Retrieved from: https://www.usenix.org/conference/usenixsecurity20/presentation/fang.

Fereidooni, H., Marchal, S., Miettinen, M., Mirhoseini, A., Möllering, H., Nguyen, T. D., Rieger, P., Sadeghi, A.-R., Schneider, T., Yalame, H. & Zeitouni, S. (2021, May). SAFELearn: Secure Aggregation for private FEderated Learning. *2021 IEEE Security and Privacy Workshops (SPW)*, pp. 56–62. doi: 10.1109/SPW53761.2021.00017.

Fung, C., Yoon, C. J. M. & Beschastnikh, I. (2018). Mitigating Sybils in Federated Learning Poisoning. *arXiv preprint arXiv:1808.04866*.

Fung, C., Yoon, C. J. M. & Beschastnikh, I. (2020). The Limitations of Federated Learning in Sybil Settings. *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, pp. 301–316. Retrieved from: https://www.usenix.org/conference/raid2020/presentation/fung.

Gassais, R., Ezzati-Jivan, N., Fernandez, J. M., Aloise, D. & Dagenais, M. R. (2020). Multi-level host-based intrusion detection system for Internet of things. *Journal of Cloud Computing*, 9(1), 62. doi: 10.1186/s13677-020-00206-6.

Gentry, C. (2009). Fully Homomorphic Encryption Using Ideal Lattices. *ACM Symposium on Theory of Computing (STOC)*.

Geyer, R., Klein, T. & Nabi, M. (2017). Differentially Private Federated Learning. *Advances in Neural Information Processing Systems (NeurIPS) Workshop*.

Ghosh, A., Kumar, H. & Sastry, P. S. (2017). Robust Loss Functions under Label Noise. *AAAI Conference on Artificial Intelligence*.

Gurghian, A., Koduri, T., Bailur, S. V., Carey, K. J. & Murali, V. N. (2016, Jun). DeepLanes: End-To-End Lane Position Estimation Using Deep Neural Networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 38–45. doi: 10.1109/CVPRW.2016.12.

Hard, A., Rao, K. & Mathews, R. (2018). Differential Privacy for Federated Learning in Mobile Applications. *arXiv preprint arXiv:1811.03604*.

Hopkins, M., Reeber, E., Forman, G. & Suermondt, J. [DOI: https://doi.org/10.24432/C53G6X]. (1999). Spambase. Retrieved from: UCIMachineLearningRepository.

Insights, F. B. (2024). Autonomous Vehicle Market Size, Share, Trends | Report [2030]. Retrieved on 2024-12-14 from: https://www.fortunebusinessinsights.com/autonomous-vehicle-market-109045.

Ioulianou, P., Vasilakis, V., Moscholios, I. & Logothetis, M. (2018, Jun). A Signature-based Intrusion Detection System for the Internet of Things. *Information and Communication Technology Form*. Retrieved from: https://eprints.whiterose.ac.uk/133312/.

Jaafar, F., Ameyed, D., Barrak, A. & Cheriet, M. (2021). Identification of Compromised IoT Devices: Combined Approach Based on Energy Consumption and Network Traffic Analysis. *2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*, pp. 514–523. doi: 10.1109/QRS54544.2021.00062.

Jebreel, N., Blanco-Justicia, A., Sánchez, D. & Domingo-Ferrer, J. (2020). Efficient Detection of Byzantine Attacks in Federated Learning Using Last Layer Biases. *Modeling Decisions for Artificial Intelligence*, pp. 154–165. doi: 10.1007/978-3-030-57524-3_13.

Kadhe, S., Rajaraman, N., Koyluoglu, O. O. & Ramchandran, K. [arXiv:2009.11248 [cs, math, stat]]. (2020, Sep). FastSecAgg: Scalable Secure Aggregation for Privacy-Preserving Federated Learning. arXiv. Retrieved on 2022-09-07 from: http://arxiv.org/abs/2009.11248.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawit, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., El Rouayheb, S., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konecný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Qi, H., Ramage, D., Raskar, R., Raykova, M., Song, D., Song, W., Stich, S. U., Sun, Z., Theertha Suresh, A., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H. & Zhao, S. (2021).

Kang, J. et al. (2019). Incentive Mechanism for Reliable Federated Learning. *IEEE Transactions on Mobile Computing*.

Karimireddy, S. P., Jaggi, M., Kale, S., Mohri, M., Reddi, S., Stich, S. U. & Suresh, A. T. (2021). Breaking the centralized barrier for cross-device federated learning. *Advances in Neural Information Processing Systems*, 34, 28663–28676. Retrieved from: https://proceedings.neurips.cc/paper/2021/hash/f0e6be4ce76-ccfa73c5a540d992d0756-Abstract.html.

Koetsier, C., Fiosina, J., Gremmel, J. N., Müller, J. P., Woisetschläger, D. M. & Sester, M. (2022a). Detection of anomalous vehicle trajectories using federated learning. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, 4, 100013. doi: 10.1016/j.ophoto.2022.100013.

Koetsier, M., Zhang, Y. & Liu, W. (2022b). Federated Learning for Abnormal Trajectory Detection in Traffic. *Springer Autonomous Vehicles Research*, 5(3), 214–230.

Kolias, C., Kambourakis, G., Stavrou, A. & Voas, J. (2017). DDoS in the IoT: Mirai and other botnets. *Computer*, 50(7), 80–84.

Konečný, J., McMahan, B. & Ramage, D. (2015). Federated Optimization:Distributed Optimization Beyond the Datacenter. doi: 10.48550/arXiv.1511.03575.

Kong, X., Wang, K., Hou, M., Xinyu, H., Shen, G., Xin, C. & Xia, F. (2021a). A Federated Learning-Based License Plate Recognition Scheme for 5G-Enabled Internet of Vehicles. *IEEE Transactions on Industrial Informatics*, PP, 1–1. doi: 10.1109/TII.2021.3067324.

Kong, X., Li, M. & Chen, J. (2021b). Federated Learning for Traffic Sign Recognition in Intelligent Transportation Systems. *Springer ITS Journal*, 19(4), 324–338.

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.

Li, J., Lyu, L., Liu, X., Zhang, X. & Lyu, X. [arXiv:2012.06150 [cs, math]]. (2021a). FLEAM: A Federated Learning Empowered Architecture to Mitigate DDoS in Industrial IoT. arXiv. Retrieved on 2022-07-27 from: http://arxiv.org/abs/2012.06150.

Li, J., Xu, H., Tang, Z. et al. (2022). FL-Defender: Enhancing Federated Learning Robustness through Gradient Reweighting and PCA. *arXiv preprint arXiv:2207.00872*.

Li, Q., Zhang, T. & Wang, F. (2023a). Kernel density estimation for security in federated learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Li, Q., Wang, X., Wang, F. & Wang, C. (2023b). A Label Flipping Attack on Machine Learning Model and Its Defense Mechanism (pp. 490–506). doi: 10.1007/978-3-031-22677-9_26.

Li, S., Ngai, E., Ye, F. & Voigt, T. (2021b). *Auto-weighted Robust Federated Learning with Corrupted Data Sources*. doi: 10.48550/arXiv.2101.05880.

Li, T., Sahu, A. K., Talwalkar, A. & Smith, V. (2020). Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine*, 37(3), 50–60. doi: 10.1109/MSP.2020.2975749. Conference Name: IEEE Signal Processing Magazine.

Li, X., Qu, Z., Zhao, S., Tang, B., Lu, Z. & Liu, Y. (2023c). LoMar: A Local Defense Against Poisoning Attack on Federated Learning. *IEEE Transactions on Dependable and Secure Computing*, 20(1), 437–450. doi: 10.1109/TDSC.2021.3135422. Conference Name: IEEE Transactions on Dependable and Secure Computing.

Li, Z., Gao, Y. & Chen, Y. (2010). HiFIND: A high-speed flow-level intrusion detection approach with DoS resiliency. *Computer Networks*, 54(8), 1282–1299. doi: 10.1016/j.comnet.2009.10.016.

Liu, S., Fu, Y., Zhao, P., Li, F. & Li, C. (2021a). Autonomous Braking Algorithm for Rear-End Collision via Communication-Efficient Federated Learning. *2021 IEEE Global Communications Conference (GLOBECOM)*, pp. 01–06. doi: 10.1109/GLOBECOM46510.2021.9685298.

Liu, Y. et al. (2020). Client-Edge-Cloud Hierarchical Federated Learning. *IEEE Network*.

Liu, Y. et al. (2021b). Reputation-Based Federated Learning. *IEEE Internet of Things Journal*.

Liu, Y., Chen, R. & Zhang, H. (2021c). Brake Optimization for Autonomous Vehicles Using Federated Learning and Real-Time Friction Estimation. *IEEE Transactions on Vehicular Technology*, 70(2), 1782–1795.

Mahdavinejad, M. S., Rezvan, M., Mohammadamin Barekatain, P. A. & Barnaghi, P. (2018). Machine learning for internet of things data analysis: a survey | Elsevier Enhanced Reader.

Martínez-Díaz, M. & Soriguera, F. (2018). Autonomous vehicles: theoretical and practical challenges. *Transportation Research Procedia*, 33, 275–282. doi: 10.1016/j.trpro.2018.10.103.

McMahan, B., Moore, E., Ramage, D., Hampson, S. & Arcas, B. A. y. (2017a, 20–22 Apr). Communication-Efficient Learning of Deep Networks from Decentralized Data. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 54(Proceedings of Machine Learning Research), 1273–1282. Retrieved from: https://proceedings.mlr.press/v54/mcmahan17a.html.

McMahan, B., Moore, E., Ramage, D., Hampson, S. & y Arcas, B. A. (2017b). Communication-Efficient Learning of Deep Networks from Decentralized Data. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

National Highway Traffic Safety Administration. (2015). Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey. U.S. Department of Transportation.

Nguyen, A., Nguyen, N., Tran, K., Tjiputra, E. & Tran, Q. D. (2020). Autonomous Navigation in Complex Environments with Deep Multimodal Fusion Network. pp. 5824–5830. doi: 10.1109/IROS45743.2020.9341494.

Nowroozi, E., Jadalla, N., Ghelichkhani, S. & Jolfaei, A. (2023). *Mitigating Label Flipping Attacks in Malicious URL Detectors Using Ensemble Trees*. doi: 10.13140/RG.2.2.33453.26082.

P, A., Rajendran, G. & Panda, M. (2021). Steering Angle Prediction for Autonomous Driving using Federated Learning: The Impact of Vehicle-To-Everything Communication. pp. 1–7. doi: 10.1109/ICCCNT51525.2021.9580097.

Paudice, A., Muñoz-González, L. & Lupu, E. (2019). Label Sanitization Against Label Flipping Poisoning Attacks (pp. 5–15). doi: 10.1007/978-3-030-13453-2_1.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

Qayyum, A., Janjua, M. U. & Qadir, J. (2022a). Making federated learning robust to adversarial attacks by learning data and model association. *Computers & Security*, 121, 102827. doi: 10.1016/j.cose.2022.102827.

Qayyum, Z., Wu, L. & Kittler, J. (2022b). Latent feature analysis for detecting label-flipping attacks in federated learning. *ACM Transactions on Privacy and Security*, 25(2), 12.

Rahman, S. A., Tout, H., Talhi, C. & Mourad, A. (2020). Internet of Things Intrusion Detection: Centralized, On-Device, or Federated Learning? *IEEE Network*, 34(6), 310–317. doi: 10.1109/MNET.011.2000286. Conference Name: IEEE Network.

Rajasekhar, M. V. & Jaswal, A. K. (2015). Autonomous vehicles: The future of automobiles. *2015 IEEE International Transportation Electrification Conference (ITEC)*, pp. 1–6. doi: 10.1109/ITEC-India.2015.7386874.

Resende, P. A. A. & Drummond, A. C. (2019). A Survey of Random Forest Based Methods for Intrusion Detection Systems. *ACM Computing Surveys*, 51(3), 1–36. doi: 10.1145/3178582.

Rey, V., Sánchez, P. M. S., Celdrán, A. H., Bovet, G. & Jaggi, M. (2022). Federated Learning for Malware Detection in IoT Devices. *Computer Networks*, 204, 108693. doi: 10.1016/j.comnet.2021.108693. arXiv:2104.09994 [cs].

Ring, M., Wunderlich, S., Scheuring, D., Landes, D. & Hotho, A. (2019). A survey of network-based intrusion detection data sets. *Computers & Security*, 86, 147–167. doi: 10.1016/j.cose.2019.06.005.

Rosenfeld, E., Kolter, J. Z. & Harutyunyan, A. (2020a). Certified robustness to label-flipping attacks via randomized smoothing. *Advances in Neural Information Processing Systems*, 33, 21433–21444.

Rosenfeld, E., Winston, E., Ravikumar, P. & Kolter, J. Z. (2020b). Certified robustness to label-flipping attacks via randomized smoothing. *Proceedings of the 37th International Conference on Machine Learning*, 119(ICML'20), 8230–8241.

Sarhan, M., Layeghy, S., Moustafa, N. & Portmann, M. (2021). NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. *Big Data Technologies and Applications*, (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering), 117–135. doi: 10.1007/978-3-030-72802-1_9.

Singh, A. P. & Singh, M. (2014). Analysis of Host-Based and Network-Based Intrusion Detection System. *International Journal of Computer Network and Information Security*, 6, 41-47. doi: 10.5815/ijcnis.2014.08.06.

Smith, T., Zhang, L. & Lee, S. (2024). Adaptive Consensus-Based Model Update Validation for Federated Learning. *arXiv preprint arXiv:2403.04803*.

Smith, V., Chiang, C.-K., Sanjabi, M. & Talwalkar, A. (2017). Federated Multi-Task Learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Steinhardt, J., Koh, P. W. & Liang, P. (2017). Certified Defenses for Data Poisoning Attacks. *Advances in Neural Information Processing Systems (NeurIPS)*.

Su, M.-Y., Yu, G.-J. & Lin, C.-Y. (2009). A real-time network intrusion detection system for large-scale attacks based on an incremental mining approach. *Computers & Security*, 28(5), 301–309. doi: 10.1016/j.cose.2008.12.001.

Sun, J., Lee, K. & Zhao, Y. (2022). Label-flipping attacks in federated learning for autonomous driving. *Journal of Intelligent Transportation Systems*, 26(4), 325–340.

Sun, Y. et al. (2019). Detecting Malicious Clients in Federated Learning. *IEEE Transactions on Information Forensics and Security*.

SunFounder. SunFounder PiCar-X Video Robot Car Kit for Raspberry Pi 5/4/3B+/3B, Python/Blockly (Scratch), Video Courses, Rechargeable Battery (Raspberry Pi NOT Included). Retrieved on 2024-10-01 from: https://www.sunfounder.com/products/picar-x.

Tolpegin, V., Truex, S., Gursoy, M. E. & Liu, L. (2020a). Data Poisoning Attacks Against Federated Learning Systems. 12308, 480–501. doi: 10.1007/978-3-030-58951-6_24.

Tolpegin, V., Truex, S., Gursoy, M. E. & Liu, L. (2020b). Data Poisoning Attacks Against Federated Learning Systems. *European Symposium on Research in Computer Security (ESORICS)*.

Wang, X., Garg, S., Lin, H., Hu, J., Kaddoum, G., Jalil Piran, M. & Hossain, M. S. (2022). Toward Accurate Anomaly Detection in Industrial Internet of Things Using Hierarchical Federated Learning. *IEEE Internet of Things Journal*, 9(10), 7110–7119. doi: 10.1109/JIOT.2021.3074382. Conference Name: IEEE Internet of Things Journal.

Weihong, W. & Jiaoyang, T. (2020). Research on License Plate Recognition Algorithms Based on Deep Learning in Complex Environment. *IEEE Access*, PP, 1–1. doi: 10.1109/ACCESS.2020.2994287.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B. & Wesslén, A. (2012). *Experimentation in Software Engineering*. Germany: Springer. doi: 10.1007/978-3-642-29044-2.

Wu, T., Zhang, F. & Sun, K. (2021a). Cooperative Speed Control for Autonomous Vehicles at Intersections Using Federated Learning. *Proceedings of Transportation Research Part C: Emerging Technologies*, pp. 120–130.

Wu, T., Jiang, M., Han, Y., Yuan, Z., Li, X. & Zhang, L. (2021b). A Traffic-Aware Federated Imitation Learning Framework for Motion Control at Unsignalized Intersections with Internet of Vehicles. *Electronics*, 10(24), 3050. doi: 10.3390/electronics10243050. Number: 24 Publisher: Multidisciplinary Digital Publishing Institute.

Xia, F., Yang, L. T., Wang, L. & Vinel, A. (2012). Internet of Things. *International Journal of Communication Systems*, 25(9), 1101–1102. doi: 10.1002/dac.2417.

Xiao, H., Biggio, B., Nelson, B., Xiao, H. & Eckert, C. (2015). Is Feature Selection Secure Against Training Data Poisoning? *International Joint Conference on Artificial Intelligence (IJCAI)*.

Xie, C., Koyejo, S. & Gupta, I. (2019). Asynchronous Federated Optimization. *Advances in Neural Information Processing Systems (NeurIPS)*.

Xie, R., Li, C., Zhou, X. & Dong, Z. (2023, Jun). Asynchronous Federated Learning for Real-Time Multiple Licence Plate Recognition Through Semantic Communication. *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. doi: 10.1109/ICASSP49357.2023.10097251.

Xu, Y., Li, H. & Wang, T. (2021). Communication-Efficient Federated Learning for Vehicular Networks Using Compression. *IEEE Transactions on Vehicular Technology*, 70(6), 5483–5497.

Yaacoub, J.-P. A., Noura, H. N. & Salman, O. (2023). Security of federated learning with IoT systems: Issues, limitations, challenges, and solutions. *Internet of Things and Cyber-Physical Systems*, 3, 155–179. doi: 10.1016/j.iotcps.2023.04.001.

Yamany, W., Moustafa, N. & Turnbull, B. (2023). OQFL: An Optimized Quantum-Based Federated Learning Framework for Defending Against Adversarial Attacks in Intelligent Transportation Systems. *IEEE Transactions on Intelligent Transportation Systems*, 24(1), 893–903. doi: 10.1109/TITS.2021.3130906. Conference Name: IEEE Transactions on Intelligent Transportation Systems.

Yang, H., Ge, M., Xue, D., Xiang, K., Li, H. & Lu, R. (2024). Gradient Leakage Attacks in Federated Learning: Research Frontiers, Taxonomy, and Future Directions. *IEEE Network*, 38(2), 247–254. doi: 10.1109/MNET.001.2300140. Conference Name: IEEE Network.

Yao, A. C.-C. (1982). Protocols for Secure Computations. *Foundations of Computer Science (FOCS)*.

Yin, D., Chen, Y., Kannan, R. & Bartlett, P. (2018). Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates. *International Conference on Machine Learning (ICML)*, 5650–5659.

Yin, D., Chen, Y., Ramchandran, K. & Bartlett, P. L. (2021). Byzantine-robust distributed learning: Towards optimal statistical rates. *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 139, 6928–6938.

Yu, H., Jiang, R., He, Z., Zheng, Z., Li, L., Liu, R. & Chen, X. (2021). Automated vehicle-involved traffic flow studies: A survey of assumptions, models, speculations, and perspectives. *Transportation Research Part C: Emerging Technologies*, 127, 103101. doi: 10.1016/j.trc.2021.103101.

Zeng, Q., Liu, F. & Wang, L. (2022a). Dynamic Controller Optimization for Autonomous Vehicles Using Federated Learning. *IEEE Intelligent Vehicles Symposium*, 12(6), 1234–1246.

Zeng, T., Semiari, O., Chen, M., Saad, W. & Bennis, M. (2022b). Federated Learning on the Road Autonomous Controller Design for Connected and Autonomous Vehicles. *IEEE Transactions on Wireless Communications*, 21(12), 10407–10423. doi: 10.1109/TWC.2022.3183996. Conference Name: IEEE Transactions on Wireless Communications.

Zhang, H., Bosch, J. & Olsson, H. (2021a). End-to-End Federated Learning for Autonomous Driving Vehicles. pp. 1–8. doi: 10.1109/IJCNN52387.2021.9533808.

Zhang, J., Springenberg, J., Boedecker, J. & Burgard, W. (2016). Deep Reinforcement Learning with Successor Features for Navigation across Similar Environments. doi: 10.48550/arXiv.1612.05533.

Zhang, Y., He, X. & Liu, J. (2021b). Federated Multimodal Learning for Steering Angle Prediction in Autonomous Vehicles. *Proceedings of the IEEE Transactions on Vehicular Technology*, pp. 1020–1035.

Zhou, H., Zheng, Y., Huang, H., Shu, J. & Jia, X. (2023). Toward Robust Hierarchical Federated Learning in Internet of Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 24(5), 5600–5614. doi: 10.1109/TITS.2023.3243003. Conference Name: IEEE Transactions on Intelligent Transportation Systems.

Zhou, X., Wang, Y. & Liu, Z. (2022a). LFighter: Mitigating Label Flipping Attacks in Federated Learning via Gradient Clustering. *arXiv preprint arXiv:2207.01982*.

Zhou, X., Ke, R., Cui, Z., Liu, Q. & Qian, W. (2022b). STFL:Spatio-temporal Federated Learning for Vehicle Trajectory Prediction. *2022 IEEE 2nd International Conference on Digital Twins and Parallel Intelligence (DTPI)*, pp. 1–6. doi: 10.1109/DTPI55838.2022.9998967.

Zhu, G., Li, D., Gu, H., Han, Y., Yao, Y., Fan, L. & Yang, Q. (2024). Evaluating Membership Inference Attacks and Defenses in Federated Learning. doi: 10.48550/ARXIV.2402.06289. Publisher: arXiv Version Number: 1.

Zhuang, H., Yu, M., Wang, H., Hua, Y., Li, J. & Yuan, X. [arXiv:2308.04466 [cs]]. (2024, Apr). Backdoor Federated Learning by Poisoning Backdoor-Critical Layers. arXiv. Retrieved on 2024-10-01 from: http://arxiv.org/abs/2308.04466.

Özgür, A. & Erdem, H. (2016). *A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015*. Retrieved on 2021-10-11 from: https://peerj.com/preprints/1954v1.