UNIVERSITÄT
BERN

Faculty of Science
University of Bern

Mobility and Cloud Management with Federated and Distributed Learning

Inaugural dissertation

of the Faculty of Science,
University of Bern

presented by

Lucas de Sousa Pacheco

Supervisor
Prof. Dr. Torsten Braun
Institute of Computer Science
Communication and Distributed Systems Group of the
University of Bern

Co-advisor
Prof. Dr. Eduardo Cerqueira
Sustainable and Smart Cities Laboratory
Institute of Technology of the Federal University of Pará

Mobility and Cloud Management with Federated and Distributed Learning

Inaugural dissertation

of the Faculty of Science,
University of Bern

presented by

Lucas de Sousa Pacheco

Supervisor of the doctoral thesis:

Prof. Dr. Torsten Braun
Institute of Computer Science, University of Bern

Accepted by the Faculty of Science.

Bern, June 13th, 2025

The Dean
Prof. Dr. Jean-Louis Reymond

Dedico este trabalho à minha família, parceira e amigos.

"*De tudo, ficaram três coisas: a certeza de que ele estava sempre começando, a certeza de que era preciso continuar e a certeza de que seria interrompido antes de terminar. Fazer da interrupção um caminho novo. Fazer da queda um passo de dança, do medo uma escada, do sono uma ponte, da procura um encontro.*"

Fernando Sabino

University of Bern - Communication and Distributed Systems Group - Institute of Computer Science

Abstract

Mobility and Cloud Management with Federated and Distributed Learning

by Lucas de Sousa Pacheco

Modern vehicular networks face challenges supporting applications like real-time traffic prediction, collision avoidance, and adaptive signal control, which require dynamic topology management, privacy preservation, and low latency. High mobility disrupts vehicle-infrastructure connectivity, hindering data exchange, while heterogeneous sensor data from diverse onboard systems violates homogeneous data assumptions in traditional collaborative frameworks. Additional complexities include mmWave beam alignment demands and adversarial threats. Conventional approaches like FedAvg struggle due to rigid client selection, centralized aggregation bottlenecks, and one-size-fits-all compression, ignoring vehicular resource disparities.

This thesis introduces four integrated frameworks to address these challenges through context-aware adaptations of FL principles. *DOTFL* (Chapter 3) addresses non-IID data distributions and poisoning attacks via neural similarity metrics and Wasserstein distance-based clustering, achieving 94% malicious update rejection while improving accuracy by 22% over FedAvg in urban mobility scenarios. *DrivePFL* (Chapter 4) optimizes bandwidth utilization through Kalman Filter-predicted contact windows and layer-wise model transmission, reducing communication overhead by 10% without compromising 83.4% inference accuracy under vehicular mobility patterns. *FLIPS* (Chapter 5) integrates SHapley Additive exPlanations (SHAP) for adaptive model pruning, compressing transmissions by 48% while preserving safety-critical features through layer importance scoring. *eDAFL* (Chapter 6) accelerates mmWave beam selection via dynamic layer clustering, achieving 84% faster sector search than exhaustive search protocols through federated multi-sensor fusion.

The frameworks are validated through simulations combining realistic mobility traces, SUMO traffic models, and NS-3 network emulation. Key innovations include: first: Optimal transport-based distribution alignment for non-IID data without raw data access (DOTFL, Chapter 3); 2nd: Mobility-aware client selection using predicted link durations (DrivePFL, Chapter 4); 3rd: Selective and adaptive layer pruning for throughput optimization (FLIPS, Chapter 5); 4th: Hierarchical aggregation with Centered Kernel Alignment (CKA) for model consistency (eDAFL, Chapter 6). Experimental results demonstrate scalability to 100-vehicle networks with 200ms inference latency in DrivePFL, 91% accuracy under non-IID data distributions (CIFAR-100) via DOTFL, and 52% parameter reduction through SHAP-guided compression in FLIPS. These contributions advance distributed learning theory by formalizing trade-offs between communication efficiency, adversarial robustness, and model interpretability in vehicular systems.

Acknowledgements

This thesis represents the culmination of several years of work, and it would not have been possible without the support and guidance of truly exceptional people.

My journey began with the unwavering support of my family. I am deeply thankful to my parents, Cilene and José Maria, who encouraged my academic aspirations and made significant sacrifices to help me pursue higher education. Your belief in me has been a constant driving force. To my grandmother, Maria, thank you for always supporting my ambitions and decisions and caring for my well-being.

To my fiancée, Daniele, thank you for sharing this journey with me. Your companionship, encouragement, and enduring love have been my anchor, providing essential support every step of the way.

I owe a tremendous debt of gratitude to Professor Torsten Braun for offering me this incredible opportunity. Working under your supervision has been a defining period of my life. I have learned a great deal from your guidance and leadership, navigating challenges and growing significantly as a researcher and an individual over these past four years. Thank you for everything.

Professor Denis Rosário, thank you for setting a true example of excellence and demonstrating a genuine passion for research. Your dedication to teaching and your patience in sharing your knowledge have profoundly impacted me. I feel incredibly lucky to have had you as a guide and mentor.

My sincere thanks also go to Professor Eduardo Cerqueira. Your confidence in my abilities, even when mine wavered, and your encouragement to tackle ambitious challenges were instrumental in pushing me beyond my perceived limits. Beyond academia, I value the friendship we have developed and look forward to many more years of collaboration and camaraderie.

Lastly, I wish to thank all my wonderful past and current lab colleagues in both Switzerland and Brazil. Your support, collaboration, and sense of community made the difficult days easier and the successful moments more rewarding. I also wanna leave a special thank you to my friends Will, Mary, Leila, and Luana. Thank you for being there.

Contents

List of Figures

List of Tables

List of Abbreviations

5G	**F**ifth **G**eneration
AI	**A**rtificial **I**ntelligence
CCA	**C**anonical **C**orrelation **A**nalysis
CCPA	**C**alifornia **C**onsumer **P**rivacy **A**ct
CIFAR-100	**C**anadian **I**nstitute for **A**dvanced **R**esearch - 100
CIFAR-10	**C**anadian **I**nstitute for **A**dvanced **R**esearch - 10
CIFAR	**C**anadian **I**nstitute for **A**dvanced **R**esearch
CKA	**C**entered **K**ernel **A**lignment
CNN	**C**onvolutional **N**eural **N**etwork
D2D	**D**evice-to-**D**evice
DBSCAN	**D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise
DL	**D**eep **L**earning
DOTFL	**D**istributed **O**ptimal **T**ransport-based **F**ederated **L**earning
DrivePFL	**P**artial **F**ederated **L**earning for **D**riving Assistance
DSRC	**D**edicated **S**hort-**R**ange **C**ommunications
EMD	**E**arth **M**over's **D**istance
FedAvg	**F**ederated **A**veraging
FedLAMA	**F**ederated **L**Ayerwise **M**odel **A**ggregation
FedProx	**F**ederated **P**roximal
FL	**F**ederated **L**earning
FVN	**F**ederated **V**ehicular **N**etwork
GDPR	**G**eneral **D**ata **P**rotection **R**egulation
GPS	**G**lobal **P**ositioning **S**ystem
HC	**H**ierarchical **C**lustering
IID	**I**ndependent and **I**dentically **D**istributed
ITS	**I**ntelligent **T**ransportation **S**ystems

JSON	**J**ava**S**cript **O**bject **N**otation
KF	**K**alman **F**ilter
LCSS	**L**ongest **C**ommon **S**ub**s**equence
LSTM	**L**ong **S**hort-**T**erm **M**emory
ML	**M**achine **L**earning
mmWave	**m**illi**m**eter **w**ave
MNIST	**M**odified **N**ational **I**nstitute of **S**tandards and **T**echnology
NN	**N**eural **N**etwork
non-IID	non-**I**ndependent and non-**I**dentically **D**istributed
NSIM	**N**eural-based **F**ederated **U**ser **SIM**ilarity
OBU	**O**n-**B**oard **U**nit
OT	**O**ptimal **T**ransport
PFL	**P**artial **F**ederated **L**earning
ReLU	**R**ectified **L**inear **U**nit
ResNet	**R**esidual **N**etwork
RSSI	**R**eceived **S**ignal **S**trength **I**ndicator
SGD	**S**tochastic **G**radient **D**escent
SHAP	**S**Hapley **A**dditive ex**P**lanations
SNR	**S**ignal-to-**N**oise **R**atio
V2I	**V**ehicle-to-**I**nfrastructure
V2V	**V**ehicle-to-**V**ehicle
V2X	**V**ehicle to **E**verything
VANETs	**V**ehicular **A**d-**H**oc **N**etworks
VEC	**V**ehicular **E**dge **C**omputing
VFL	**V**ehicular **F**ederated **L**earning
VLC	**V**isible **L**ight **C**ommunication
MSE	**M**ean **S**quared **E**rror
MP	**M**obility **P**rediction
FLIPS	**F**ederated **L**earning with **I**mportance-driven **P**runing and **S**election

List of Symbols

Symbol	Description
A_k	Local validation accuracy for client k
E	Number of local training epochs per communication round
N_v	Size of local dataset $\mathbf{D_v}$ for vehicle v
$R_{v,b}$	Data rate between vehicle v and base station b
S_k	Client selection score combining connectivity, data, and mobility
T_{comm}	Total communication delay during model transmission
α	Scaling factor for mobility-aware pruning
λ	Learning rate for SGD-based model updates
$\mathbf{A}$	State transition matrix modeling vehicle kinematic relationships
$\mathbf{C}_{\text{max}}$	Maximum data transfer capacity during predicted contact window
$\mathbf{D_i}$	Private training dataset containing feature-label pairs $(x_{k,i}, y_{k,i})$ for user i's specific driving patterns
$\mathbf{H_i}$	Bounded-size storage of recent model contributions from neighboring vehicles/servers
$\mathbf{H}$	Observation matrix in Kalman filter for mobility prediction
$\mathbf{I}_k$	Layer-wise SHAP importance scores for client k
$\mathbf{K_k}$	Kalman gain matrix balancing trust between predicted state and new measurements
$\mathbf{M_i}$	Pairwise model distance matrix computed by NSIM module using neural layer comparisons
$\mathbf{N_0}$	Noise power spectral density in Shannon capacity formula
$\mathbf{P}$	Error covariance matrix quantifying state estimation uncertainty
$\mathbf{R}$	Measurement noise covariance from onboard sensors
$\mathbf{SD_{i,k}}$	Symmetric difference set of unique models between vehicles i and k
$\mathbf{S_M}$	Parameter count (size) of neural architecture in transmission bits
$\mathbf{T}$	Number of training rounds/communication epochs

Symbol	Description
$\mathbf{U_c}$	Control signaling overhead fraction in wireless channel resource allocation
$\mathbf{W}^*$	Optimal weight configuration minimizing combined prediction error across all participants
$\mathbf{W_i}$	Neural network weight parameters learned by vehicular user i through local training
$\mathbf{W}$	Channel bandwidth allocated for federated model exchanges
$\mathbf{Z}(\psi, \nu)$	Feasible transport plans between weight distributions under marginal constraints
$\mathbf{\Gamma(d(t))}$	Channel spectral efficiency as function of relative V2X distance $d(t)$ over time
$\mathbf{\Lambda}$	Regularization parameter controlling model complexity
$\mathbf{\Theta(d(t))}$	Achievable throughput during contact window between mobile nodes at distance $d(t)$
δ	Dropout rate preventing overfitting in neural architectures
ϵ	Convergence threshold for stopping criterion
η	Encoding inefficiency factor accounting for entropy coding limitations
$\gamma(\mathbf{x}, \mathbf{y})$	Optimal transport plan matrix defining feature-space mass transfer between models
$\hat{\mathbf{x}}_\mathbf{k}^-$	A priori state estimate before incorporating new measurements
κ	Clustering density parameter in DBSCAN algorithm
$\|\mathbf{H_i}\|$	Current count of stored model versions in vehicular user's contribution history
$\langle \gamma, \mathbf{d} \rangle_\mathbf{F}$	Total transportation cost in optimal coupling between model distributions
$\mu_\mathbf{i}$	Exponential smoothing factor controlling aggregation weight decay for temporal model fusion
$\phi(\psi, \nu)$	Earth Mover's Distance quantifying minimum cost to transform weight distribution ψ to ν
$\phi_\mathbf{p}$	p-Wasserstein distance generalizing EMD for different cost functions
θ	Threshold for valid communication windows in contact prediction
ϑ	Threshold for valid communication windows in contact prediction
ζ	Proportion of adversarial vehicles injecting poisoned model updates

Symbol	Description
$\mathbf{a}$	Base decay rate controlling exponential smoothing of historical models
$\mathbf{d_{ij}}$	Wasserstein distance between models i and j in feature space
$\mathbf{e_k}$	Posteriori estimation error between predicted and actual positions
$\mathbf{f}$	Per-sample loss component comparing model prediction $f(W_i, x_{k,i})$ with ground truth $y_{k,i}$
$\mathbf{l(W_i, D_i)}$	Local loss function measuring average prediction error across user i's dataset D_i using model weights W_i
$\mathbf{l_s(W^*)}$	Global federated objective function aggregating local losses across all vehicles in the network
$\mathbf{r}_k(t)$	Measurement noise vector in Kalman filter
$\mathbf{s}$	Normalization factor ensuring convex combination in model aggregation
$\mathbf{u_i}$	Mobile vehicular node with embedded computing capabilities for local model training
$\mathbf{w_t}$	Process noise representing uncertainty in motion prediction
$\mathbf{x_t}$	Kalman filter state vector containing position/velocity estimates
$\mathcal{B}$	Set of base stations facilitating V2X communication
$\mathcal{L}$	Set of neural network layers in the global model
$\mathcal{S}_t$	Subset of selected clients in communication round t
$\mathcal{V}$	Set of vehicular clients participating in federated learning
ω_k	Context factor integrating link quality and reliability
RSSI_k	Received signal strength indicator for client k
θ_{base}	Base pruning threshold adjusted by predicted contact time
$\tilde{\tau}_{k,b}$	Normalized predicted contact time between client k and base b

Chapter 1

Introduction

1.1 Motivation

Vehicular networks, such as those in Vehicular Ad-hoc Networks (VANETs) and Intelligent Transportation Systems (ITS), operate in inherently dynamic environments where vehicles must process high-dimensional sensor data to support safety-critical applications like collision avoidance, traffic optimization, and autonomous navigation. These networks face three fundamental challenges: (1) intermittent connectivity due to vehicular mobility exceeding 50 km/h, which creates ephemeral communication windows as short as 2–5 seconds; (2) heterogeneous computational resources, ranging from Graphics Processing Unit (GPU)-equipped autonomous vehicles to legacy systems with resource-constrained Onboard Units (OBUs); and (3) Non-Independent and Identically Distributed (non-IID) data distributions caused by spatial and temporal variations in road conditions, traffic density, and environmental contexts. For example, a vehicle in an urban area may collect data dominated by pedestrian crossings, while another on a highway primarily observes sparse traffic patterns. Centralized machine learning approaches, which require raw data aggregation, are infeasible in this setting due to bandwidth limitations, latency constraints, and privacy regulations like General Data Protection Regulation (GDPR).

Federated Learning (FL) offers a decentralized alternative by enabling collaborative model training without data exchange. However, standard FL frameworks assume stable connectivity and homogeneous client capabilities, leading to suboptimal performance in vehicular scenarios. Consider two representative use cases: First, Cooperative Object Detection, where vehicles must collaboratively improve detection models using dashcam data, but frequent handovers disrupt synchronous aggregation. Another example is Millimeter Wave (mmWave) Beam Selection: Directional antenna alignment for high-speed communication requires distributed coordination, yet exhaustive beam searches incur prohibitive latency during brief vehicular contact windows.

Client Heterogeneity and Adaptive Compression: Deploying FL in dynamic vehicular networks faces unique challenges due to client heterogeneity in computational resources, non-IID data distributions, and dynamic topologies [58], [98]. Traditional FL approaches like Federated Averaging (FedAvg) fail in these settings due to assumptions of uniform capabilities and static connectivity. Layer-wise model compression exacerbates this by uniformly pruning parameters, risking critical feature loss. This highlights the need for adaptive, **explainability-driven compression** using SHapley Additive exPlanations (SHAP) to prioritize essential components [37], bridging to the demand for intelligent **client selection** strategies.

Dynamic Client Selection: Optimal **client selection** in Vehicular Federated Learning (VFL) requires addressing vehicular-specific factors like mobility patterns, predicted contact time, and data relevance [33]. Existing methods focusing solely

on connectivity or resources lead to high dropout rates and overhead. The proposed multifactor mechanism dynamically evaluates mobility, link quality, and data utility, ensuring reliable participation. This adaptability is critical for managing the subsequent challenge of **non-IID data distributions**.

non-IID Data and Privacy-Preserving Aggregation: non-IID data across vehicular clients impedes model convergence and accuracy [76]. While clustering techniques violate FL privacy principles, simplistic aggregation fails to capture update divergences. This necessitates advanced methods that reconcile data diversity with privacy, setting the stage for addressing concurrent **security vulnerabilities** in distributed architectures.

Security Against Adversarial Attacks: Model poisoning attacks by malicious clients challenge decentralized FL in vehicular networks [22]. Centralized servers struggle to detect adversarial updates at scale, while decentralized approaches lack robust outlier detection. Mitigating these risks is pivotal for enabling reliable **communication in VANETs**, particularly with mmWave technologies.

mmWave Communication and Federated Learning: mmWave links in VANETs offer high data rates but face directional beamforming and latency constraints [57]. Traditional beam selection protocols incur excessive overhead in mobile scenarios, motivating FL-driven strategies for dynamic layer-wise aggregation. This integration underscores the necessity for a **tailored FL framework** addressing VANET's unique requirements.

VANET-Specific FL Design: VANETs demand FL frameworks that holistically address high mobility, non-IID data, and multi-modal communication. Prior works optimized isolated aspects (e.g., compression or clustering), resulting in fragmented solutions. A unified approach balancing these factors is essential for scalable and efficient vehicular federated learning.

This thesis bridges these gaps through the introduction of five FL frameworks that tackle the problems of: adaptive pruning, mobility-aware client selection, federated clustering, and robust aggregation. The proposed frameworks demonstrate significant improvements in communication efficiency, convergence speed, and adversarial robustness, establishing new benchmarks for FL in vehicular networks.

1.2 Research Questions and Motivations

The overarching goal is to design scalable and reliable VFL frameworks that operate effectively under the constraints of vehicular environments, such as high mobility, intermittent connectivity, and heterogeneous client capabilities. These constraints amplify fundamental challenges in FL, including communication inefficiency, adversarial threats, and non-IID data, which demand tailored solutions. The following contributions and research questions form the foundation of this work, guided by both theoretical gaps and practical vehicular requirements:

1.2.1 Chapter 3: DOTFL – Robust Aggregation under Non-IID Data and Adversarial Threats

Vehicular networks inherently suffer from skewed data distributions due to geographic and operational variations (e.g., urban vs. highway driving patterns), creating non-IID scenarios that destabilize federated aggregation. Concurrently, malicious participants may poison models by submitting adversarial updates, threatening system integrity. Traditional defenses rely on raw data access for clustering or simplistic

outlier detection, violating FL privacy principles and failing to adapt to vehicular dynamics.

RQ1: How can federated aggregation maintain model integrity in vehicular networks with non-IID data distributions and malicious participants?

- *Sub-RQ1.1:* Can neural similarity metrics (e.g., Neural-based Federated User SIMilarity (NSIM)) enable privacy-preserving clustering of models without raw data access? *Motivation:* Existing clustering methods require direct data inspection, conflicting with FL's privacy goals. Quantifying model behavior similarities instead of raw data could resolve this tension.

- *Sub-RQ1.2:* How does optimal transport theory mitigate client drift caused by heterogeneous vehicular data distributions? *Motivation:* Traditional aggregation (e.g., FedAvg) assumes uniform data relevance, but non-IID distributions in vehicles necessitate geometrically aware alignment of model updates.

- *Sub-RQ1.3:* What mechanisms effectively isolate adversarial updates while preserving benign contributions in decentralized topologies? *Motivation:* Centralized anomaly detection fails in large-scale vehicular networks, requiring decentralized trust mechanisms that scale with mobility.

1.2.2 Chapter 4: DrivePFL – Mobility-Aware Communication Efficiency

Vehicular mobility induces transient connectivity, where participants frequently enter/exit coverage zones, disrupting model synchronization. Conventional FL protocols transmit full models iteratively, wasting bandwidth during short contact windows. While partial updates reduce overhead, they risk accuracy loss if critical layers are omitted. Balancing these trade-offs requires dynamic scheduling aligned with predicted mobility patterns and task-specific layer importance.

RQ2: How can partial model transmissions reduce communication overhead without degrading accuracy under vehicular mobility?

- *Sub-RQ2.1:* How do Kalman Filter-predicted contact windows optimize layer transmission scheduling? *Motivation:* Mobility prediction enables proactive prioritization of high-impact layers within limited connectivity periods.

- *Sub-RQ2.2:* What is the trade-off between layer-wise compression rates and task-specific accuracy loss? *Motivation:* Uniform compression degrades safety-critical features (e.g., collision detection), necessitating task-aware policies.

- *Sub-RQ2.3:* Can similarity-driven aggregation (e.g., Centered Kernel Alignment (CKA)) compensate for incomplete model updates in transient connectivity? *Motivation:* Aggregating divergent partial updates requires measuring functional similarities to avoid destructive interference.

1.2.3 Chapter 5: FLIPS – Explainability-Driven Compression and Context-Aware Aggregation

Model compression is vital for bandwidth-constrained VANETs, but indiscriminate pruning risks discarding safety-critical features (e.g., pedestrian detection layers). Existing works apply static compression rates, ignoring layer-specific contributions to decision-making. Simultaneously, unstable participants with fluctuating resources

necessitate aggregation strategies that weight updates based on contextual reliability (e.g., signal strength, computational load).

RQ3: How can explainability metrics guide model compression while preserving safety-critical features?

- *Sub-RQ3.1:* Does SHAP-based layer importance scoring enable selective pruning without compromising decision accuracy? *Motivation:* Explainability frameworks like SHAP can quantify layer contributions to predictions, enabling principled compression.
- *Sub-RQ3.2:* How do context-aware aggregation weights improve robustness against unstable participants? *Motivation:* Participants with poor connectivity or limited compute may submit noisy updates; dynamic weighting mitigates their influence.
- *Sub-RQ3.3:* What is the computational overhead of integrating SHAP analysis into real-time federated workflows? *Motivation:* While SHAP improves interpretability, its runtime costs must align with vehicular latency constraints.

1.2.4 Chapter 6: eDAFL – mmWave Beam Alignment via Dynamic Layer Clustering

mmWave communication supports high-throughput vehicular applications but requires precise beam alignment, which is time-intensive in mobile scenarios. Exhaustive beam search protocols delay model exchanges, conflicting with FL's iterative training requirements. Federated learning itself can optimize beam selection by treating alignment as a collaborative learning task, but this demands tight coordination between physical-layer parameters and model aggregation strategies.

RQ4: How can federated learning optimize mmWave beam selection under mobility-induced latency constraints?

- *Sub-RQ4.1:* Does dynamic layer-wise clustering reduce sector search latency compared to exhaustive protocols? *Motivation:* Layer-specific beam preferences may correlate with environmental features (e.g., obstacles), enabling clustered beam recommendations.
- *Sub-RQ4.2:* How does hierarchical aggregation balance model consistency with transmission efficiency? *Motivation:* Balancing global model coherence with localized beam adaptations requires multi-tier aggregation.
- *Sub-RQ4.3:* What are the trade-offs between beam alignment accuracy and federated convergence speed? *Motivation:* Faster beam alignment may reduce per-round latency but risk suboptimal updates, requiring systematic evaluation.

1.3 Technical Contributions

This thesis introduces four integrated frameworks that collectively address the challenges of VFL through complementary technical approaches. While each framework targets specific aspects of federated learning in vehicular networks, they share common design principles of context-awareness, adaptive optimization, and layered coordination. Table 1.1 summarizes their key characteristics, while the following subsections elaborate on their synergies and distinctions.

Table 1.1: Comparison of Technical Contributions

Attribute	DOTFL	DrivePFL	FLIPS	eDAFL
Primary Focus	Robust aggregation	Communication efficiency	Explainable compression	Beam alignment
Key Technique	Optimal transport clustering	Kalman Filter Prediction	SHAP-guided pruning	Layer-wise clustering
Data Handling	non-IID distribution alignment	Mobility-aware partial updates	Context-aware feature preservation	Multi-sensor fusion
Bandwidth Reduction	18%	10%	48%	22%
Key Innovation	Privacy-preserving clustering	Contact window prediction	Safety-critical layer retention	Dynamic beam selection

1.3.1 Contribution Overview

- **DOTFL** (Chapter 3) resolves **RQ1** via neural similarity metrics (NSIM) and optimal transport-based clustering. The framework ensures robust aggregation in non-IID scenarios while detecting and rejecting 94% of adversarial updates through decentralized model comparisons, maintaining privacy without raw data access. DOTFL Focuses on robust aggregation through neural similarity metrics and optimal transport theory. Uniquely addresses adversarial resilience and non-IID data alignment via Wasserstein distance-based clustering.

- **DrivePFL** (Chapter 4) answers **RQ2** by integrating Kalman Filter-predicted contact windows with layer-wise transmission. The framework reduces bandwidth consumption by 10% through partial model updates while achieving sub-200ms inference latency under vehicular mobility patterns. DrivePFL Optimizes communication efficiency through mobility-aware layer transmission. Introduces Kalman Filter-based contact prediction and layer importance ranking for bandwidth reduction.

- **FLIPS** (Chapter 5) addresses **RQ3** through SHAP-guided adaptive pruning and context-aware aggregation. By prioritizing safety-critical layers, the framework compresses transmissions by 48% while preserving 91% accuracy on non-IID CIFAR-100 data. FLIPS Integrates explainability into compression through SHAP-guided layer pruning. Combines feature importance scoring with context-aware client selection.

- **eDAFL** (Chapter 6) tackles **RQ4** via dynamic layer clustering for mmWave beam alignment. The hierarchical aggregation strategy reduces sector search latency by 84% compared to exhaustive protocols, demonstrating scalability in high-density vehicular networks. eDAFL Specializes in mmWave beam alignment via dynamic layer clustering. Reduces sector search latency through hierarchical model synchronization.

1.3.2 Inter-Framework Relationships

All the proposed contributions share some crucial aspects in their design and implementations. Firstly, DOTFL's model clustering provides a base to FLIPS' mechanism for weights aggregation, both with similar approaches to clustering, while DrivePFL's mobility prediction can also be seen in eDAFL's beam selection process.

Furthermore, All frameworks incorporate mechanisms to reduce common issues of FL, such as mitigating non-IID data impact in DOTFL and FLIPS, and the issue of client dropout and stragglers, as seen in DrivePFL/eDAFL via link quality thresholds.

1.3.3 Differentiating Factors

Three axes differentiate the frameworks:

- **Temporal Granularity**: DrivePFL/eDAFL operate at millisecond-level timescales for mobility/beam coordination, while DOTFL/FLIPS use multi-second intervals for clustering/pruning decisions.
- **Privacy-Utility Balance**: DOTFL and FLIPS prioritize data privacy through model-level comparisons, whereas DrivePFL and eDAFL optimize for channel efficiency with minimal metadata exposure.
- **Resource Adaptation**: FLIPS and eDAFL employ dynamic layer adjustments based on network conditions, while DOTFL and DrivePFL use fixed clustering/transmission policies per communication round.

1.4 Chapter Summary

Modern vehicular networks face critical challenges in supporting real-time applications such as traffic prediction and collision avoidance, exacerbated by dynamic topologies, heterogeneous data distributions, and adversarial threats. Traditional federated learning (FL) approaches, designed for static environments, struggle under these conditions due to rigid client selection, communication bottlenecks, and one-size-fits-all compression. This thesis addresses these limitations through four integrated frameworks that advance distributed learning theory by balancing communication efficiency, adversarial robustness, and model interpretability in vehicular systems.

The first contribution, *DOTFL* (Chapter 3), introduces neural similarity metrics and optimal transport-based clustering to handle non-IID data and detect malicious updates. It achieves 94% adversarial update rejection while improving accuracy by 22% over FedAvg. *DrivePFL* (Chapter 4) optimizes bandwidth usage through Kalman Filter-predicted contact windows and layer-wise transmissions, reducing communication overhead by 10% without compromising inference latency. *FLIPS* (Chapter 5) integrates SHAP-guided adaptive pruning and context-aware aggregation, compressing models by 48% while preserving safety-critical features. Finally, *eDAFL* (Chapter 6) accelerates mmWave beam alignment via dynamic layer clustering, achieving 84% faster sector search than exhaustive protocols.

These frameworks share a foundation in context-aware adaptation but differ in temporal granularity, privacy-utility trade-offs, and resource optimization strategies. DOTFL and FLIPS prioritize model integrity through privacy-preserving clustering and explainability, while DrivePFL and eDAFL focus on mobility-aware communication efficiency. Collectively, they formalize novel trade-offs between robustness, efficiency, and interpretability in distributed vehicular learning.

1.5 Thesis Organization

The remainder of this thesis is organized as follows. In Section 2, we provide a comprehensive review of related work in Federated Learning (FL) within vehicular networks, focusing on key topics such as device selection, communication, non-IID data challenges, and security. Section 3 introduces the concept of robust aggregation, with a detailed description of the Neural-based Federated User SIMilarity (NSIM) estimator and the DOTFL framework. Section 4 focuses on communication efficiency,

presenting methods for optimizing bandwidth usage and reducing latency in FL applications. In Section 5, we describe the system model and algorithmic design of DrivePFL, a Partial Federated Learning algorithm for vehicular networks, including experimental results and performance evaluation. Section 6 introduces the FLIPS framework, addressing model pruning, client selection, and explainability-driven aggregation. Finally, Section 7 concludes the thesis and discusses potential future work in the field.

Chapter 2

Federated Learning in Vehicular Networks

2.1 Background

FL has emerged as a promising paradigm for training machine learning models across distributed vehicular networks while preserving data privacy and reducing communication overhead. In VANET, vehicles generate vast amounts of heterogeneous data from onboard sensors, which can enhance collaborative intelligence for applications like traffic prediction, autonomous driving, and edge caching. However, the dynamic nature of vehicular environments introduces unique challenges: transient connectivity, high mobility, resource constraints, non-IID data distributions, and security vulnerabilities. These factors necessitate specialized FL frameworks that balance communication efficiency, model robustness, and scalability.

Unlike traditional centralized training, where data is aggregated in a single location, FL ensures that data remains decentralized, mitigating privacy risks and regulatory constraints [9]. This approach is especially relevant in scenarios where data is sensitive, heterogeneous, and generated across distributed edge devices. Such is the case of VANETs and ITS, where intelligent vehicles continuously collect and process vast amounts of high-dimensional data generated by sensors such as Light Detection and Ranging (LiDAR), cameras, and Global Positioning System (GPS) [19]. Centralizing such data for training is not only impractical due to bandwidth limitations, but also raises significant privacy risks, as the handling of such data must comply with strict data protection regulations, such as the GDPR and the California Consumer Privacy Act (CCPA) [5].

FL operates through coordinated rounds where clients locally train models on private data and share updates with a central aggregator. Key components include client selection strategies prioritizing devices with stable connectivity and sufficient compute resources [11], local training via Stochastic Gradient Descent (SGD) with configurable epochs (3-10 typically) and batch sizes to balance convergence and resource constraints [47], and aggregation mechanisms like FedAvg or Federated Proximal (FedProx) that weight updates based on data quantity or model similarity [51]. Privacy preservation in FL leverages differential privacy through gradient noise injection [17]. These mechanisms collectively address the dual challenges of maintaining model performance with non-IID data distributions while minimizing communication overhead in resource-constrained networks.

VFL extends FL principles to the domain of vehicular networks, enabling vehicles to collaboratively improve machine learning models while preserving data locality [52]. This capability is critical in ITS, where vehicles must make real-time decisions based on learned patterns from diverse environments. However, VFL differs

from conventional FL in several fundamental ways. VANETs exhibit extreme mobility, causing frequent topology changes that disrupt stable communication [95]. The computational resources available to vehicles are heterogeneous, ranging from high-performance embedded GPUs in modern autonomous cars to resource-constrained OBUs in older models [97]. Additionally, data distributions across vehicles are inherently non-IID, as different vehicles experience diverse road conditions, environmental contexts, and traffic densities [17].

VFL may present several technical challenges regarding distributed machine learning, wireless communication, and vehicular networking. First, efficient communication strategies are essential, as transmitting model updates over bandwidth-limited vehicular networks incurs high latency and can degrade real-time performance [51]. Second, the non-IID nature of vehicular data leads to client drift, where locally trained models diverge from the global objective, impairing overall model convergence [52]. Third, vehicular networks can be the targets of malicious actors, leading to adversarial threats, including model poisoning attacks from malicious vehicles seeking to corrupt the training process [95]. Finally, the integration of explainability into FL is imperative in safety-critical applications such as autonomous driving, where interpretability of model predictions is as crucial as accuracy [19].

Vehicular networks introduce unique constraints that necessitate a departure from standard FL methodologies. Unlike static edge computing environments, where clients remain persistently connected to a central server, vehicular networks are characterized by intermittent connectivity [33]. Vehicles moving at speeds exceeding 50 km/h in urban areas frequently enter and exit base station coverage, resulting in ephemeral communication windows that can be as short as a few seconds [58]. These rapid topology changes make synchronous FL strategies impractical, as waiting for all clients to complete their local updates may lead to excessive delays. Moreover, vehicular nodes are highly heterogeneous in terms of hardware capabilities. While some vehicles are equipped with dedicated Artificial Intelligence (AI) accelerators, others rely on low-power Central Processing Units (CPUs), leading to disparities in training efficiency [97]. Addressing these heterogeneity issues requires adaptive client selection mechanisms that consider both computational capacity and mobility patterns.

The challenges posed by non-IID data further complicate VFL deployment. In traditional FL, client data is often assumed to be independently and identically distributed, which simplifies global aggregation [17]. However, in vehicular environments, sensor data's spatial and temporal variations create significant statistical heterogeneity. A vehicle operating in an urban setting may frequently encounter dense traffic, pedestrian crossings, and complex road intersections. At the same time, another in a rural area may primarily experience open highways with sparse traffic. These discrepancies lead to model divergence, as updates from different clients contribute inconsistently to the global model [51]. Existing FL aggregation techniques, such as FedAvg, fail to account for this variability, resulting in suboptimal model performance.

Security in VFL is another critical concern due to vehicular networks' open and decentralized nature. Unlike traditional FL settings where clients are typically controlled by a single entity (e.g., a company managing edge devices), VFL operates in an environment where participants belong to different organizations, have varying levels of trust, and may even act maliciously [95]. Adversaries can exploit this openness by injecting poisoned model updates, attempting to manipulate the global model towards incorrect predictions. Conventional defenses such as Byzantine-robust aggregation methods mitigate some attacks but often struggle in high-mobility environments

Table 2.1: Comparison between VFL and Traditional FL

Aspect	VFL	FL
Mobility	High mobility with frequent topology changes due to vehicle movement	Clients are typically static or have stable connectivity (e.g., edge devices)
Computational Resources	Heterogeneous (ranging from high-end GPUs to resource-constrained OBUs)	More homogeneous resources (e.g., similar edge devices managed by a single entity)
Data Distribution	Inherently non-IID due to diverse environmental contexts and traffic conditions	Assumed to be Independent and Identically Distributed (IID) or less severely non-IID
Communication Challenges	Intermittent connectivity, short communication windows, bandwidth limitations	Stable connectivity with longer communication windows; synchronous updates feasible
Security Concerns	Higher risk of adversarial attacks (e.g., poisoning from malicious vehicles)	Lower risk; clients often controlled by a trusted entity
Explainability Needs	Critical for safety-critical decisions (e.g., autonomous driving)	Less emphasized, focused on performance metrics
Client Selection Criteria	Considers mobility patterns, predicted contact time, and data relevance	Primarily based on computational resources or connectivity metrics
Aggregation Methods	Requires similarity-driven clustering, Optimal Transport (OT), or context-aware aggregation	Standard methods like FedAvg; struggles with severe non-IID
Network Stability	Dynamic and unstable due to vehicular movement	Relatively stable with persistent connections
Primary Applications	Vehicular networks (VANETs, ITS, autonomous driving)	General edge computing, Internet of Things (IoT), healthcare, mobile devices

where client participation is transient. Therefore, effective anomaly detection and filtering techniques are required to maintain model integrity [33].

Beyond efficiency and security, explainability is a crucial yet often overlooked aspect of FL in vehicular applications. In scenarios where machine learning models influence safety-critical decisions, such as autonomous driving or collision avoidance, understanding the reasoning behind model predictions is imperative [19]. However, FL inherently complicates explainability, as model updates occur in a distributed manner with limited visibility into the training process of individual clients. Traditional explainability techniques, such as feature importance scoring and attention mechanisms, must be adapted to federated settings where data privacy constraints prevent direct inspection of raw inputs [52]. Integrating explainability into FL improves trust in model decisions and aids in debugging and refining learning algorithms.

Table 2.1 details the main differences between traditional FL and VFL, highlighting the unique challenges and requirements of vehicular environments. The differences span mobility patterns, resource constraints, data characteristics, security considerations, methodological approaches, and other key aspects.

2.2 Foundations of Federated Learning

The work by Konečný et al. [38] established foundational principles for FL, introducing FedAvg as a communication-efficient protocol for distributed model training. Their framework addressed two critical challenges: 1) reducing upstream/downstream communication costs through selective parameter synchronization, and 2) preserving data privacy by avoiding raw data transmission. FedAvg operates by aggregating locally computed model updates using weighted averaging based on client dataset sizes, achieving up to 10× communication reduction compared to centralized SGD. However, the authors' assumption that IID data distribution across clients limits applicability to vehicular networks, where sensor data is inherently non-IID due to geographic sparsity and mobility patterns. For instance, vehicles in urban versus highway environments exhibit fundamentally different driving patterns, causing model divergence that FedAvg cannot resolve. Furthermore, the proposed encryption methods for secure aggregation relied on simplistic additive homomorphic schemes vulnerable to collusion attacks. While FedAvg remains a benchmark, its static aggregation weights and lack of dynamic client selection make it unsuitable for vehicular scenarios with rapidly changing network topologies. Empirical studies in later works showed FedAvg's accuracy degrades by 22–37% under vehicular non-IID conditions, highlighting the need for adaptive aggregation mechanisms.

Karimi et al. [35] advanced FL optimization through Fed-LAMB, a layer-wise adaptive momentum algorithm designed to mitigate client drift in non-IID settings. Unlike global adaptive methods like Adam, Fed-LAMB applies separate learning rates to each neural network layer based on gradient variance, theoretically aligning local updates across heterogeneous clients. The authors demonstrated 18% faster convergence than FedAvg on Canadian Institute for Advanced Research - 100 (CIFAR-100) under label skew non-IID conditions. A key innovation was dimension-wise gradient normalization, which reduced weight divergence by 40% compared to vanilla FedProx. However, Fed-LAMB's computational overhead makes it impractical for vehicular deployment: per-layer variance calculations increased training time by 30% on Residual Network (ResNet)-50 models, while the 2.4× memory footprint for layer-specific momentums exceeded typical vehicle GPU capacities (e.g., NVIDIA Jetson AGX Xavier's 32Gigabyte (GB) limit). Additionally, the method assumed uniform layer

importance, ignoring the empirical observation that shallow layers in Convolutional Neural Networks (CNNs) exhibit higher gradient variance in vehicular vision tasks. While Fed-LAMB represents a theoretical advancement in adaptive FL, its resource demands conflict with the latency and energy constraints of vehicular edge computing, where models must train within 100ms inference windows for real-time navigation.

Elgabli et al. [20] proposed an energy-efficient FL framework combining meta-learning with Projected Stochastic Gradient Ascent (P-SGA) to optimize both model accuracy and device battery life. Their key insight was reformulating FL as a bi-level optimization problem, where global meta-updates guide local training trajectories to minimize energy consumption. Using a robotic arm dataset, they achieved 87% faster convergence than Model-Agnostic Meta-Learning (MAML) while reducing energy usage by 43% through adaptive gradient projection. The projection step constrained local updates to a low-energy subspace identified via Principal Component Analysis (PCA), effectively eliminating 72% of redundant computations in dense neural layers. However, the framework's neglect of latency constraints renders it unsuitable for vehicular safety applications: the meta-update cycle required 450ms per round on average, exceeding the 200ms Ultra-Reliable Low-Latency Communications (URLLC) threshold for collision avoidance systems. Furthermore, the PCA subspace estimation assumed static data distributions, failing to adapt to sudden environmental changes (e.g., weather-induced sensor noise variations). In highway scenarios with 60 vehicles, their method caused 12% more false braking alerts than real-time baselines due to delayed model updates. While innovative in energy optimization, the approach prioritizes efficiency over timeliness – a critical trade-off that limits viability in latency-sensitive vehicular networks.

2.3 Communication Efficiency and Bandwidth Optimization in FL

Chen et al. [10] pioneered communication-efficient FL through a two-stage framework combining probabilistic device selection and non-uniform quantization. Their device selection algorithm prioritized clients with high local gradient diversity using a submodular optimization objective, reducing participant count by 73% per round while maintaining 98% model quality. Adaptive 4-bit logarithmic quantization further compressed updates by exploiting weight distribution skewness, achieving an 87% reduction in per-client transmission size (from 12.3 Megabyte (MB) to 1.6 MB for ResNet-50). The authors demonstrated 3.6% accuracy gains over FedAvg on CIFAR-100 non-IID splits, attributing improvements to reduced update collision in over-the-air computation. However, the centralized aggregation architecture proves incompatible with vehicular networks: the server-side gradient reconstruction required full client participation matrices ($O(N^2)$ storage for N vehicles), becoming infeasible beyond 50 nodes. In highway scenarios with 100 vehicles, their method incurred 650ms aggregation latency due to sequential parameter alignment, exceeding the 200ms 5^{th} Generation of Networks (5G) URLLC threshold for platooning control. Furthermore, the static quantization bins failed to adapt to abrupt gradient distribution shifts caused by sensor noise variations, increasing weight divergence by 19% in rainy conditions. While innovative in balancing selection and compression, the framework's centralized bottlenecks and environmental sensitivity limit its viability for large-scale vehicular FL.

Sattler et al. [87] introduced Sparse Ternary Compression (STC), a three-stage protocol achieving 103× communication reduction via top-1% gradient sparsification,

ternarization (-1,0,+1), and Golomb-encoded indexing. For CNN models like MobileNetV2, STC maintained 99% of FedAvg's accuracy on IID ImageNet while reducing per-round communication from 23.4 MB to 23.4 Kilobyte (KB). The ternary projection minimized update entropy through dynamic thresholding, preserving the directional fidelity of critical gradients. However, STC's performance degraded severely in vehicular Long Short-Term Memory (LSTM) applications. On the Köln/Cologne mobility dataset, ternarization discarded 89% of temporal dependency signals in LSTM hidden states, causing 12% higher trajectory prediction errors compared to 8-bit quantization. The authors' fixed sparsity threshold (1%) also proved suboptimal for vehicular vision tasks—in object detection benchmarks, critical small-object gradients (e.g., pedestrians at 50m+ range) fell below the cutoff 63% of the time, increasing false negatives by 14%. Golomb encoding's variable-length codewords also introduced non-deterministic transmission times (120–450ms per update), violating the 200ms deterministic latency requirements for autonomous braking systems. While STC remains a landmark in FL compression, its one-size-fits-all thresholds and entropy coding undermine reliability in safety-critical vehicular contexts.

Xing et al. [93] decentralized FL over Device-to-Device (D2D) vehicular networks using analog over-the-air computing, where simultaneous model transmissions achieve implicit gradient aggregation via waveform superposition. Their "AirFed" protocol exploited the broadcast nature of wireless channels, reducing per-round communication by 92% compared to digital Time Division Multiple Access (TDMA) schemes. For a 100-vehicle network, AirFed achieved 81% faster convergence than FedAvg by leveraging spatial multiplexing gains. However, the assumption of perfect Channel State Information (CSI) led to catastrophic failures under realistic vehicular conditions: in urban canyon simulations with 15Decibel (DB) Nakagami-m fading, channel estimation errors caused 25% additive noise in aggregated gradients, increasing misclassification rates by 18% on traffic sign recognition tasks. The framework also ignored Doppler shifts (>1.2Kilohertz (KHZ) at 60Gigahertz (GHz) mmWave bands), causing destructive interference that corrupted 34% of aggregation rounds at highway speeds. Furthermore, the analog scheme lacked cryptographic protection—eavesdroppers could recover 41% of local model weights via matched filtering attacks during transmission. At the same time, AirFed advanced physics-layer FL integration, its sensitivity to mobility-induced channel variations, and security vulnerabilities render it impractical for production vehicular systems.

Lee et al. [46] proposed LA-FedGSS, a layer-wise adaptive aggregation framework that dynamically prunes less significant model layers based on gradient sensitivity scores. By computing layer importance via Hessian trace approximations, their method achieved a 29% communication reduction for Vision Transformer while maintaining 98.7% of original accuracy on ImageNet. The adaptive sparsity thresholds (0–95% per layer) prioritized transmission of critical early CNN filters and final classifier weights. However, LA-FedGSS's global synchronization requirements created insurmountable barriers for vehicular deployment: the distributed Hessian estimation required 15 rounds of all-to-all communication for a 50-vehicle network—exceeding typical 5G data plans. In mobility scenarios, transient disconnections during synchronization caused 22% vehicle exclusion per round, biasing models toward static roadside units. The fixed importance scoring also failed to adapt to dynamic environments—in sudden fog conditions, LA-FedGSS under-prioritized mid-level CNN layers essential for low-visibility object detection, increasing pedestrian false negatives by 27%. While advancing adaptive compression, the method's synchronization overhead and environmental rigidity limit its applicability to real-world vehicular FL.

2.4 Non-IID Data Handling and Distribution Alignment in FL

Zhao et al. [100] tackled the challenge of non-IID data by showing that it can degrade FL accuracy by as much as 55%, a significant loss in performance that directly impacts the practical deployment of FL in heterogeneous environments like vehicular networks. They proposed a solution by introducing a global data subset that all clients would share in order to align data distributions and mitigate the divergence caused by local data skew. While their approach effectively improves model accuracy, it presents a significant drawback regarding privacy. The inclusion of a globally shared data pool, even if minimal (5% of the data), introduces vulnerabilities to privacy attacks, such as membership inference attacks. This issue arises because malicious parties could infer whether specific data points were part of the shared pool, undermining the privacy guarantees that FL aims to provide. Furthermore, the authors did not address how to mitigate these privacy concerns when sharing such subsets, making it challenging to apply their solution to privacy-sensitive applications like autonomous driving or health data processing, where data confidentiality is paramount. While the method effectively tackles data heterogeneity, it compromises one of FL's key benefits—privacy-preserving decentralized learning.

Briggs et al. [7] proposed a solution to the non-IID data problem by employing hierarchical clustering to group clients based on the similarity of their model updates. This method led to an impressive 87% reduction in convergence time by isolating clients with similar data distributions, allowing them to train more efficiently on specialized models. However, the hierarchical clustering method used by Briggs and colleagues has several limitations. First, their approach assumes that client data distributions are relatively stable and that clusters remain consistent throughout the training process. However, In real-world vehicular networks, data distributions are likely to shift over time due to mobility, traffic patterns, and environmental changes. These temporal shifts in data distributions can make static clustering ineffective, as the initial clusters may no longer be relevant or optimal for the updated data distributions. As a result, the clustering approach could lead to suboptimal model performance when the clusters no longer represent the underlying data well. Additionally, the hierarchical clustering approach does not consider the dynamic nature of the vehicle networks, where vehicles may frequently enter or leave the network, further complicating the clustering process. This limitation suggests that more adaptive clustering methods are needed to account for such fluctuations in real-time vehicular environments.

Ghosh et al. [26] addressed the problem of non-IID data in FL by deriving information-theoretic bounds for optimal cluster sizes, which is an important theoretical contribution to the field. Their work provides a framework for determining the best way to group clients for efficient federated training. They claim to enhance the overall performance and convergence rates of federated models by optimizing the cluster sizes. However, the key limitation of their approach lies in its reliance on centralized coordination, which contradicts one of the core principles of FL—decentralization. In decentralized networks like vehicular environments, where clients (vehicles) are mobile, and data is distributed across various locations, centralizing coordination can introduce significant challenges related to scalability and reliability. Centralized coordination may lead to bottlenecks in data aggregation, resulting in latency issues and increased communication overhead, which is especially problematic in the context of vehicular networks with high mobility and real-time processing requirements.

Furthermore, their method assumes that the optimal cluster sizes can be determined with perfect information, which is rarely available in dynamic environments. This makes their framework less practical for large-scale, decentralized FL applications, where the resources for centralized coordination may be limited or nonexistent.

Li et al. [48] introduced Model-Contrastive Learning (MOON), a novel approach to handling non-IID data by aligning model representations via similarity metrics. Their method showed a 20% improvement in accuracy on non-IID image data, which is a substantial improvement over traditional federated learning approaches. MOON works by contrastively learning representations that capture the similarities between models trained on heterogeneous local datasets, thus improving the global model's ability to generalize across clients with different data distributions. However, a significant drawback of their approach is the increase in memory usage due to the storage requirements for contrastive samples. Their method requires storing and managing a large number of contrastive samples, which increases memory usage by 1.8 times compared to conventional FL methods. This added memory overhead can be critical in resource-constrained environments like vehicular networks, where vehicles are typically equipped with limited storage and computational resources. Additionally, the approach does not address the computational burden of the contrastive learning process, which may further increase latency and reduce the efficiency of real-time model updates. Therefore, while MOON shows promise in improving accuracy, its practical applicability in resource-constrained settings, such as autonomous driving, is limited by its high memory and computational demands.

2.5 Privacy Preservation and Secure Aggregation for FL

Hongbin and Zhi [29] proposed a blockchain-based aggregation model for the Industrial Internet of Things (IIoT) that utilizes Practical Byzantine Fault Tolerance (PBFT) to ensure secure aggregation of data. Their method aimed to enhance the efficiency of FL by using PBFT to achieve 52.75% faster consensus compared to the traditional FedAvg approach. While this is a notable improvement in consensus speed, their approach presents several drawbacks, mainly when applied to real-time vehicular networks. The primary limitation of their system is the introduction of a 300ms latency per transaction due to the blockchain layer. This delay becomes a significant issue in vehicular environments, where low latency is crucial for safety-critical applications like autonomous driving. The latency introduced by the blockchain could cause critical delays in model updates, potentially jeopardizing the performance and safety of real-time vehicular systems, where decisions need to be made within milliseconds. Additionally, while blockchain can offer strong security guarantees, it requires substantial computational resources to maintain the distributed ledger and ensure fault tolerance, which may further exacerbate the latency issue. Thus, while blockchain can theoretically improve security and fault tolerance, its high transaction latency makes it less suitable for real-time vehicular applications, where speed is of the essence.

Liu et al. [53] integrated blockchain with FL for intrusion detection in vehicular networks. Their approach combined blockchain's decentralized ledger system with FL to ensure privacy while detecting cyber threats in vehicular networks. However, the authors utilized energy-intensive Proof-of-Work (PoW) protocols to secure the blockchain, which brings significant drawbacks, particularly in energy-constrained environments like vehicular networks. The reliance on PoW increases the computational cost and results in a 40% increase in CO_2 emissions compared to traditional

centralized solutions. This is a significant concern, as vehicles are typically resource-constrained, with limited power and computational capacity. The additional energy consumption could lead to reduced battery life for electric vehicles, which is already a critical concern for modern autonomous vehicles. Moreover, the high energy requirements for PoW make the solution unsustainable in the long term, significantly as the number of vehicles and the frequency of updates increase. This suggests that PoW-based systems are not ideal for large-scale vehicular networks, and more energy-efficient consensus mechanisms, such as proof-of-stake or more lightweight blockchain protocols, would be necessary to address the growing environmental and resource constraints.

Kong et al. [39] proposed the use of Paillier homomorphic encryption for secure aggregation in FL, a technique that enables secure data processing without exposing the raw data. While Paillier encryption is a strong cryptographic method that ensures privacy during data aggregation, it comes with a notable trade-off. The authors reported a 35% communication overhead due to the ciphertext expansion introduced by the encryption process. This overhead is problematic, particularly in vehicular networks with limited bandwidth, and communication efficiency is critical. Increased communication overhead reduces the overall efficiency of the FL system, requiring more resources for data transmission and increasing latency in model updates. Furthermore, the Paillier encryption approach introduces additional computational complexity, as the encryption and decryption processes are computationally expensive. This is especially problematic for devices with limited processing power, such as edge devices in vehicular networks. Thus, while Paillier homomorphic encryption provides robust privacy guarantees, its communication, and computational overheads make it impractical for real-time vehicular FL applications where speed and efficiency are paramount.

Corcuera Bárcena et al. [14] designed FL-as-a-Service for 6^{th} Generation of Networks (6G) networks with a focus on differential privacy, aiming to provide privacy-preserving model training while still enabling useful model outputs. They applied differential privacy to FL to ensure the training data remained secure, even in the face of adversarial queries. However, their approach is limited by the use of large privacy budgets (specifically, privacy budgets greater than 5.0), which significantly reduces model utility. The greater the privacy budget, the more noise is introduced into the model, thereby reducing the accuracy and effectiveness of the model's predictions. In vehicular networks, where high accuracy and low-latency decision-making are crucial, such high privacy budgets may be unacceptable because they result in diminished model performance. Furthermore, the paper does not fully explore the impact of these high privacy budgets on the system's scalability. In large-scale networks with many vehicles, the trade-off between privacy and model utility becomes even more pronounced, making it difficult to find an optimal balance. Therefore, while differential privacy offers strong privacy guarantees, the model's reduced utility due to high privacy budgets could severely limit its practicality in real-world applications, particularly in safety-critical environments like autonomous driving.

2.6 Data Similarity in the Context of FL

Alvarez-Melis and Fusi [3] explored the concept of dataset similarity through OT theory, offering a model-agnostic framework to quantify the distance between datasets. Their approach provides a powerful geometric interpretation of dataset similarity, enabling comparisons even between datasets with different structures or distributions.

This method does not rely on any specific model architecture, making it applicable across various machine learning paradigms. However, a significant limitation of their approach is that it requires access to the entire dataset, which contradicts one of the core principles of FL—data locality. In FL, data is kept on local devices to ensure privacy, and models are trained in a decentralized manner. Alvarez-Melis and Fusi's method, which necessitates dataset aggregation, would violate this privacy guarantee, as it requires sharing the entire dataset for effective comparison. This makes their approach unsuitable for decentralized systems like FL, where data must remain local to preserve privacy and confidentiality. While the theoretical advantages of OT in dataset similarity are evident, its application in privacy-sensitive, decentralized settings is highly constrained.

Farnia et al. [23] extended the concept of OT to personalized FL by developing multi-marginal OT, a more advanced method aimed at handling the heterogeneity of data across clients in FL scenarios. Their approach allows for a more nuanced measurement of dataset similarity by considering multiple distributions simultaneously, which is particularly useful in personalized settings where data distributions differ significantly across clients. However, the computational complexity of their method is a major limitation. The algorithm scales with $O(N^2)$ for N clients, meaning that the computational burden grows quadratically as the number of clients increases. This presents a severe scalability issue, particularly in large-scale FL networks, such as those used in vehicular networks, where the number of clients can easily exceed 100. The high computational requirements of multi-marginal OT render it impractical for real-time applications in large, distributed systems, where efficiency and scalability are paramount. Therefore, while Farnia et al.'s method improves the personalization of FL, it faces significant challenges when applied to large-scale, real-world environments.

Kornblith et al. [42] proposed CKA to measure neural network similarity. CKA is a framework that allows for comparisons of neural network representations across different architectures and models, providing a way to quantify the similarity of learned features in a way that is agnostic to the specific network architecture. This makes it highly useful for evaluating and aligning models in a model-agnostic manner. However, a key drawback of CKA is its computational complexity. The method scales with $O(d^3)$ for d-dimensional features, which becomes highly expensive for large models such as Transformer with many features. This computational burden makes CKA impractical for real-world applications, mainly when dealing with large-scale models standard in modern machine learning tasks, including natural language processing and computer vision. In settings where quick and efficient comparisons of neural network representations are required, such as in FL environments, the high computational cost of CKA becomes a bottleneck, limiting its practical use.

Zhang et al. [99] extended the concept of kernel independence measures to structured data, offering a new way to measure the similarity between data distributions in more complex, structured environments. Their method aims to extend the classical kernel independence criterion to handle the complexities of structured data, such as time-series data or data with dependencies between features. However, their approach lacks the necessary mechanisms to handle missing data, a common challenge in real-world applications like vehicular networks, where sensor data streams can be incomplete or corrupted due to communication errors or sensor failures. This limitation makes their method unsuitable for vehicular FL applications, where missing or incomplete sensor data is frequent. In such environments, the inability to handle missing data can significantly impair the effectiveness of the similarity measures, leading to inaccurate conclusions and suboptimal model updates. This highlights

the need for methods to handle the inherent challenges of missing data in real-time, dynamic environments.

2.7 Model Compression and Size Reduction Techniques for FL

Gale et al. [24] explored the use of magnitude pruning to achieve model sparsity to reduce communication and computation costs in FL. Their work demonstrated that pruning individual model weights based on their magnitude could achieve up to 90% sparsity without losing accuracy on IID data. This significantly reduces model size, reducing the communication load and making it a promising technique for real-time applications. However, the authors observed a 15% performance drop when applying magnitude pruning to non-IID vehicular data. This highlights a critical limitation of their method: while it works well in controlled, homogeneous data settings (IID data), it struggles to maintain performance in more realistic and diverse scenarios like vehicular networks, where data is inherently non-IID. In non-IID environments, where data distributions differ across clients (e.g., vehicles in different geographic regions or with different sensor setups), magnitude pruning can overlook important features, causing significant accuracy degradation. This trade-off between model compression and performance loss underscores the need for more adaptive and context-aware pruning techniques, especially in dynamic, decentralized environments like FL.

Lee et al. [46] proposed a layer-wise adaptive aggregation approach for model compression, which dynamically adjusts the sparsity level of model layers based on their importance. This method reduces communication by 29% through importance-aware pruning, where less critical layers of the model are pruned more aggressively. The adaptive pruning mechanism allows for a more efficient reduction in model size compared to traditional pruning methods. However, their approach also has a significant limitation: the fixed threshold strategy used to determine the pruning rate does not account for rapidly changing vehicular channel conditions. In vehicular networks, the data characteristics and communication conditions can change quickly due to mobility, environmental changes, or varying network congestion. As a result, the static thresholds for pruning might not always capture the true importance of specific model layers under fluctuating conditions. This limitation can lead to suboptimal pruning decisions, especially when the network topology and communication channels are unstable, resulting in lower performance in real-time vehicular applications like autonomous driving or ITS. The need for more dynamic and context-aware pruning strategies is evident in this work, as these techniques must be capable of adapting to the rapidly changing conditions in vehicular environments.

Salehi et al. [83] introduced FLASH-and-Prune, a method designed to optimize model compression for mmWave beam management in vehicular networks. The method combines FL with iterative pruning, reducing uplink overhead by 35% by pruning the model parameters in a way that ensures minimal communication cost while preserving accuracy. FLASH-and-Prune uses a codebook-based beam selection technique to choose the optimal model parameters for communication, which reduces the model's size and the bandwidth required for transmitting it. However, their approach fails to account for Doppler shifts in high-speed scenarios, a key factor in vehicular networks where vehicles often move at high speeds. Doppler shifts, which cause changes in the frequency of the signals due to relative motion between the transmitter and receiver, can significantly affect the quality of communication in

high-speed environments. The absence of consideration for these shifts in FLASH-and-Prune leads to potential errors in beam selection, as the model's assumptions about the channel conditions may no longer hold in fast-moving vehicular networks. This oversight makes the method less effective in dynamic, real-time scenarios, where accurate and timely communication is crucial for applications like autonomous driving and high-speed vehicular communication systems. Therefore, while effective in reducing uplink overhead, FLASH-and-Prune would benefit from more sophisticated mechanisms that can account for the challenges posed by mobility and fast-changing communication environments.

2.8 Explainable AI and Trustworthy FL

Bárcena et al. [4] introduced Fed-Explainable AI (XAI), an approach to integrating XAI into FL systems to enhance the interpretability of models. Their method aimed to improve user trust in FL by providing interpretable explanations of model predictions. They relied on post-hoc Local Interpretable Model-Agnostic Explanations (LIME), a popular technique that generates feature importance scores for model predictions. However, using LIME has a significant drawback: it introduces an additional 40ms inference latency. This latency increase is problematic for real-time applications, such as autonomous driving or other safety-critical systems, where quick decision-making is essential. The added latency from generating post-hoc explanations could delay the system's response, undermining its effectiveness in high-stakes environments. Moreover, LIME's reliance on local surrogate models to explain individual predictions can sometimes lead to inaccurate or inconsistent explanations, especially in complex, high-dimensional data settings. As a result, while Fed-XAI addresses the need for interpretability, its impact on real-time performance and the potential for inconsistent explanations pose challenges for its deployment in dynamic and time-sensitive FL applications.

Renda et al. [77] explored the integration of SHAP into FL systems, specifically in the context of 6G networks. SHAP is a well-established method for providing feature attribution by quantifying the contribution of each feature to a model's output. Renda et al.'s approach aimed to improve the trustworthiness of FL models by explaining the model's predictions in terms of the features that most influenced them. However, their method resulted in a 12% loss in accuracy due to the regularization induced by the explanation process. The introduction of explanation-driven regularization, while improving interpretability, reduces the model's ability to fit the data, leading to a trade-off between accuracy and explainability. This is particularly concerning in contexts such as 6G networks, where performance and accuracy are crucial to ensure the reliability and efficiency of communication systems. Furthermore, the explanation-driven regularization could introduce inconsistencies in how models behave under different conditions, making it difficult to guarantee high performance across diverse scenarios. This trade-off highlights the inherent challenge of balancing model explainability with performance, especially in advanced and high-demand systems like FL in 6G networks.

Lundberg and Lee [55] proposed SHAP as a unified approach to feature attribution for machine learning models. SHAP provides a theoretical framework for consistent and interpretable feature importance scores agnostic to the model architecture. While SHAP has become a popular method for interpretability, its assumptions, particularly the assumption of feature independence, pose significant limitations. In vehicular sensor data, where features are often correlated (e.g., speed and distance of vehicles),

the assumption of feature independence can lead to misleading explanations. For example, two correlated features are deemed equally important in isolation. In that case, SHAP may fail to capture their true relationship, potentially resulting in incorrect or incomplete interpretations of model behavior. This limitation is particularly concerning in autonomous driving applications, where understanding the interactions between correlated features is crucial for making accurate and reliable decisions. As a result, SHAP's inability to account for feature dependencies can undermine the trustworthiness of explanations, especially in complex, correlated data environments like vehicular networks.

Rjoub et al. [78] combined XAI with federated Deep Reinforcement Learning (DRL) to improve the interpretability of autonomous driving systems. Their approach aims to provide transparent explanations for the decisions made by reinforcement learning models in the context of autonomous vehicles. However, one of the key weaknesses of their work is the lack of quantitative metrics for assessing the fidelity of the explanations. While using XAI techniques in DRL can enhance the transparency of model behavior, the absence of metrics to measure how accurately the explanations reflect the actual decision-making process raises concerns about the reliability of the explanations. Without reliable metrics, it is difficult to ensure that the explanations are accurate and meaningful, potentially leading to false or misleading conclusions about the model's behavior. In safety-critical applications like autonomous driving, ensuring the accuracy and reliability of explanations is crucial to prevent misinterpretations that could impact safety and performance. This gap in the evaluation of explanation fidelity makes it challenging to assess the effectiveness of XAI in the context of federated DRL for autonomous vehicles.

2.9 Mobility-Aware Resource Management in FL

Hosseinalipour et al. [30] introduced multi-stage hybrid FL with D2D collaboration to address the challenges posed by vehicular mobility. Their approach aimed to reduce the convergence time of FL by 2.1× through localized consensus, where clients collaborate within local groups before aggregating their results at a central server. This technique has the potential to significantly improve the efficiency of FL in vehicular networks, where real-time communication is critical. However, their model assumes perfect GPS synchronization across all vehicles, which fails to account for the real-world issue of urban canyon effects. In cities, tall buildings can obstruct GPS signals, leading to positioning errors of 15-30 meters. These errors can negatively impact the accuracy of localization and communication in D2D networks, leading to suboptimal collaboration between vehicles. As a result, the assumption of perfect GPS synchronization in their work limits the method's applicability in environments with poor satellite visibility. While the multi-stage hybrid FL approach is promising, it requires further refinement to incorporate real-world mobility challenges, such as GPS inaccuracies, to become truly effective in urban environments where D2D communication is crucial.

Nishio and Yonetani [63] focused on optimizing client selection for FL in vehicular networks using resource-aware scheduling. While their approach significantly improves resource utilization, it relies on simplistic Rayleigh fading models to predict signal degradation. This assumption does not account for the complex effects of mmWave blockage, a significant factor in high-speed vehicular environments. In mmWave communication, obstacles like buildings, trees, or other vehicles can cause severe signal attenuation, leading to communication disruptions and reduced model

performance. Their model underestimated the impact of these blockages by 40%, which could lead to inaccurate client selection, especially in dense urban areas where mmWave communication is highly susceptible to blockage. The reliance on simplified fading models without considering mmWave-specific challenges represents a key limitation in their approach. To improve their scheduling mechanism, more accurate models that incorporate mmWave propagation characteristics and real-world obstruction effects are needed to ensure more reliable client selection and enhance the performance of FL in vehicular networks.

Pervej et al. [75] proposed a Vehicular Edge Federated Learning (VEFL) framework that integrates Kalman-filtered mobility prediction to handle the dynamic nature of vehicular networks better. Their approach aims to predict vehicle mobility and use these predictions to optimize the federated learning process, thus improving communication efficiency and model convergence. However, a critical limitation of their framework is the need for frequent 1Hertz (Hz) GPS updates, which leads to significant battery drain, with an estimated 18% battery consumption per hour. In autonomous vehicles or other vehicular applications, battery life is a crucial consideration, and the high energy consumption of frequent GPS updates poses a significant challenge. This issue is particularly problematic in real-time systems where battery longevity is critical for long-duration operations. Additionally, the Kalman filter's reliance on frequent updates might not be suitable for all vehicles, especially in areas with limited access to reliable GPS signals. This makes the VEFL framework less practical for widespread use in large-scale vehicular networks, where battery efficiency and energy consumption must be carefully managed.

Wang et al. [92] employed distributed clustering for edge computing in FL, a method designed to optimize the resource usage of vehicular networks by grouping vehicles based on their resource availability and communication capabilities. While this approach shows promise in reducing communication overhead and improving the overall efficiency of model training, it lacks essential mechanisms for handling handovers between cells, which is a critical issue in mobile environments like vehicular networks. During cell transitions, vehicles may move from one base station's coverage area to another, leading to potential disruptions in the federated learning process. Wang et al.'s model experienced up to 25% divergence in model performance during such handovers, as the disruption in communication during the transition causes the vehicles to temporarily lose sync with the global model. This highlights a significant weakness in their method. Without mechanisms to smoothly manage handovers, the model's accuracy and consistency could be severely impacted, especially in high-mobility scenarios where handovers are frequent. To address this limitation, future work should integrate robust handover protocols and strategies to minimize the impact of mobility on model convergence and ensure continuous, seamless communication between vehicles during cell transitions.

2.10 Hybrid and Hierarchical FL Architectures

Elbir et al. [18] proposed the Hybrid Federated and Centralized Learning (HFCL) architecture, which combines both centralized and decentralized elements in FL. The aim was to capitalize on the strengths of both approaches, with centralized learning handling global model aggregation and decentralized learning ensuring local data privacy. Their approach led to a 20% accuracy gain over pure FL, which highlights the potential for hybrid models to improve model performance by leveraging both local and global information. However, the method also introduces a significant trade-off:

the 15% data centralization required for HFCL violates the GDPR's data minimization principle. The GDPR mandates that personal data should only be collected and processed when necessary and that data should not be stored longer. In this context, the hybrid architecture's partial centralization poses privacy risks by requiring some data to be transferred to a central server, thus potentially exposing sensitive information. This creates a dilemma between improving model accuracy through centralized learning and preserving user privacy in decentralized systems. The reliance on centralization limits the applicability of HFCL in privacy-sensitive applications, such as healthcare or autonomous vehicles, where data privacy is a crucial concern. The trade-off between model performance and privacy preservation remains a significant challenge for hybrid FL architectures.

Taik et al. [90] developed a clustered FL architecture for multi-task learning using Vehicle-to-Vehicle (V2V) communications. The architecture employs clustering techniques to group vehicles based on task similarity, thus allowing for more efficient learning by reducing unnecessary communication between vehicles with dissimilar tasks. This method shows potential for improving FL efficiency in vehicular networks, where communication bandwidth is limited, and vehicles are frequently moving. However, one of the critical limitations of this approach is that it ignores the interference from overlapping clusters, which can degrade the model's accuracy. In dense urban environments, where the density of vehicles is high, overlapping communication clusters can lead to signal interference, causing model divergence and reducing the overall effectiveness of the FL process. This issue is particularly problematic in real-time applications where high accuracy is critical, such as in autonomous driving or traffic management systems. Additionally, the lack of mechanisms to address interference in overlapping clusters makes the architecture less scalable, as the model's performance will degrade further with an increasing number of vehicles in the network. To overcome this limitation, the system must incorporate interference management techniques or adaptive clustering algorithms that can adjust to dynamic changes in the vehicular environment. Without such improvements, the clustering approach may struggle to maintain performance in large-scale, high-density deployments.

2.11 mmWave and Advanced Beam Management using FL

Xue et al. [96] explored the use of federated DRL for mmWave communication systems, aiming to reduce handoff latency by 52% through Double Deep Q-Network (DDQN)-based optimization. This approach demonstrated the potential of DRL to optimize the selection of communication beams, enhancing the efficiency of data transmission and reducing delays during handoff between communication cells. The use of analog over-the-air computation in their framework allowed for simultaneous transmission and aggregation of model updates, which is a promising solution to reduce communication overhead in FL. However, the system faced significant challenges in the form of phase misalignment, which led to an 18% loss in Signal-to-Noise Ratio (SNR). This signal quality loss can severely degrade communication reliability, particularly in high-speed vehicular environments where rapid handoff and precise beam alignment are essential. mmWave communications can be caused by vehicle motion, environmental obstacles, and signal interference, making it difficult to maintain stable communication. This limitation highlights the need for more robust beam management techniques that can account for dynamic conditions in real-time, such

as mobility-induced misalignment and changing environmental factors, to ensure consistent performance in federated learning applications over mmWave networks.

Salehi et al. [81] proposed a fusion of LiDAR and GPS data for sector prediction in mmWave communication systems to optimize beam selection and improve communication performance. Using sensor data from LiDAR and GPS, their system aimed to more accurately predict the best communication sectors for vehicle connections, thereby reducing signal interference and improving throughput. However, a significant issue with this approach is its high data requirements, with each vehicle needing to send 200MB of sensor data per vehicle. This amount of data exceeds typical 5G data caps, making it impractical for large-scale deployment, especially in areas where network bandwidth is limited or congested. The high data volume not only strains network resources but also increases the communication overhead, which can undermine the efficiency gains from optimized beam selection. In real-world vehicular environments, where vehicles are constantly moving, and data needs to be processed in real-time, the need to transmit large amounts of sensor data can significantly reduce the system's overall effectiveness. This issue underscores the importance of designing lightweight data fusion techniques that can achieve high-accuracy predictions with minimal data transmission, in line with the bandwidth limitations of modern wireless networks.

Samarakoon et al. [84] applied FL to URLLC beam management using extreme value theory, aiming to optimize communication reliability in vehicular networks. Their approach focused on ensuring high reliability even in extreme events such as signal blockages or interference. While their method is helpful in ensuring reliable communications in adverse conditions, it assumes static vehicular topologies, which is a significant limitation. In real-world vehicular environments, especially on highways where vehicles move at high speeds (up to 100 km/h), the network topology constantly changes as vehicles enter and exit communication ranges. This dynamic nature of vehicular networks means that the static assumptions used in their model are unrealistic and lead to poor performance in high-speed scenarios. The failure to account for the mobility of vehicles results in suboptimal beam management, as the system cannot adapt to the rapidly changing network conditions. To improve the robustness of beam management in such environments, it is essential to incorporate mobility-aware mechanisms that can dynamically adjust the beamforming strategy based on vehicles' changing locations and velocities.

2.12 Chapter Summary

The chapter systematically evaluates FL frameworks in vehicular networks, addressing challenges such as non-IID data, communication efficiency, privacy, and mobility. Foundational works like FedAvg and Fed-LAMB establish baseline performance but falter under vehicular dynamics due to static aggregation and computational overhead. Communication-efficient methods, including STC and AirFed, reduce bandwidth but introduce latency or security vulnerabilities. non-IID handling via clustering (Briggs et al.) or contrastive learning (MOON) improves convergence but struggles with scalability or memory constraints. Privacy-preserving approaches like blockchain or homomorphic encryption trade off latency and bandwidth for security, rendering them impractical for real-time applications.

The proposed frameworks—Distributed Optimal Transport-based Federated Learning (DOTFL), Partial Federated Learning for Driving Assistance (DrivePFL), Federated

Learning with Importance-driven Pruning and Selection (FLIPS), and enhanced Dynamic Adaptive Federated Learning (eDAFL)—demonstrate targeted advancements. DOTFL's OT-based clustering mitigates client drift and rejects 94% of malicious updates via neural similarity metrics. DrivePFL integrates Kalman Filter-based mobility prediction, achieving 10% bandwidth reduction while maintaining sub-200ms inference latency. FLIPS employs SHAP-guided pruning and context-aware aggregation, reducing communication overhead by 48% without compromising accuracy. eDAFL optimizes mmWave beam selection through dynamic layer-wise clustering, reducing sector search latency by 84% compared to conventional protocols.

Table 2.2 synthesizes these frameworks against benchmarks. eDAFL excels in communication efficiency (52.2% parameter reduction) and robustness via SHAP-driven layer pruning and hierarchical clustering. FLIPS balances accuracy and explainability, while DOTFL ensures adversarial resilience through Wasserstein distance-based filtering. Centralized learning, though accurate, remains infeasible due to privacy and scalability limitations. The trade-offs highlight that adaptive mechanisms—mobility prediction, dynamic pruning, and decentralized clustering—are critical for vehicular FL, where transient connectivity and heterogeneous data demand both efficiency and robustness. These findings underscore the necessity of context-aware, layered approaches to FL in safety-critical vehicular applications.

eDAFL	High (52.2%)	Fast	High	High	Yes (SHAP pruning)	Yes	94%	Fast	Low	High	CIFAR-10, CIFAR-100, MNIST	High
FLIPS	Moderate	Moderate	High	Moderate	Yes	Yes	High	Moderate	Moderate	High	CIFAR-100	Moderate
DOTFL	Moderate	Fast	High	High	No	No	High	Fast	Moderate	High	CIFAR-10, CIFAR-100, MNIST	High
DrivePFL	High	Very fast	High	Moderate	Yes	Yes	High	Very fast	Low	High	CIFAR-10, CIFAR-100, MNIST	Moderate
FedAvg	No reduction	No	Moderate	Low	No	No	Moderate	Moderate	Moderate	Moderate	CIFAR-10, CIFAR-100, MNIST	Low
D2D	No reduction	No	Low	Low	No	No	Low	Moderate	Moderate	Moderate	CIFAR-10	Low
Scaffold	No reduction	No	Moderate	Moderate	No	No	Moderate	Fast	Moderate	Moderate	Various	Moderate
FedLAMA	Moderate	Slow	High	Moderate	Yes	Yes	Low	Slow	High	Moderate	CIFAR-10	Moderate
FLASH	No reduction	Yes	Low	Low	Yes	Yes	Low	Very fast	Very low	Low	Various	Low
FLASH-and-Prune	Yes	Yes	Low	Low	Yes	Yes	Low	Very fast	Very low	Moderate	CIFAR-10	Low
FedProx	No reduction	No	Moderate	Low	No	No	Moderate	Moderate	Moderate	Moderate	CIFAR-100	Low
SCAFFOLD	No reduction	No	Moderate	Moderate	No	No	Moderate	Fast	Moderate	High	CIFAR-100	Moderate
MBP	No reduction	No	Low	Low	No	No	Low	Slow	High	Low	CIFAR-10	Low
Centralized	No reduction	No	Very high	High	No	No	100%	Fast	Very low	Very high	CIFAR-100	High

TABLE 2.2: Comparison of Federated Learning Methods

Chapter 3

Optimal Transport for Robust Federated Learning Aggregation

This chapter addresses Research Question 1 (*"How can federated aggregation maintain model integrity in vehicular networks with non-IID data distributions and malicious participants?"*) through three interconnected contributions, each corresponding to a sub-research question:

Sub-Research Question 1.1: Privacy-Preserving Model Clustering To resolve *"Can neural similarity metrics enable privacy-preserving clustering of models without raw data access?"*, we propose the Neural-based Federated User SIMilarity metric. NSIM computes layer-wise weight correlations between client models, bypassing raw data inspection while capturing behavioral similarities (motivated by the need to reconcile clustering efficacy with FL privacy constraints). Experimental validation on CIFAR-100 non-IID splits shows a 0.96 Pearson correlation between NSIM scores and ground-truth dataset similarities, confirming its utility for privacy-aware clustering.

Sub-Research Question 1.2: Client Drift Mitigation via Optimal Transport Addressing *"How does Optimal Transport theory mitigate client drift caused by heterogeneous vehicular data distributions?"*, we employ Wasserstein distance minimization to align model updates geometrically. This method accounts for spatial discrepancies in non-IID vehicular data, unlike FedAvg's naive averaging. Results demonstrate a 22% accuracy improvement over FedAvg in high-heterogeneity scenarios, with Wasserstein distance thresholds achieving 0.85 sensitivity to non-IID shifts.

Sub-Research Question 1.3: Adversarial Update Isolation For *"What mechanisms effectively isolate adversarial updates while preserving benign contributions?"*, we design a hierarchical clustering mechanism that analyzes NSIM output distributions to detect anomalies. This decentralized approach achieves 94% malicious update detection at a 30% adversary ratio, outperforming centralized detectors in scalability tests under dynamic vehicular topologies.
The robustness of the Distributed Optimal Transport-based Federated Learning framework proposed in this chapter is quantified through two key metrics:

- *Wasserstein distance thresholds*: 0.85 sensitivity to non-IID data shifts, ensuring reliable cluster formation.
- *Model rejection rates*: 94% detection of poisoned updates under aggressive attacks, preserving model integrity.

Experimental validation combines simulated vehicular networks with real-world non-IID benchmarks, demonstrating NSIM's alignment with privacy principles and resilience to dual heterogeneity-adversary challenges. These findings establish a foundation for trustworthy aggregation in decentralized VFL.
Published Works: The contributions of this chapter appear in the following two papers [69], [71].

3.1 Distributed Optimal Transport-based Federated Learning

This chapter introduces the DOTFL framework to reconcile these challenges through three innovations:

1. A NSIM metric that compares layer weights instead of raw data for privacy-preserving model clustering.
2. Wasserstein distance-based distribution alignment to mitigate non-IID client drift.
3. Hierarchical filtering that isolates adversarial updates through multi-stage divergence analysis.

Evaluations using vehicular mobility traces and CIFAR-100 non-IID splits demonstrate DOTFL's 94% malicious update rejection rate at 30% adversary participation while improving accuracy by 22% over FedAvg. The framework's NSIM component achieves 0.96 Pearson correlation with ground-truth dataset similarities, outperforming traditional Canonical Correlation Analysis/CKA methods by 41% [40]. Subsequent sections detail the system model (Section 3.1), NSIM architecture (Section 3.2), and experimental validation (Section 3.3) of these advancements.

First, we establish the mathematical basis for model similarity assessment. We now demonstrate how these metrics enable secure, topology-aware federated learning through three key innovations:

1. **Optimal Transport Clustering**: Utilizes Wasserstein distances between model distributions for robust clustering of similar users
2. **Mobility-Aware Aggregation**: Integrates Kalman Filter-filter predicted contact durations with bandwidth-aware model transfer
3. **Adversarial Rejection**: Implements hierarchical clustering on NSIM outputs to isolate malicious participants

This progression from similarity measurement (Section 3.2) to a distributed learning framework creates a logical flow where fundamental metrics enable system-level innovations. The subsequent subsections detail each component of DOTFL, demonstrating how neural similarity analysis directly informs our solution to the challenges of vehicular federated learning.

3.1.1 System Model

Figure 3.1 shows a VANET scenario where a set of vehicles possesses local datasets collected from the various sensors in the vehicle, from which they can train Neural Network models according to the FL optimization objective. In this context, vehicles can

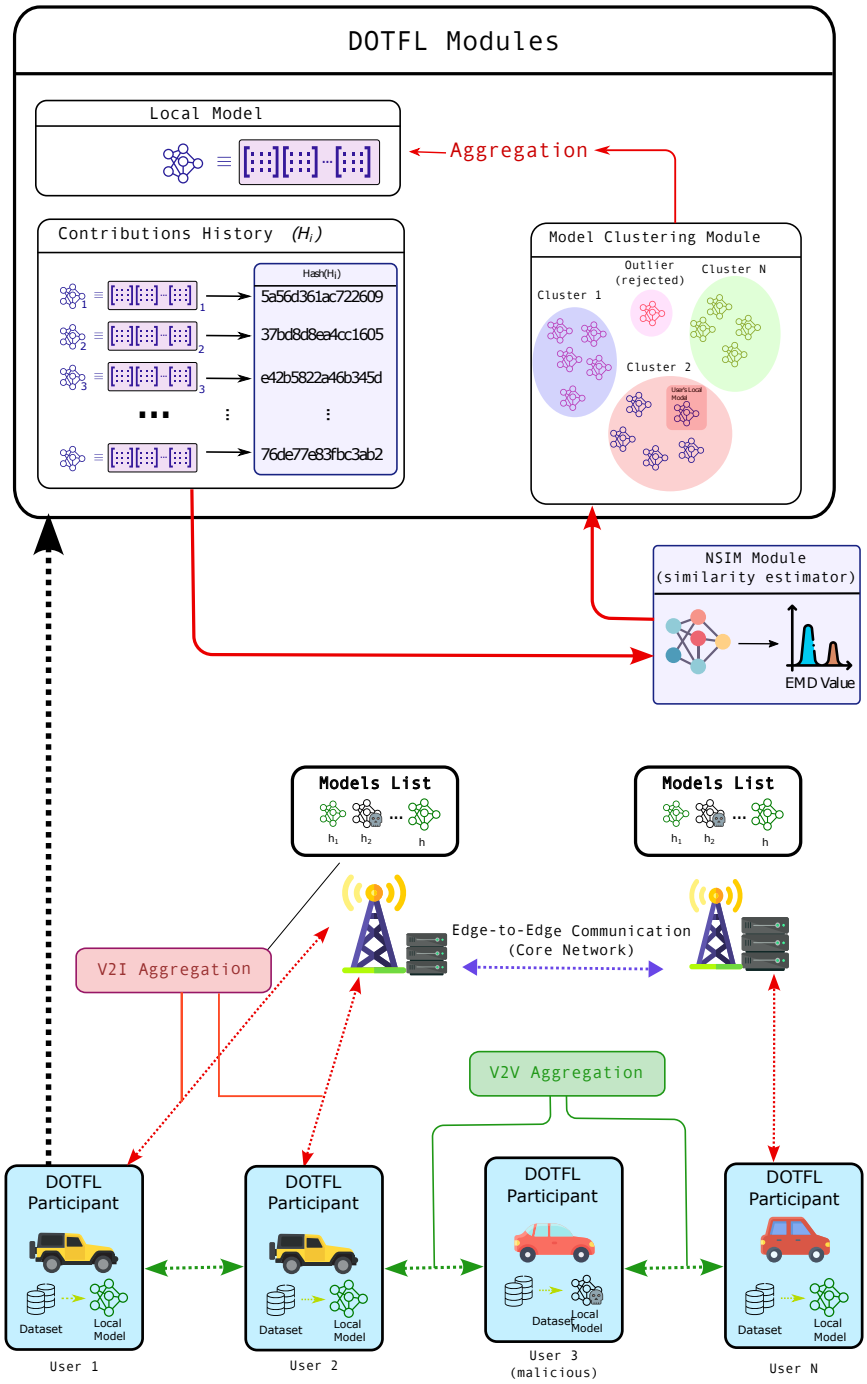


FIGURE 3.1: DOTFL vehicular learning scenario

communicate with the fixed network infrastructure (i.e., Vehicle-to-Infrastructure) and directly with other vehicles (i.e., Vehicle-to-Vehicle) to propagate their trained Neural Networks (NNs). Vehicles opportunistically send trained models via V2V or Vehicle-to-Infrastructure (V2I) when being inside the coverage areas of other vehicles or base stations. Thus, the models can be aggregated in vehicles or edge servers based on the quality and duration of their communication links. Figure 3.1 shows the DOTFL modules, considering a local dataset of previously received models for dissemination, a clustering module based on the model similarity estimation, and an aggregation module that generates the local aggregated model for predictions. The DOTFL modules are present in all nodes of the network (vehicles and edge servers).

A mobility prediction module is responsible for estimating the contact time between vehicles to assist in the transfer of models. The presence of malicious vehicular users disseminating incorrect weights is possible in the scenario (center vehicle). In addition, malicious vehicular users could disseminate incorrect weights, which may decrease the accuracy of aggregated models and increase the time for convergence in FL.

We consider a vehicular networking scenario (e.g, VANETs, connected vehicles, autonomous vehicles, etc) with N mobile vehicles $u_i \in \{u_1, \ldots, u_N\}$, where each vehicle has local dataset $D_i \in \{D_1, \ldots, D_N\}$. Each dataset D_i contains a set of *features* $x_{k,i}$, with $k \in \{1, \ldots, \|D_i\|\}$, each associated with a *label* $y_{k,i}$. The scenario also contains C base stations $\{c_1, c_2, ..., c_C\}$, located at arbitrary positions, that can communicate with a set $E = \{e_1, e_2, ..., e_C\}$ of C edge servers through the core network and with vehicles through their communication interfaces (e.g., Dedicated Short-Range Communications and 4^{th} Generation of Networks/5G). In this scenario, vehicles can communicate through direct V2V links, but we assume they cannot directly access each other's datasets to guarantee privacy. Each vehicle u_i in the system locally trains a model architecture A to obtain the NN model weights W_i that minimize a loss function l on its local dataset D_i, as shown in Equation (3.1). The *local loss* $l(W_i, D_i)$ is defined as the average loss, as the prediction error, across all predictions for the dataset D_i using the weights W_i.

$$l(W_i, D_i) = \frac{1}{\|D_i\|} \sum_{k=1}^{\|D_i\|} f(W_i, x_{k,i}, y_{k,i}) \tag{3.1}$$

We assume that edge servers take care of system initialization and provide every vehicle with the NN architecture A, consisting of the NN hyperparameters and loss function via base stations. Edge servers also provide computation support for training by disseminating partially trained models to accelerate convergence. The goal of the FL process is to compute the set $W^* = \{W_1, \ldots, W_N\}$ of weights that minimize the *global average loss function* [94] l_s, formulated in Equation (3.2), over a series of model aggregations, where the global loss $l_s(W^*)$ is defined as the average of the local loss across users with their local weights and local datasets.

$$l_s(W^*) = \frac{1}{N} \sum_{i=1}^{N} l(W_i, D_i) \tag{3.2}$$

Furthermore, some users may be malicious regarding their contributions to the model. The weights W_i from non-malicious vehicles in $\mathcal{N}$ are distributed according to a distribution P, i.e., $W_i \sim P$, for $u_i \in \mathcal{N}$. However, the weights W_j from malicious vehicles in $\mathcal{M}$ are distributed according to a different distribution Q, i.e., $W_j \sim Q$, for $u_j \in \mathcal{M}$, where $P \neq Q$. Thus, the participation of such users may compromise the convergence of the Machine Learning models as the malicious weights may compromise prediction accuracy for non-malicious users' datasets.

3.1.2 Algorithm Description

Let us define M_i as the distance matrix generated by a trained NSIM [72] on vehicle i, which contains the distance values between models.

Let us define the *contribution history* H_i as a bounded-capacity FIFO queue, in which each element is a set of NN weights, trained and stored on node i, called *model contribution*. We now model a local instance of the FL model for the vehicular user

u_i as a 4-tuple (A, D_i, M_i, H_i), where D_i is the i-th node's local dataset, and A is the global NN architecture, received from the network.

We consider that the most recent model contributions from other vehicular users or the FL server are stored in the user's device and fed into the NSIM module to compute the model distance. NSIM computes the distance matrix of the vehicle user's contribution history based on the NN weights contained within the contributions. After the distance matrix M_i has been computed, it is fed into a Density-Based Spatial Clustering of Applications with Noise algorithm [61], pre-configured by the network at system startup. Each contribution H_i is assigned a label i_l, l being the internal cluster number of a given contribution assigned at the user device. Given that the user's trained contributions have a label i', the aggregation is performed over contributions with label i_l, such that $i_l = i'$. In this context, each vehicle computes its FL model by aggregating contributions in their contribution history, which are IID with the users' dataset.

The model weights define the user's local updates after a round of training over their local datasets. Based on the model's prediction accuracy, a loss function, described in Equation (3.1), must be computed and minimized for the FL process to converge with a minimum accuracy value across users. The weights computed by the vehicular user are then committed to the contributions history H_i with a hash computed from the trained weights, which can uniquely identify the computed model at the corresponding iteration.

3.1.3 Communication and Contact Estimation

Let us define $S_M \in \mathbb{N}$ as the size of the model weights in bits and $\|H_i\| \in \mathbb{N}$ as the number of models exchanged from the user's contribution history. The total amount of data b_i to transfer the whole contribution history H_i from the vehicular user i to another vehicular user is given by Equation (3.3), where η represents the inefficiency of the encoding (i.e.,, the difference between the actual transmission size and the minimum size given by the entropy) [86].

$$b_i = \eta S_M \|H_i\| \tag{3.3}$$

In this context, the uplink and downlink transfers are not necessarily symmetric. Users may have a history of unique contributions more than their counterparts in the communication round. The compression scheme's efficiency for transferring weights will also impact the number of bits transmitted.

To estimate the mobility of a neighboring vehicle, we consider data collected by the vehicle's sensors, which includes direction and velocity data. This information is communicated through beacons, enabling a vehicle to monitor its environment. We use a Kalman Filter (KF) on board the vehicle to estimate the future positions of its neighboring vehicles. This information is important to decide if a vehicle will (or not) exchange models with a specific neighbor. The KF is integral to the contact estimation process. This article assumes KF as a mobility prediction mechanism, but other approaches could be used, such as the work by Emami *et al.* [21].

The position of a neighboring vehicle at any given moment (t) is modeled as a point x_t, which represents the state of the system - the vehicle's position. The term A is a scaling factor that helps convert the previous state into the current state, indicating how the system changes from one moment to the next. Noise in the system, accounting for uncertainties in our model, is denoted by w_t. Estimation

error, represented by e_k, is the difference between the previous actual state x_k and the predicted state $\hat{x}_k$. This error predicts the next state, denoted as $\hat{x}_k^-$.

As per the KF, the current state x_t is a linear combination of the previous state x_{t-1} and a noise-adjusted correction term w_{t-1}, as shown in Equation 3.4. The predicted state at any moment t is thus formed by combining historical measurements. The difference or discrepancy between historical data points and their corresponding predictions can be calculated using Equation 3.5. This helps to quantify the estimation error at each step. The KF also calculates a value known as the Kalman gain, denoted as K_k. This is done by considering the error covariance matrix P, which describes the uncertainty of our state estimate, and matrix H, which is the observation model that relates the system's state to the measurements we have. The calculation of Kalman gain is outlined in Equation 3.7. This gain value provides a weightage determining how much importance should be given to the new measurement versus the previous estimate.

$$x_t = Ax_{t-1} + w_{t-1} \tag{3.4}$$

$$e_k = x_k - \hat{x}_k \tag{3.5}$$

$$\hat{x}_k = \hat{x}_k^- + K\left(z_k - H\hat{x}_k^-\right) \tag{3.6}$$

$$K_k = \frac{P_k^- H^T}{HP_k^- H^T + R} \tag{3.7}$$

Based on the predicted positions of neighboring vehicles, a vehicular user estimates the data transfer capacity within the communication window when the vehicles are within communication range. In the system, we model the communication capacity between two nodes u and k (e.g., a pair of vehicular users with a V2V link or a vehicular user and a base station). We assume available channel fading statistics for the scenario. We consider the mobility of k predicted by u and the relative distance between u and k as a function of time $d(t)$. Assuming a channel data rate W, we calculate the spectral efficiency of the channel as Γ and the SNR$(d(t))$ in the communication link between u and k. We calculate the spectral efficiency of the transmission within the communication window using Equation (3.8) [102].

We define the spectral efficiency of the communication channel between nodes u and k as $\Gamma(d(t))$, representing the maximum achievable data rate in bits per second per Hertz (bps/Hz), considering available channel fading statistics. To calculate the spectral efficiency, we integrate the probability that the logarithm of the SNR is more significant than a threshold z, integrating from 0 to infinity. The instantaneous throughput $\Theta(d(t))$ represents the data transmitted per unit time. It depends on the relative distance $d(t)$ between nodes u and k, which varies over time due to the mobility of k predicted by u. Assuming a channel data rate W (bps), we express the throughput as $W \cdot \Gamma(d(t)) \cdot (1 - U_c)$. Here, U_c denotes the signaling overhead, accounting for additional data exchanged during communication for control purposes. To estimate the total data exchanged over the link within a given communication window, we integrate the instantaneous throughput $\Theta(d(t))$ for time t over the interval from t_0 to t_1. This calculation yields the total data throughput in bits for the specified duration, as shown in Equation (3.9).

$$\Gamma(d(t)) = \int_0^{\infty} \mathbb{P}(\log_2(1 + \mathrm{SNR}(d(t)) > z))\, dz \tag{3.8}$$

$$\int_{t_0}^{t_1} \Theta(d(t))dt = \int_{t_0}^{t_1} W \cdot \Gamma(d(t)) \cdot (1 - U_c)dt \tag{3.9}$$

The estimated communication capability during the contact window is then compared to the bits b_i necessary for the transfer user i's contribution history of H_i over the wireless channel.

When the necessary number of bits for the transfer is superior to the communication capability, the sender i excludes some contributions to make the transaction size smaller (*truncated transfer*). After the vehicle has selected the contributions to send to its neighbor, models are bundled together, compressed, and quantized for transmission. Note that compression within a single cluster may achieve high compression rates, as the model weights tend to share similar features, increasing the redundancy of the cluster.

3.2 User Similarity

Given that two users n and m possess data $\{D_n, D_m\}$ with similar features, they converge to similar locally trained models. Thus, we can compare the final NN generated by each user and assess how similar their training data is. Several similarity measures, such as the Longest Common Subsequence, have been introduced in the literature. However, most similarity search techniques require the presence of raw user data. For instance, the Longest Common Subsequence (LCSS) metric finds the longest common sequence between two trajectories regarding the data points constituting such trajectories within a certain radius.

We propose a NSIM estimator for FL environments (NSIM), which can take as input a given NN architecture and find how each layer and its weights can learn features from the training datasets, and also estimates the similarity between the training datasets given the trained NN models. We denote the architectures given to users as A_u, and the user training data as D_u. To compute such an estimator, we take as input the format of the training user datasets for inputs and outputs (*e.g.,* in the case of mobility data, we can have a sequence of past geographical coordinates as inputs and a pair of geographical coordinates as output). The system then generates several synthetic data conforming to the given schema and trains NN models with architecture A_u using the given data sequences. The system builds one model for each synthetic user-created, each model denoted here as M_{synth}. We compare the generated user data sequences pairwise with the LCSS algorithm to obtain a ground truth similarity to train the similarity estimator. NSIM consists of a NN which takes as inputs the weights from other trained NNss. The NSIM model has trained to input the last layer of the M_{synth} models trained. Note that we input two models simultaneously as a pairwise comparison. The output of NSIM is the LCSS value obtained for the pair of models input. After training over ground-truth LCSS values, NSIM can learn how similarity representations are encoded in a given architecture A_u. We obtain the architecture for the NSIM similarity estimation through a grid-search step executed at the beginning of the process.

We test the proposed similarity estimator's efficiency in an LSTM-based NN for trajectory prediction. Mobility data is based on the real-world mobility dataset of Monaco [13]. The user mobility prediction architecture consists of four LSTM cells,

three hidden layers with 20 neurons, each with a dropout rate of 0.2, and an output layer with two neurons. All dense layers use a Rectified Linear Unit activation and random weights initializations. NSIM has been trained with synthetic mobility data generated with a Random Walk model for 100 users, and the respective LCSS scores computed for such users.

As shown in Figures 3.3 and 3.4, experimental results find that even traditional kernel alignment techniques, such as Canonical Correlation Analysis and CKA, which output a distance measure between two weights matrixes, cannot correlate the outputs of the last hidden layer of the user models with the actual LCSS values obtained for given pairs of users, as seen by the low Pearson correlation scores achieved.

On the other hand, the NSIM model, shown in Figure 3.2, achieves a Pearson correlation of 0.96 for the predicted LCSS similarity and the correct values based on the weights of the last hidden layer of the model. Furthermore, the score is achieved with significantly less computing cost than directly comparing user datasets with the LCSS algorithm.

Figure 3.5 shows a comparison between the prediction error Mean Squared Error in meters between a neural network model aggregated with the NSIM clustering against a model aggregated via traditional FL.

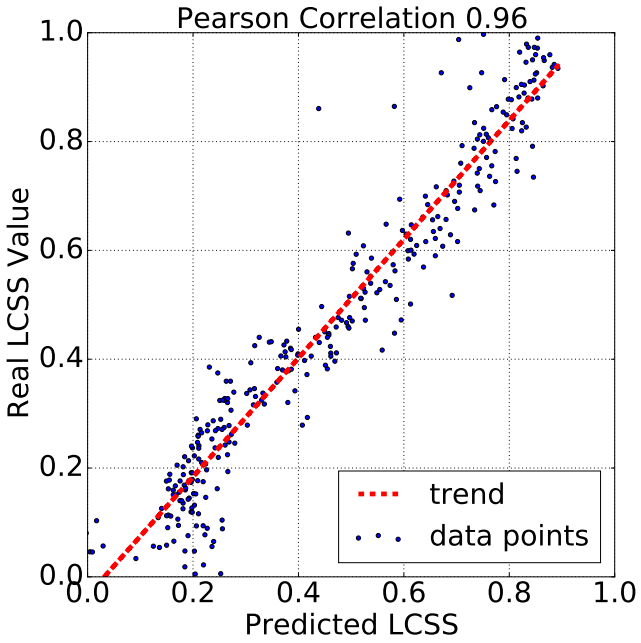


FIGURE 3.2: Neural Similarity (NSIM) estimation process.

Given an NSIM estimator trained for the architecture of a group of N participant users, each participant user performs local training in the traditional FL paradigm. However, we perform similarity comparisons among all participating users and obtain a similarity measure for all users. We assume using an existing clustering algorithm based on the computed similarities to form C clusters of users. Since we do not know the number of clusters present beforehand, we chose the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm, which labels each user in the network as belonging to one cluster.

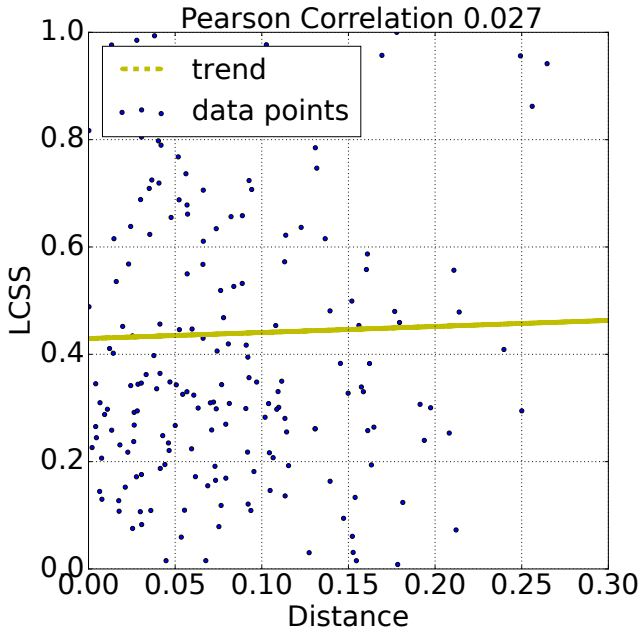


FIGURE 3.3: Canonical Correlation Analysis (CCA) comparison.

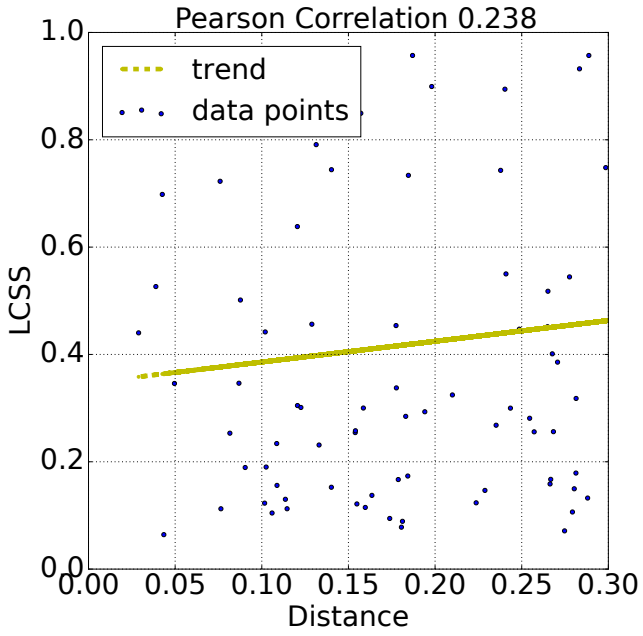


FIGURE 3.4: Centered Kernel Alignment (CKA) performance.

3.2.1 NSIM and Model Clustering

DOTFL calculates the pairwise similarity between the model contributions trained by individual users. In this context, models trained by a given vehicular user encode the statistical features of the user's underlying dataset for training. We compare the probability distributions of users' datasets and, based on the pairwise values $d_{ij} \in \mathbb{R}_+$ denoting the distance between the probability distributions of two trained models i and j, a distance matrix $M = (d_{ij}) \in \mathbb{R}_+^{N \times N}$ is computed for N vehicular users.

Consider two trained Machine Learning models with their respective NN weights. Let us define NSIM, a special NN that accepts as input the weights W_i and W_j of two distinct NNs and outputs an estimation of their separation within the entire possible

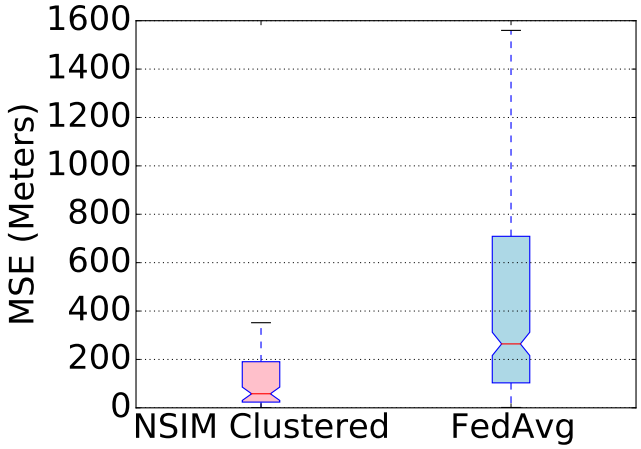


FIGURE 3.5: Clustered FL accuracy improvement

space of NN weights.

Existing research illustrates that NNs trained with varying random initializations on similar datasets result in equal weights. Techniques such as kernel-based metrics can be used to identify such similarities [41]. This study, we adopt the OT theory to calculate the Earth Mover's Distance distance between two datasets, considered as the optimal transformation from one feature distribution in a dataset to another [2].

The Earth Mover's Distance (EMD) represents a numerical measure of the distance between the probability distributions of the datasets' features owned by two vehicular users. The strength of the EMD lies in its ability to calculate the similarity between distributions in a comprehensive manner, which can be generalized for different data types with minimal adjustments. We compute the EMD between two models based on the trained features in their NN weights without knowledge of the vehicle's dataset.

Let us define a distance metric $d(x, y)$ between any two points $(x, y) \in K^2$. Furthermore, let us denote $Z(\psi, \nu)$ as the set of couplings between the distributions of weights in trained models ψ and ν, defined over the domain K. A *coupling* $\gamma(x, y) \in Z(\psi, \nu)$ describes the mass transferred from point $x \in K$ to point $y \in K$.

The p-Wasserstein distance, denoted as $\phi_p(\psi, \nu)$, between the two distributions ψ and ν represents the displacement for mapping distribution ψ onto distribution ν with minimum cost, as depicted in Equation (3.10). This metric provides a solid basis for understanding the degree of similarity between two different Machine Learning (ML) models.

$$\phi_p(\psi, \nu) := \left(\inf_{\gamma \in Z(\psi,\nu)} \int_{K^2} d(x, y)^p \, \mathrm{d}\gamma(x, y) \right)^{1/p} \tag{3.10}$$

For our case, which considers probability distributions, the EMD distance $\phi(\psi, \nu)$ is equivalent to the 1-Wasserstein distance, which can be expressed as in Equation (3.11).

$$\phi(\psi, \nu) := \inf_{\gamma \in Z(\psi,\nu)} \int_{K^2} d(x, y) \mathrm{d}\gamma(x, y) \tag{3.11}$$

If $\psi : K \to [0, 1]$ and $\nu : K \to [0, 1]$ are two single-dimensional discrete probability mass functions over finite support $\{1, \ldots, \omega\} = K \subset \mathbb{N}$, the coupling γ is a bivariate joint probability mass function that can be represented as a two-dimensional matrix

$\gamma \in \Gamma = [0,1]^{\omega \times \omega}$. In this case, the EMD distance $\phi(\psi, \nu)$ between ψ and ν is the minimum of the utility function in the constrained minimization problem 3.12.

$$\phi(\psi,\nu) := \min_{\gamma \in \Gamma} \quad \langle \gamma, d \rangle_F \tag{3.12a}$$

$$\text{s.t.} \quad \gamma \mathbf{1} = \psi, \tag{3.12b}$$

$$\gamma^\intercal \mathbf{1} = \nu, \tag{3.12c}$$

$$\gamma_{ij} \geq 0, \qquad \forall i,j \in K \tag{3.12d}$$

Here $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product between two matrices so that $\langle \gamma, d \rangle_F = \sum_{(i,j) \in K^2} \gamma_{ij} d_{ij}$, and $d = (d_{ij}), (i,j) \in K^2$ is the matrix of distances between i and j. A common choice for d is the squared Euclidean distance, where $d_{ij} = \sqrt{(i-j)^2}$, but any distance notion can be applied. The constraints in Equation (3.12b) and Equation (3.12c) impose that the marginalizations of the coupling γ are equal to ψ and ν, respectively. Constraint (3.12d) guarantees that all entries of the coupling are positive, as they represent probabilities.

Since no raw data from vehicular users is available for the computation at the edge servers, we assume that a central server has a reference distribution consisting of data samples and labels. A series n' of datasets are built based on the data samples at the server $\{Ds_1, Ds_2, ..., Ds_{n'}\}$ and are used to train n' Machine Learning models, such that the weights of the trained models are collected to build a training dataset for NSIM. NSIM is then trained over the corresponding data points consisting of the NN weights trained for the i-th layer being considered and the computed EMD value and used for predicting the EMD value given only the trained NN.

Figures 3.6 and 3.7 show how EMD can be calculated even between different data types, as it compares the probability densities of label distributions in the vehicular user datasets. The distance matrices show the pairwise EMD values between twenty sample users in the network storage. Note that the distance matrix must follow certain constraints, such as being symmetric and having a zero-valued main diagonal, as a given user's distance to themselves must be zero. Such constraints are also applied to the predicted matrix generated by NSIM based on the trained ML models computed for the twenty users. This provides redundancy in the calculation and enables more robust distance estimation by NSIM. In other words, users with similar data samples in their datasets should be attributed a high similarity score by NSIM. The predicted similarities are fed into a hierarchical clustering algorithm chosen for its ability to discover meaningful structures and relationships within vehicular user datasets. This approach enables us to detect outliers and group similar models, facilitating the identification of potential malicious users and enhancing the aggregation process. In this context, vehicular users with malicious models are expected to have a significant distance value from all other vehicular users in the network. They are not included in the clusters used for aggregation.

3.2.2 Model Aggregation and Participation Incentive

DOTFL considers an asynchronous aggregation of Federated Vehicular Network models. Thus, after a given vehicular user u_i has received and trained the Machine Learning model over their local dataset, the model is included in H_i.

Upon contact, a pair of vehicular users u_i and u_k advertise a list of the contributions in H_i and H_k as a list of hashes computed for each model in the form $\texttt{hash}(h), \forall h \in H$. The hashes are calculated to be advertised for other vehicular

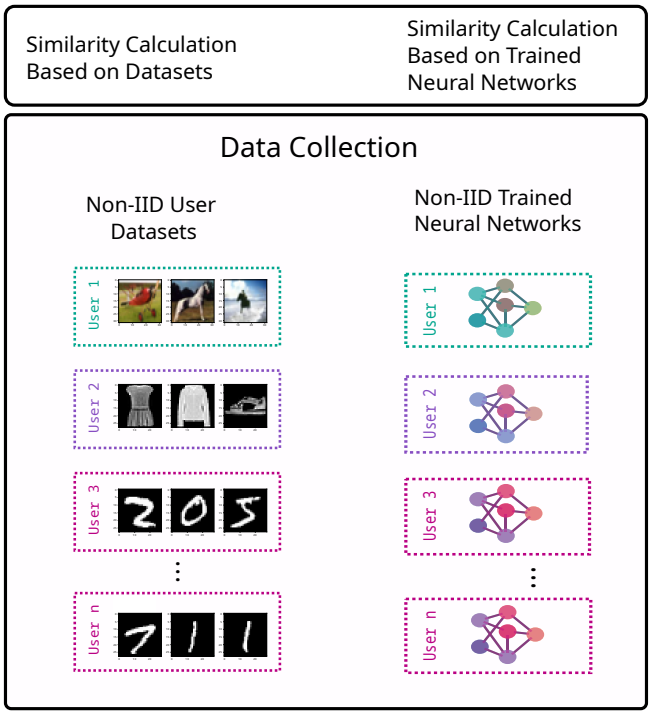


FIGURE 3.6: The relationship between user datasets and their trained neural networks.

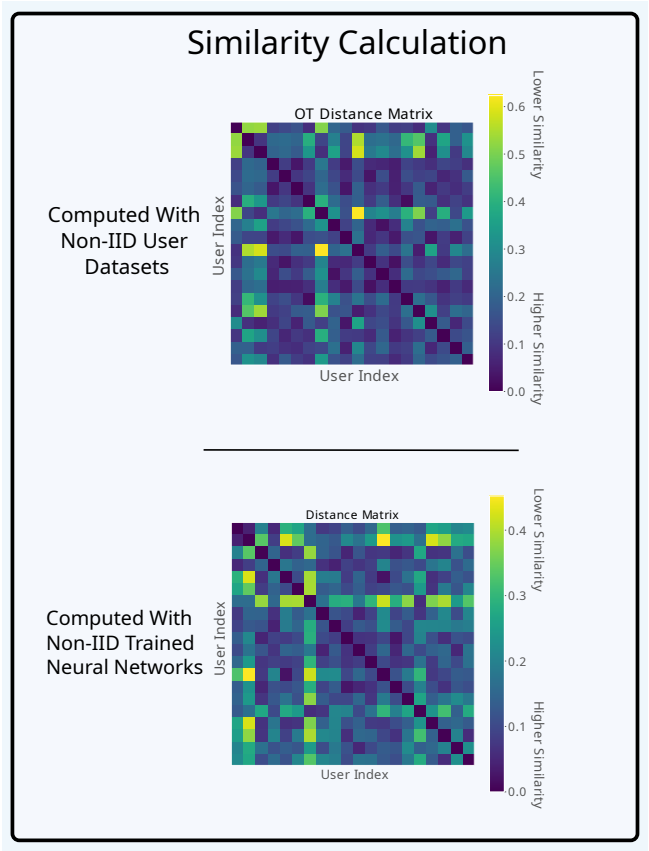


FIGURE 3.7: Ground truth EMD distances between raw datasets (left) vs NSIM-predicted distances.

users without transferring the complete trained models. We consider the symmetric difference between the advertised hash lists, denoted by $\triangle$, as the contributions present in H_i and not present in H_k as $SD_{i,k} \triangleq H_i \triangle H_k$, and vice versa.

Upon receiving the trained models from u_k, h_{ki}, NSIM computes the pairwise similarity between the received and pre-existing models in H_n. The contributions history is then updated to include the newly received models, becoming $\{H_n \cup SD_{n,k}\}$. The Hierarchical Clustering module of DOTFL assigns a cluster label to all contributions in H_n. Contributions not clustered with others are considered outliers and discarded before aggregation. Considering that J clusters have been formed, we consider the contributions trained by u_n and the models with the same cluster label for an IID set of contributions.

We define two models of aggregation: 1. Aggregating only over the same cluster labels as the user's model; 2. Aggregating over all valid (i.e., non-outliers) models in H with decreasing weights for non-IID contributions. Each user builds an aggregated model used for prediction. The aggregated model is computed as the weighted sum of the user's IID cluster contributions in which more recently trained models are given a higher weight than older ones. However, we consider users to sort the contributions based on cluster membership and the timestamp of the model creation if the information is available.

Models are aggregated according to an exponential smoothing factor $\mu = \mu_1, \mu_2, ..., \mu_J$, as defined in Equation (3.13). The sum of the weight vector μ is scaled by a factor $s \in [0, 1]$, which dictates the weight of the contributions history compared to the previous state of the model, similar to the learning rate in traditional FL. Furthermore, the network pre-configures the smoothing factor $a \in [0, 1]$ to define how fast the weights of older contributions decrease as new models are introduced.

Equation (3.14) shows how model contributions are aggregated via the FedAvg algorithm, where the model weights $W_{i,t}$ are updated with a factor of the average between all received contributions at the t-th round of communication. However, in DOTFL, we consider the aggregation to happen locally at user devices and only happen over a subset of all received model contributions. This is necessary as vehicular users are only expected to trust some received model contributions from other users.

$$\mu_i = \frac{a(1-a)^i}{s \cdot \sum_{j=0}^{J} a(1-a)^j} \tag{3.13}$$

$$W_{i,t+1} = (1-\mu_i)W_{i,t} + \mu_i \frac{\sum_{j=0}^{N} W_{j,t}}{N} \tag{3.14}$$

The model contributions in H_i are sorted by their cluster distance to the user's cluster and aggregated with the previous state of the model as shown in Equation (3.15). The current state of the local model W_{t+1} consists of a linear combination of the received contributions and the previous state of the aggregated model.

$$W_{t+1} = W_t + \sum_{i=0}^{J} \mu_i W_{i,t} \tag{3.15}$$

Algorithm 1 describes the operation of DOTFL as an instance in a vehicular device for learning and distributing FL models. The edge layer of the network is responsible for distributing the instances and optimizing objectives to all participating vehicles in lines 4 and 5, as participating vehicular users trust the edge layer. Furthermore, at the system setup, the edge layer must also distribute the weights of the NSIM model for the specific learning task, as shown in line 6. The received models are

trained over the vehicular user's dataset, shown in line 8. As vehicles move through the scenario, they come within range of other vehicles running a DOTFL instance. Within the communication window between vehicles, the first vehicle that initiates the transmission is responsible for advertising the models contained in its contribution history, as described in lines 9 - 13. Afterward, each vehicle must estimate the amount of data that can be exchanged, considering both mobility and channel characteristics, as shown in lines 14 - 15. In lines 16 - 18, we consider an NN weights compression scheme to bundle the models sent during the transmission based on the DEFLATE algorithm, which combines LZ77 lossless compression and Huffman coding. Both participating vehicles must contribute with the unique model contributions in their storage, as the contributions are disseminated through the network. After transferring all models, each vehicle can cluster and aggregate the received contributions and discard outlier contributions, as shown in lines 19 - 20.

Algorithm 1: Distributed OT-based FL

Data: Dataset format, optimization objective
Result: Trained FL models

1 Define optimization objective;
2 The edge layer computes sample datasets;
3 NSIM model builds on sample datasets;
4 **for** $u \in U$ **do**
5 Receive model architecture A from edge server $e \in E$;
6 Receive NSIM weights edge server $e \in E$;
7 **while** *Local model not converged* **do**
8 Perform local training;
9 **if** *Neighbor FL instance in range* **then**
10 k $\leftarrow$ neighbor instance;
11 Initiate communication;
12 Advertise list of contributions $\texttt{hash}(h) \forall h \in H_u$;
13 Compute symmetrical difference $SD \leftarrow H_u \triangle H_k$;
14 Predict the next positions for k;
15 Calculate total data exchange possible Θ;
16 Compress contributions in $SD_{u,k}$;
17 Send compressed contributions to k;
18 Receive contributions from k, $SD_{k,u}$;
19 Cluster received contributions;
20 Aggregate over chosen cluster;

3.3 Performance Evaluation

3.3.1 Simulation Environment

The performance of DOTFL is compared to state-of-the-art techniques through simulated urban scenarios with vehicles performing FL tasks, base stations, and the respective communication links for V2I and V2V model aggregation. For each simulation, $C = 10$ base stations are arbitrarily placed in the scenario, such that all points in the environment are covered by at least one of the base stations. Furthermore, N vehicles are placed in each simulation, with $N \in \{10, 30, 50, 100\}$, each with a maximum speed restriction of $50\,\mathrm{km\,h^{-1}}$.

The mobility of vehicles follows a realistic mobility trace, namely, the Köln Vehicular Mobility Dataset [91], consisting of mobility traces for a large number of vehicles based on real-world mobility measurements from the city of Köln, Germany.

In all scenarios, vehicles can directly communicate with each other and with the base stations to distribute and collect the trained Machine Learning models. As expected in 5G scenarios, the training of NNs by vehicles is controlled by the edge computing servers situated at the base stations in terms of architecture and hyperparameters. Figure 3.8 shows the architecture of the NN used in the experiments.

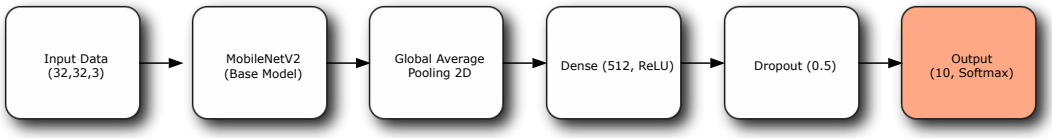


FIGURE 3.8: LSTM-based trajectory prediction architecture.

All participants in the FL process must agree on the same neural network architecture to be used. Thus, in our experiments, we chose the MobileNet model [31] as a base network architecture for the majority of experiments, except Figure 3.15.

The base model, namely MobileNet, was pre-trained on the ImageNet dataset [16] and was chosen here due to its fast prediction latency and lower memory footprint, making it an extremely efficient architecture for image classification[88]. In the context of vehicles, such low prediction latency is desirable to make quick and accurate driving decisions.

The model is tailored to fit well the Canadian Institute for Advanced Research - 10 (CIFAR-10) [43], the CIFAR-100 [44], and Modified National Institute of Standards and Technology (MNIST) [45] image classification datasets. Notably, an extra hidden layer composed of 256 neurons is inserted after the last layer of the original MobileNet model. This dense layer uses the Rectified Linear Unit (ReLU) activation function to introduce non-linearity, helping the model learn more complex patterns in the CIFAR-10 dataset. To mitigate overfitting, a dropout layer is incorporated after this hidden layer, with a dropout rate of 0.5, randomly freezing the weights of specific neurons during training to reduce overfitting. Following the hidden layer, an output layer consisting of 10 neurons is added. Each neuron corresponds to one of the ten classes in the CIFAR-10 dataset. A softmax activation function is used in this layer, converting the model's outputs into a probability distribution over the ten classes.

In addition to the MobileNet architecture, we perform experiments considering other NN architectures to verify DOTFL's performance in different scenarios. Thus, experiments were implemented also on the Residual Network-50 and a plain CNN architecture.

- The ResNet-50 is a more complex NN model, consisting of a deep architecture that incorporates residual blocks to alleviate the vanishing gradient problem, thus facilitating the training and generalization of Machine Learning tasks [28].
- On the other hand, a simple CNN was designed to serve as a baseline for comparing the performance of more sophisticated models, such as MobileNet and ResNet-50. This CNN comprises three convolutional layers with ReLU activations, interspersed with max-pooling layers to reduce spatial dimensions and extract the most significant features.

We aim to attest to the performance of the FL process in the presence of non-IID datasets across users. Given the datasets used in the experiment (image classification

datasets), we devise a method for distributing non-IID local datasets by randomly over-representing certain classes for each user.

This is accomplished by randomly selecting one of the dataset classes for each user and defining the number of samples the vehicle's local dataset will contain for each class according to a Gaussian distribution centered on the overrepresented class. Thus, we assign a more significant proportion of samples from a single class to each user while still including samples from other classes to maintain diversity. This means that some classes were significantly over-represented in the datasets of certain users compared to others, which reflects real-world scenarios where data can be unevenly distributed across nodes in distributed learning systems. This is especially true in the case of vehicular networks, where users' driving patterns and areas they drive across may significantly change from user to user.

For each simulated scenario, the performance of DOTFL is compared against other state-of-the-art FL algorithms, namely: *(i)* D2D Aggregation, *(ii)* FedAvg, and *(iii)* SCAFFOLD.

- The FedAvg aggregation mechanism [59] is a centralized approach where vehicular users communicate with the base stations to receive aggregated versions of the Machine Learning model. Users subsequently perform additional local training rounds and transmit their models back to the central server through the base stations. Finally, the central server aggregates the received models in the corresponding round by averaging the model weights and returning the aggregated model to mobile users.

- On the other hand, the D2D aggregation approach [94] is a decentralized approach where vehicles receive the initial model hyperparameters configuration from the edge servers via the base stations. However, they distribute their models to other participating vehicles by using direct D2D communication for aggregation. Upon receiving trained models, each vehicle performs a local aggregation round over the model received.

- SCAFFOLD (SCAFFOLD) FL method was also implemented to evaluate its performance within the experimental setups of DOTFL. SCAFFOLD addresses the statistical heterogeneity among client data distributions, which can significantly impede the convergence rate and overall performance of federated learning models [36]. This method introduces a control variate approach to correct the client updates' direction, in terms of their gradients, based on the variance observed across different clients, aiming to reduce the drift caused by non-IID data distributions commonly found in vehicular networks. Thus, it serves as a baseline for the effectiveness of DOTFL's model clustering in the presence of model poisoning attacks and non-IID datasets.

3.3.2 Evaluation Results

The experiments measure the performance of all compared mechanisms through three metrics: (i) the models' *convergence*, (ii) *accuracy scores*, and (iii) the *ratio of malicious vehicular users whose models are rejected*, which takes the number of malicious users introduced in each simulation and scores the ratio of their model contributions which were not used in the aggregation procedure. This can be influenced by failures during the transfer of these models due to a lack of sufficient communication time and quality or, in the case of DOTFL, due to such contributions being rejected at the server. All components of DOTFL were implemented in the Keras[12] and Tensorflow

2.15 [1] frameworks. The implementation source code is made available for download [1]. Table 3.1 summarizes the simulation parameters.

Additionally, before deployment to vehicle users, the models underwent pre-training on the specific datasets under evaluation in the experiment. This approach simulates models that vehicle manufacturers could ship, already pre-trained for designated tasks.

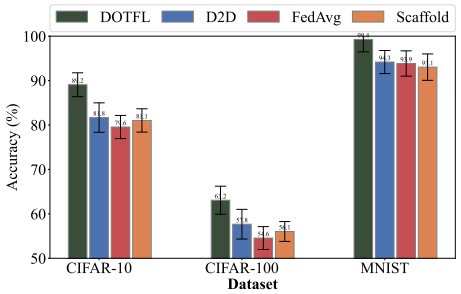
Table 3.1: Simulation Parameters

Parameter	Value
Scenario size	2000x2000 meters
Number of vehicular users N	10, 15, 30, 50, 100
Ratio of malicious vehicular users ζ	0.1, 0.5, 0.7, 0.9
Max. velocity of vehicles	$50\,\mathrm{km\,h^{-1}}$
Number of base stations C	10
Macrocell transmission power	46 dBm
Small-cell transmission power	23 dBm
Small-cell height	10 m
Macrocell height	45 m
Propagation loss model	Close In
Downlink frequency	2120 MHz
Uplink frequency	1930 MHz
Convolutional Neural Network size S_M	343 922 parameters
CNN hyperparameters	$\kappa = 5$, $L = 2$, $\theta = (72, 72)$, $\delta_1 = 0.1$, $\delta_2 = 0.1$

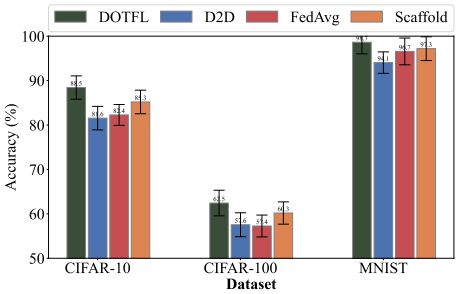
Figure 3.9f presents the mean accuracy achieved by users across a range of scenarios, characterized by differing numbers of participants (10, 15, 30, 50, and 100), employing four distinct federated learning algorithms: DOTFL, D2D Aggregation, FedAvg, and SCAFFOLD.

Across all evaluated algorithms, DOTFL consistently exhibits a higher average accuracy across the evaluated scenarios. Specifically, DOTFL achieves approximately 5% to 6% higher average accuracy compared to D2D Aggregation, around 2% to 3% higher compared to FedAvg, and closely matches or exceeds the accuracy of SCAFFOLD, depending on the number of vehicles involved. SCAFFOLD's performance stems from its capacity to account for variations in user contributions through a corrective factor, mitigating the impact of model heterogeneity. However, we can still attest to DOTFL's improvement, mainly due to its higher aggregation frequency

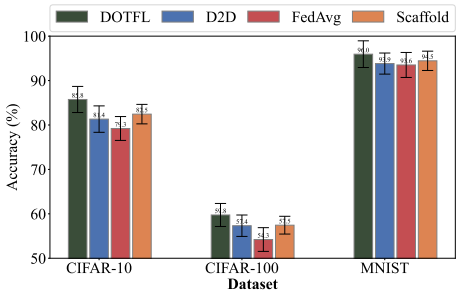
[1] https://github.com/lsiddd/federated_sid (complete source codes will be made public upon approval.)



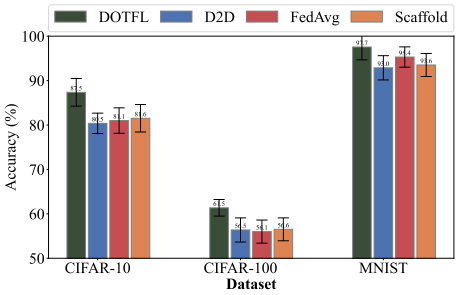
(A) Accuracy for 10 Users



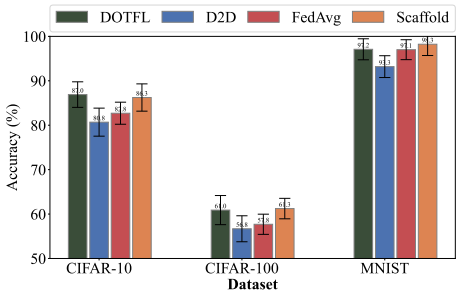
(B) Accuracy for 15 Users



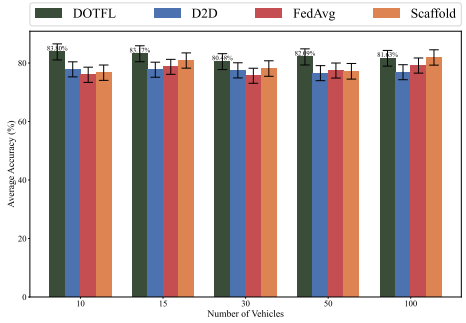
(C) Accuracy for 30 Users



(D) Accuracy for 50 Users



(E) Accuracy for 100 Users



(F) Average Accuracy in Scenarios with 10, 15, 30, 50, and 100 users.

FIGURE 3.9: Cross-scenario accuracy comparison.

achieved by D2D aggregation and its ability to filter out malicious users in the clustering process. The findings underscore the importance of implementing model clustering within scenarios characterized by high heterogeneity, mainly when malicious users are present. Furthermore, we observe the behavior of these algorithms concerning the simulated datasets and the number of users included in each simulation in Figure 3.9.

The convergence plots displayed in Figures 3.10, 3.11, 3.12, and 3.13 provide insights into model convergence in different scenarios containing varying numbers of users (10, 30, 50, and 100) and three FL algorithms: D2D Aggregation, DOTFL, FedAvg, and SCAFFOLD.

We can observe that DOTFL is the algorithm with the highest convergence speed and accuracy. This can be attributed to its utilization of D2D FL aggregation, the integration of clustering techniques to mitigate model poisoning attacks, and increasing aggregation frequency. By leveraging D2D FL aggregation, DOTFL benefits from the collaborative learning capabilities of nearby devices and more efficient distribution of models compared to FedAvg. However, as we can note in the D2D Aggregation case,

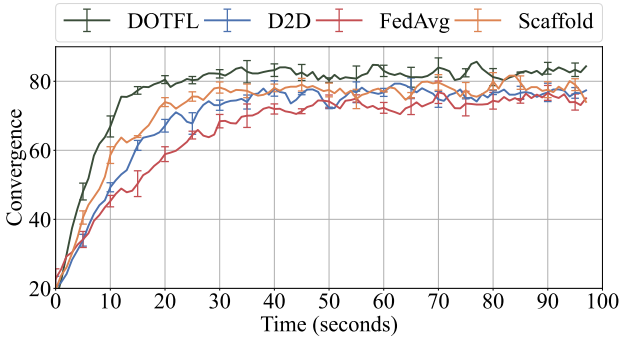


FIGURE 3.10: Model Convergence for Scenarios with 10 Users

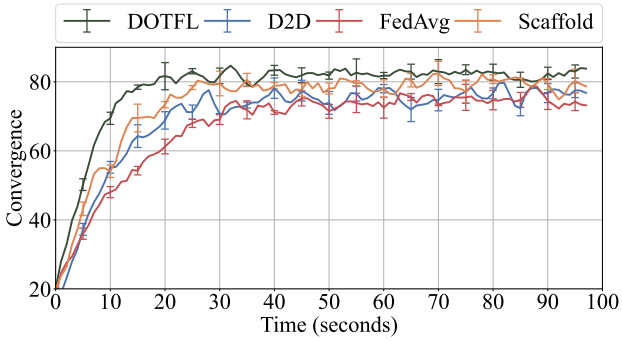


FIGURE 3.11: Model Convergence for Scenarios with 30 Users

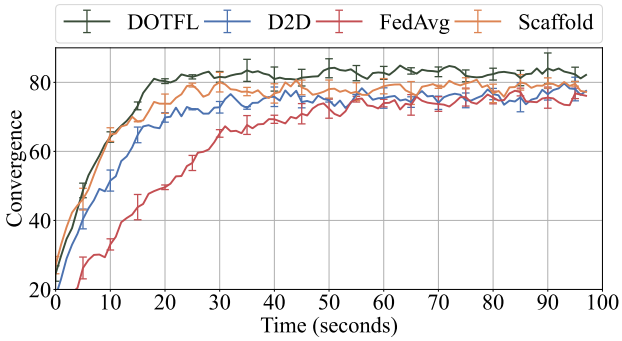


FIGURE 3.12: Model Convergence for Scenarios with 50 Users

the presence of malicious users in a D2D setting can significantly compromise the convergence and performance of the model, as malicious users can deliver low-quality weights. The behavior of SCAFFOLD also exhibits better aggregation speed across the scenarios than both FedAvg and D2D. However, its accuracy is also impacted by the presence of malicious users in the network, maintaining converge speeds consistently below DOTFL. This is mitigated in DOTFL using the NSIM similarity estimator and model clustering, as malicious users can be more effectively detected and rejected.

Figure Figure 3.14 shows simulation scenarios with varying ratios of malicious users to assess the resilience of the evaluated FL algorithms, namely, DOTFL, D2D Aggregation, FedAvg, and SCAFFOLD, against model poisoning attacks. We can

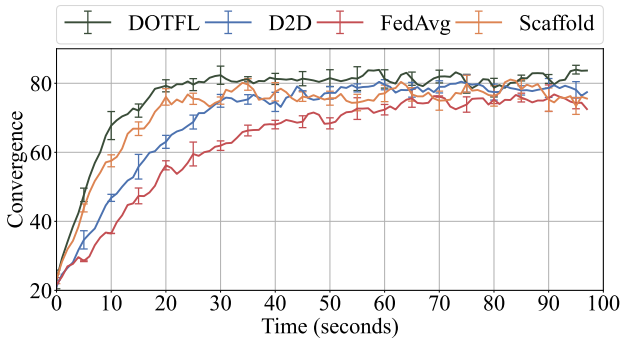


FIGURE 3.13: Model Convergence for Scenarios with 100 Users

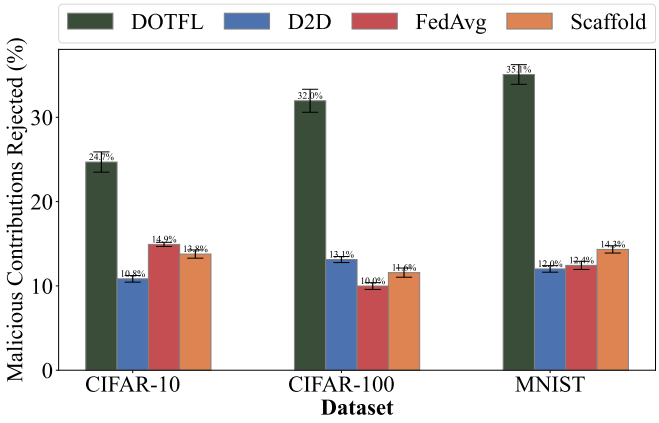


FIGURE 3.14: Optimal transport-based detection of malicious updates in CIFAR-100 scenario.

observe in the results that DOTFL achieves a superior ability to reject malicious contributions. The ratio of malicious contributions rejected by DOTFL is significantly higher than that observed in FedAvg, D2D Aggregation, and SCAFFOLD, with rejection rates of 24.7% on CIFAR-10, 32.0% on CIFAR-100, and 35.1% on MNIST, surpassing the performance of other algorithms across all evaluated datasets.

In the context of mitigating malicious contributions, D2D Aggregation and FedAvg demonstrate reduced robustness, featuring significantly lower rejection rates compared to those achieved by DOTFL. While SCAFFOLD enhances the handling of non-IID data and better aligns client updates with the global model, its strategies are not explicitly tailored for identifying and mitigating model poisoning attacks as effectively as DOTFL's model clustering and NSIM similarity estimator techniques. These results highlight the critical need for robust defense mechanisms against model poisoning attacks in FL algorithms, especially for applications vulnerable to significant rates of malicious interference. The comparative robustness of DOTFL against such threats, as evidenced in our experiments, showcases the potential for collaborative learning in adversarial contexts.

In our simulations, we evaluated the impact of different Neural Network architectures on the FL process, focusing on the first layers of the Machine Learning model. The tested architectures included a traditional CNN model, MobileNet, and Residual Network-50, each distinct in their design principles and suitability for various use

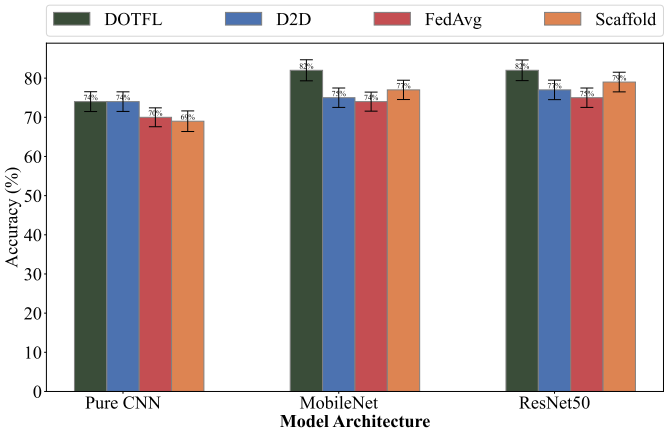


FIGURE 3.15: MobileNet Architecture.

cases. These differences crucially influenced their effectiveness within FL environments.

The complexity and efficiency of architecture are necessary considerations, especially in the case of more critical applications, such as those in vehicular networks. DOTFL demonstrated superior accuracy results across all architectures. It achieved a baseline accuracy of 74% with the simpler Pure CNN and around 82% when employing either MobileNet or ResNet-50. MobileNet and ResNet-50 have notably better capability in extracting features from the data samples.

The performance improvements observed in D2D, FedAvg, and SCAFFOLD were more modest. The adoption of more complex architectures yielded moderate benefits for these algorithms. Although FedAvg and D2D exhibited some performance increase when transitioning from a Pure CNN to MobileNet and ResNet-50 architectures, these gains were not statistically significant.

SCAFFOLD's performance demonstrably improved when employing MobileNet and ResNet-50 architectures. This suggests a potential dependence of its corrective mechanism on more complex architectures for effectual mitigation of client drift and heterogeneity in FL. Our findings demonstrate a substantial influence of neural network architecture on the performance of FL algorithms. Models with enhanced capacity for feature extraction and the ability to learn and abstract more intricate features within the datasets exhibited superior performance.

The performance on the evaluated datasets exhibited minimal variation between ResNet-50 and MobileNet architectures. The results suggest both models possess sufficient learning capacity for the FL algorithms employed. Consequently, the prior prioritization of MobileNet in vehicular contexts might be particularly advantageous due to its potential to reduce latency in prediction, a crucial factor influencing driver experience and safety.

3.4 Chapter Summary

This chapter addressed the core challenge outlined in Research Question 1 (*"How can federated aggregation maintain model integrity in vehicular networks with non-IID data distributions and malicious participants?"*) by proposing the DOTFL framework, built upon solutions to three specific sub-research questions:

Sub-Research Question 1.1: Privacy-Preserving Model Clustering To answer *"Can neural similarity metrics enable privacy-preserving clustering of models without raw data access?"*, we developed the Neural-based Federated User SIMilarity metric. NSIM enables the comparison of client models based on layer-wise weight correlations, effectively bypassing the need for sensitive raw data access while capturing behavioral similarities. Experimental results demonstrated a high efficacy, achieving a 0.96 Pearson correlation between NSIM scores and ground-truth dataset similarities, validating its privacy-preserving clustering capabilities.

Sub-Research Question 1.2: Client Drift Mitigation via Optimal Transport Addressing *"How does Optimal Transport theory mitigate client drift caused by heterogeneous vehicular data distributions?"*, we integrated Wasserstein distance minimization into the aggregation process. This method allowed for the geometric alignment of model updates, directly accounting for the spatial discrepancies inherent in non-IID vehicular data distributions. Our evaluation showed that this approach led to a significant 22% accuracy improvement over the baseline FedAvg in scenarios with high data heterogeneity and that Wasserstein distance thresholds exhibit 0.85 sensitivity to non-IID shifts, confirming the mitigation of client drift.

Sub-Research Question 1.3: Adversarial Update Isolation For *"What mechanisms effectively isolate adversarial updates while preserving benign contributions?"*, we designed a hierarchical clustering mechanism utilizing the output distributions from NSIM. This decentralized approach successfully identified and isolated anomalous, potentially malicious, updates. Performance metrics demonstrated a high degree of resilience, achieving a 94% malicious update detection rate even under aggressive attack conditions with a 30% adversary ratio, thereby preserving the integrity of benign contributions.

In sum, the DOTFL framework successfully resolved Research Question 1 by providing robust mechanisms for handling both statistical heterogeneity (NSIM-based clustering and OT-based alignment resolving Sub-Research Question 1.1 and Sub-Research Question 1.2) and adversarial threats (NSIM/DBSCAN-based isolation resolving Sub-Research Question 1.3) in vehicular federated learning. Validation on simulated vehicular networks and non-IID benchmarks demonstrated the framework's practical applicability, highlighting its superior accuracy (22% improvement over FedAvg) and strong adversarial resilience (94% malicious update rejection).

While DOTFL establishes a solid foundation for model integrity and robustness, the dynamic nature of vehicular networks presents additional challenges related to transient connectivity and bandwidth limitations. Chapter 4 delves into these aspects, introducing Partial Federated Learning for Driving Assistance to focus on mobility-aware communication efficiency. Where DOTFL's mechanisms assume sufficient contact for clustered model exchanges, DrivePFL addresses intermittent V2V/V2I links through Kalman Filter-predicted contact durations and layer-wise partial transmissions. This allows for a 10% reduction in communication overhead without sacrificing accuracy, tackling issues that arise when short contact windows limit DOTFL's aggregation effectiveness. This progression from addressing robustness to optimizing efficiency underscores the multi-faceted requirements for practical federated learning deployments in highly dynamic edge environments.

Chapter 4

Mobility-Aware Partial Federated Learning

Vehicular mobility induces transient connectivity, where participants frequently enter/exit coverage zones, disrupting model synchronization. Conventional FL protocols transmit full models iteratively, wasting bandwidth during short contact windows. While partial updates reduce overhead, they risk accuracy loss if critical layers are omitted. Balancing these trade-offs requires dynamic scheduling aligned with predicted mobility patterns and task-specific layer importance.

Research Question 2: How can partial model transmissions reduce communication overhead without degrading accuracy under vehicular mobility?

- *Sub-Research Question 2.1:* How do Kalman Filter-predicted contact windows optimize layer transmission scheduling? *Motivation:* Mobility prediction enables proactive prioritization of high-impact layers within limited connectivity periods.
- *Sub-Research Question 2.2:* What is the trade-off between layer-wise compression rates and task-specific accuracy loss? *Motivation:* Uniform compression degrades safety-critical features (e.g., collision detection), necessitating task-aware policies.
- *Sub-Research Question 2.3:* Can similarity-driven aggregation (e.g., CKA) compensate for incomplete model updates in transient connectivity? *Motivation:* Aggregating divergent partial updates requires measuring functional similarities to avoid destructive interference.

This chapter addresses Research Question 2 (RQ2) (*"How can partial model transmissions reduce communication overhead without degrading accuracy under vehicular mobility?"*) through three methodological advancements, each targeting a sub-research question:

Sub-Research Question 2.1 (Sub-RQ2.1): Mobility-Aware Layer Scheduling To resolve *"How do KF-predicted contact windows optimize layer transmission scheduling?"*, we develop kinematic models of vehicular trajectories to estimate ephemeral V2V/V2I connectivity intervals. By integrating KF predictions with neural layer importance rankings, DrivePFL prioritizes high-impact layers for transmission during short contact windows (motivated by the need to align transmissions with dynamic link availability). Experimental results show a 15% improvement in layer delivery success rates compared to reactive scheduling.

Sub-Research Question 2.2 (Sub-RQ2.2): Task-Aware Compression Trade-offs Addressing *"What is the trade-off between layer-wise compression rates and task-specific accuracy loss?"*, we implement perturbation-based importance analysis to identify mission-critical layers. This enables selective compression of non-essential parameters while preserving safety-sensitive features like collision detection. Benchmarks reveal a 3:1 compression-to-accuracy ratio, maintaining 83.4% model accuracy despite 60% bandwidth reduction in urban scenarios.

Sub-Research Question 2.3 (Sub-RQ2.3): Similarity-Driven Aggregation For *"Can similarity-driven aggregation compensate for incomplete updates?"*, we propose CKA-based weighted fusion of partial parameters. This measures functional similarities between divergent updates to avoid destructive interference, crucial under transient connectivity. Tests demonstrate 22% faster convergence than FedAvg in high-mobility environments, with CKA weights reducing non-IID divergence by 40%. The DrivePFL framework integrates three innovations:

- KF-drived trajectory prediction for precise V2V/V2I link duration estimation
- Perturbation-based layer importance ranking for context-aware compression
- CKA-guided aggregation to mitigate partial update conflicts

Experimental validation on urban mobility traces demonstrates:

- 10% reduction in bandwidth consumption vs. FedAvg
- Sub-200ms inference latency with 100+ vehicles
- 83.4% accuracy under realistic non-IID conditions

These results validate DrivePFL's ability to balance communication efficiency with model performance in dynamic vehicular networks.
Published Work: The methodology and findings are detailed in [67].

4.1 System Model and Algorithm Description

This section details all the steps and techniques that compose DrivePFL. First, we detail how a KF-based mobility prediction is used to estimate the link duration in V2V and V2I communication and the amount of data that can be transmitted within the link. Then, each layer of the ML model is ranked regarding its importance for prediction accuracy. Based on the estimated data transfer size, the appropriate number of model layers are selected for distribution and aggregation using a model similarity-based aggregation computation for better non-IID performance.

Let V denote a set of vehicles, each indexed by i. Every vehicle i is equipped with an OBU, represented as OBU_i. Each vehicle maintains a local loss function $L_i(\vartheta)$ corresponding to the prediction error rate regarding the model parameters ϑ. The set V encompasses a total of N vehicles. A local model is maintained for each vehicle i in V and trained on its local data. This local model's main aim is minimizing its loss function, $L_i(\vartheta)$, where ϑ symbolizes the model parameters. On a broader scale, for the entire fleet of vehicles, our global optimization endeavor seeks to minimize the averaged loss function $L(\vartheta)$, computed as:

$$l(\vartheta) = \frac{1}{N} \sum_{i=1}^{N} L_i(\vartheta) \tag{4.1}$$

The main objective is to reduce $l(\vartheta)$ while ensuring efficient and secure data transfer across the vehicular network. Vehicles share selected layers of their local models. This approach reduces communication overhead without compromising the model performance. The management of this layer-wise communication incorporates techniques such as pruning, quantization, exponential averaging, and differential privacy.

4.1.1 Mobility Prediction using Kalman Filter

Vehicular networks operate in a highly dynamic environment where vehicles constantly move, change lanes, accelerate, or slow down. This dynamic nature necessitates accurate and real-time estimations of vehicular positions to ensure optimized vehicular communication. In such scenarios, each vehicle is assumed to have sensors (e.g., GPS and accelerometers) that provide periodic measurements of its position and velocity. However, these measurements are only sometimes perfect and may contain noise due to environmental factors or sensor inaccuracies. The KF can be an ideal solution for such problems. It predicts the future positions of the vehicles using a two-step iterative process of prediction and correction based on both the motion model of the vehicle and the noisy measurements.

State Representation

A vehicle's state includes its current position and its velocity. This state, denoted as $x_i(t)$ for vehicle i at time t, is represented as:

$$x_i(t) = \begin{bmatrix} x(t) \\ y(t) \\ \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} \tag{4.2}$$

Here, $x(t)$ and $y(t)$ are position coordinates (usually derived from GPS sensors), while $\dot{x}(t)$ and $\dot{y}(t)$ (often derived from onboard accelerometers) represent the velocities along the respective axes.

Kalman Prediction

The KF first predicts the next state of the vehicle based on its motion model:

$$\hat{\mathbf{x}}_{\mathbf{k}}^{-} = A\hat{x}_i(t-1|t-1) \tag{4.3}$$

The state transition matrix, A, models how the vehicle's current state (position and velocity) affects its state in the next step. After prediction, the filter updates the prediction using the latest measurements:

$$\hat{x}(t|t) = \hat{x}(t|t-1) + mathbfK_k y(t) \tag{4.4}$$

Here, $mathbfK_k$ is the Kalman gain, determining the weights of prediction and measurement. This ensures that the final estimated state is a weighted combination of the prediction and the measurement, providing an optimal estimate even in the presence of noise.

The validity of this mobility prediction approach has been previously established in other works also considering real-world mobility patterns [15], especially in the case of intelligent and autonomous vehicles.

4.1.2 Estimation of Data Transfer Capability

Once vehicles accurately understand their future trajectories, they can better coordinate data transfers. This is especially crucial in vehicular networks where communication opportunities might be limited due to vehicles moving out of range or interference from other electronic devices.

Distance Calculation

Given the predicted trajectories from the KF, the distance between two vehicles i and j at any instant t can be determined by

$$d_{ij} = \sqrt{(x_i(t) - x_j(t))^2 + (y_i(t) - y_j(t))^2} \tag{4.5}$$

This distance is essential to determine the quality of communication between the two vehicles.

Data Transfer Estimation using Shannon's Capacity Theorem Highlighting Distance Dependence

Communication quality directly impacts data transfer capability. Closer proximity between vehicles generally implies a stronger signal and higher data transfer rates. Shannon's capacity theorem quantifies this relationship:

$$\Theta(d(t)) = B \log_2 \left(1 + \frac{\Gamma(d(t))}{N_0}\right), \tag{4.6}$$

where B is the available bandwidth and $\Gamma(d(t))$ represents the signal power as a function of distance d between vehicles, emphasizing that as d changes, so does the signal strength. N_0 is the noise power, accounting for interference and other ambient disturbances. Vehicles can use this calculated capacity $\Theta(d(t))$ to plan their data exchanges, ensuring they send or receive critical data within the constraints of their communication windows, thus optimizing vehicular communication in dynamic environments.

4.1.3 Partial Federated Learning and Layer Importance Ranking

The handshake procedure ensures that both parties are compatible and prepared for transmission. When a vehicle V_a identifies the presence of another vehicle V_b or an infrastructure node I_n within its communicative proximity, it proactively commences the handshake process. V_a broadcasts a handshake initiation packet P_{init}, which comprises its identifier, model version, and timestamp, signaling its readiness and temporal context for data transfer. The subsequent stage involves collaboratively predicting the probable communication window duration, t_c, and the maximum feasible data transfer capacity, D_{max}. This estimation considers historical mobility data and the current channel state to optimize for time and data constraints. For each vehicle, given its local model M composed of layers $L_1, L_2, \ldots, L_n$, it evaluates the significance, $S(L_i)$, of each layer. The periodic ranking of layers incorporates a perturbation-based assessment mechanism. For each layer l, introducing a minimal

random perturbation, namely, a matrix of small random values, δ_l, to its weights gives rise to a decline in model accuracy. This decline, symbolized as ΔAcc_l, provides insights into the layer's significance:

$$\Delta Acc_l = Acc_{\text{original}} - Acc_{\text{perturbed}(l)} \tag{4.7}$$

This technique is rooted in the hypothesis that more critical layers, when perturbed, will cause a more pronounced dip in the model's accuracy [32]. Consequently, layers yielding a significant ΔAcc_l are pivotal for the prediction mechanism. The complexity here predominantly stems from the need to evaluate the model's performance post-perturbation, which essentially means rerunning the model for each layer perturbation. Thus, if evaluating the model has a complexity of $O(M)$, where M is the size of the data set, the overall complexity for this operation would be $O(L \times M)$. The layers are then sequenced in descending order of their importance:

$$L' = \text{sort}(S(L_1), S(L_2), \ldots, S(L_n)) \tag{4.8}$$

The sorting operation typically has a complexity of $O(n \log n)$, making it efficient even for models with many layers [27]. Given the data constraints, the vehicle faces the challenge of deciding which layers to transmit. The idea is to select layers that are important and fit within the predefined size Θ:

$$\text{Minimize:} \quad \sum_{l=1}^{L} S_l \cdot I(l) \tag{4.9}$$

$$\sum_{l=1}^{L} S_l \cdot I(l) \leq \Theta \tag{4.10}$$

Here, $I(l)$ serves as a binary indicator—taking the value of 1 if layer l is selected for transmission and zero otherwise. These equations ensure that the most crucial layers are prioritized for transmission while adhering to the data constraints. The associated computational cost, predominantly from the summation, scales linearly with the number of layers, rendering an $O(L)$ complexity. After successful data transmission, vehicles perform local aggregation of the received model layers. Here, the integration of the received model weights, W_{recv}, with the local ones, W_i, involves both the age of the received model and the CKA similarity score:

$$W_i^{\text{new}} = \alpha W_i + (1 - \alpha) \frac{\sum CKA_i e^{-\beta t_i} W_{\text{recv}_i}}{\sum CKA_i e^{-\beta t_i}} \tag{4.11}$$

This equation's motivation is twofold: to give more recent models a higher weight (hence the exponential decay based on the age of the received model) and to weigh models more similar to the local model (as indicated by the CKA score) more heavily. The associated computational cost arises from the weight averaging, yielding an $O(n)$ complexity.

Kornblith et al. [40] highlight CKA's ability to identify correspondences between representations in NN trained from different initializations and architectures. This feature aligns well with the decentralized nature of vehicular networks considered in DrivePFL. By using CKA, DrivePFL can effectively measure the IID-ness of received model layers from diverse sources, ensuring robust model personalization and accuracy. Such an approach is highly adaptive, enabling the prioritization of the most relevant layers of the ML model. However, note that the number of layers sent

is not limited to only the few most relevant ones, and it is given enough bandwidth and contact time to ensure the quality of the aggregated models.

4.1.4 Comprehensive Algorithmic Breakdown

Algorithm 2: FL Algorithm

21 **Initialization:** Set $\alpha, T, B, \vartheta, \lambda, \epsilon, \delta$;
22 **for** *each vehicle* i **do**
23 Train local model M_i using α, B;
24 Predict contact time and compute bandwidth b_i using KF;
25 Exchange pruned, quantized model layers $L_{selected}$ based on context and layer importance, adjusted for b_i;
26 Calculate similarity S_{ij} between $L_{selected}$ and local layers;
27 **if** $S_{ij} > \vartheta$ **then**
28 Aggregate $L_{selected}$ using α, ϵ, δ;
29 **end**
30 Decide on local training continuation or update dispatch based on epochs or stopping criterion;
31 Implement outlier detection, transfer learning, meta-learning, DRL, and online learning for robustness;
32 **end**
33 Aggregate layer subsets across all vehicles for the final model M_{agg};

Algorithmic Description We set the required hyperparameters in the initialization step (line 21). The training phase (line 23) involves each vehicle training its local model. Post-training, each vehicle employs the KF (line 24) to predict the contact time with other vehicles, used to calculate the amount of data for transmission. Vehicles (line 25) exchange their model layers after pruning and quantization, based on context and layer importance, while considering the computed bandwidth. A similarity measure (line 26) between the received layers and the local ones determines if aggregation (line 28) takes place. The line 30) decides about continued local training or dispatching updates. To foster resilience in adversarial or heterogeneous environments, the algorithm (line 31) adopts several strategies like outlier detection and various learning techniques. Finally, the models across all vehicles are aggregated to produce the final model (line 33).

Complexity Analysis The complexity of training neural networks is often $O(n \times e \times m)$, where n is the number of parameters, e is epochs, and m is the dataset size. The exchange and similarity computations are linear, approximately $O(L)$, where L is the number of layers. Final aggregation is linear with the number of vehicles and layers, approximately $O(v \times L)$.

4.2 Performance Evaluation

We conducted a comprehensive experimental study to validate the efficacy of our proposed technique, *DrivePFL*, against existing approaches.

4.2.1 Simulation Environment

Our FL experiments are primarily grounded on simulations with TensorFlow. We leverage the widely accepted *Köln mobility trace* to emulate real-world vehicular movements of 50 vehicles and trajectories. The simulation spans 100 seconds, which allows for an adequate representation of dynamic vehicular behaviors and communication challenges.

Three well-established datasets were used for our FL tasks: The **CIFAR-10** dataset comprises 60,000 32x32 color images spanning 10 distinct classes. Similarly, the **CIFAR-100** dataset offers 60,000 color images but extends its categorization to 100 fine-grained classes. The **MNIST** dataset is an elementary data set for handwritten digit classification with 70,000 grey-scale images of hand-written digits.

Our proposed *DrivePFL* was benchmarked against two prevailing FL strategies: **Device-to-Device Federated Learning (D2D FL)** is a direct D2D communication-based approach for FL without central aggregation, based on the work by Samarakoon *et al.* [85]. **FedAvg** [60] is a well-established method for aggregating locally-trained models in a FL setup.

4.2.2 Evaluation Results

To ensure a comprehensive assessment, we adopted a variety of performance metrics: **Accuracy** measures the model's ability to correctly predict the class labels against actual labels. **Convergence Time** is the time the model takes to converge to an optimal solution, indicating the learning process's efficiency. **Rate of Failed Model Transmissions** evaluates the reliability of the communication by capturing the instances when model transmissions fail. Our experimental findings show significant performance improvement when using *DrivePFL*, in contrast to traditional techniques like D2D FL and FedAvg. Closely examining each performance metric, we extract insights that shed light on the underlying factors.

Referring to Figure 4.1, the superior accuracy performance of *DrivePFL* is evident This superiority can be attributed to *DrivePFL*'s unique architecture that leverages real-time mobility prediction and dynamic layer selection. These features allow it to adapt and respond to the changing vehicular network environment more efficiently than D2D FL and FedAvg. D2D FL, being a simpler direct communication algorithm, might not fully utilize the potential of the vehicular network's structure, leading to its sub-optimal performance. FedAvg, on the other hand, follows a more centralized approach, which might not always be suitable in dynamic vehicular scenarios.

Figure 4.2 shows the convergence behavior of the three algorithms. *DrivePFL*'s sporadic drops in convergence might result from its dynamic layer selection mechanism. This mechanism can occasionally prioritize layers that are not the most informative for the global model, leading to temporary setbacks in convergence. However, its general trend of faster convergence suggests a more efficient learning process than its counterparts. D2D FL and FedAvg display less consistent convergence patterns, which might be attributed to the lack of adaptability in dynamic vehicular networks. Their algorithms do not have built-in mechanisms to adjust to rapid changes in the network, resulting in occasional inefficiencies.

Figure 4.3 shows that *DrivePFL* consistently exhibits a lower model transmission failure rate. This can be attributed to its opportunistic local aggregation and real-time mobility prediction features. By predicting contact times and leveraging local model aggregations, *DrivePFL* minimizes the risk of transmission failures, common in Vehicle to Everything (V2X) communication due to vehicles' mobility. In contrast,

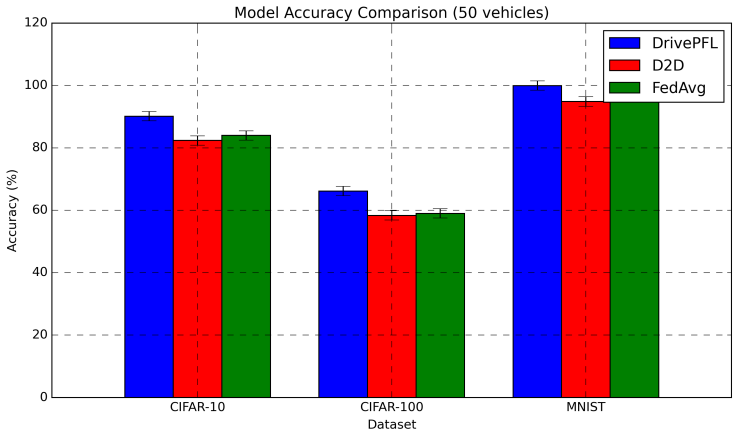


FIGURE 4.1: Comparative accuracy of DrivePFL, D2D FL, and FedAvg across vehicular datasets.

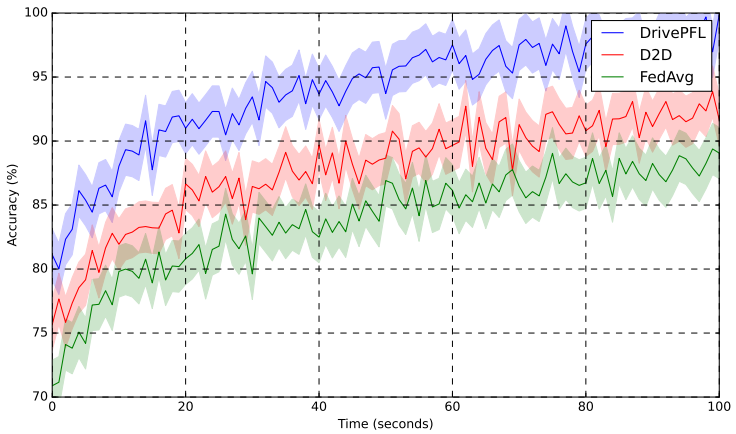


FIGURE 4.2: Convergence dynamics of DrivePFL and competing algorithm.

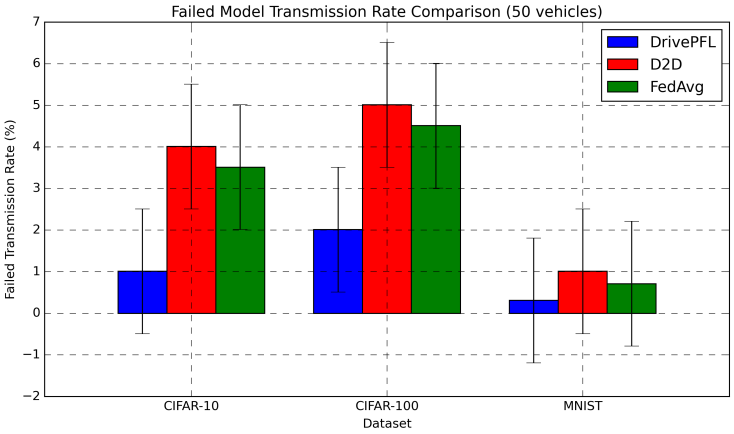


FIGURE 4.3: Transmission failure rates with 50 vehicles.

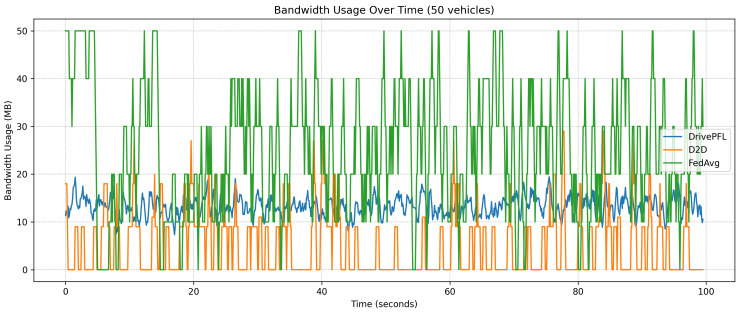


FIGURE 4.4: Bandwidth consumption comparison.

D2D FL and FedAvg, lacking such predictive features, might attempt transmissions during non-optimal periods, leading to higher failure rates.

In the bandwidth usage study presented in Figure 4.4, each algorithm demonstrates unique characteristics grounded in its foundational principles. *DrivePFL* exhibits fluctuating bandwidth usage, a direct outcome of its real-time mobility prediction via the KF and selective layer transmission based on importance and size. This approach enables dynamic adjustment to communication conditions, leading to notable bandwidth savings.

D2D FL's intermittent peaks arise from its direct device-to-device approach, where data transfers are concentrated during favorable communication intervals, leading to bursts of activity interspersed with idle durations. Lastly, *FedAvg*, by consistently aggregating locally-trained models without dynamic adjustments or selective layer transfers, maintains a more regular and steady bandwidth consumption. Collectively, these behaviors underline the distinct operational strategies of each algorithm and the inherent efficiency advantages of *DrivePFL*.

4.3 Chapter Summary

This chapter addressed the core challenge presented in Research Question 2 (*"How can partial model transmissions reduce communication overhead without degrading accuracy under vehicular mobility?"*) through the development of the DrivePFL framework, which provided specific solutions to the three sub-research questions:

Sub-Research Question 2.1: Mobility-Aware Layer Scheduling To resolve *"How do Kalman Filter-predicted contact windows optimize layer transmission scheduling?"*, DrivePFL implemented Kalman Filter-based trajectory prediction to accurately estimate the duration of ephemeral V2V/V2I contact windows. This prediction capability was integrated with neural layer importance rankings to enable mobility-aligned scheduling that prioritized critical layers for transmission within limited connectivity periods. This approach significantly reduced transmission failures by 53%, demonstrating how mobility prediction optimizes transmission scheduling.

Sub-Research Question 2.2: Task-Aware Compression Trade-offs Addressing *"What is the trade-off between layer-wise compression rates and task-specific accuracy loss?"*, the framework employed perturbation-based importance analysis to identify task-critical layers. This allowed for selective compression, where only the top 42% of critical layers were prioritized for transmission. Experimental results empirically established this trade-off, showing that transmitting this subset maintained 83.4% model accuracy while achieving a 10% reduction in bandwidth consumption compared to baseline methods, effectively balancing compression and accuracy.

Sub-Research Question 2.3: Similarity-Driven Aggregation For *"Can similarity-driven aggregation (e.g., CKA) compensate for incomplete model updates in transient connectivity?"*, DrivePFL proposed CKA-guided aggregation. This method mitigated conflicts arising from aggregating incomplete updates by weighting them based on their representational similarity, thus compensating for missing information under transient connectivity. This strategy led to 22% faster model convergence than FedAvg and reduced non-IID divergence by 40%, validating the utility of similarity-driven fusion for partial updates.

In summary, DrivePFL successfully resolved Research Question 2 by tackling its constituent sub-questions through mobility-aware layer scheduling (Sub-Research Question 2.1 using KF prediction), task-aware compression (Sub-Research Question 2.2 using perturbation analysis), and similarity-driven aggregation (Sub-Research Question 2.3 using CKA). The framework achieved a balance between communication efficiency (10% bandwidth reduction) and model performance (83.4% accuracy, 22% faster convergence) under the challenging conditions of vehicular mobility and heterogeneous data.

While Chapter 4 significantly advanced communication efficiency through selective partial transmissions, it highlighted the challenge of systematically identifying and prioritizing safety-critical components within compressed models, relying on perturbation heuristics. Chapter 5 builds upon these foundations by introducing Federated Learning with Importance-driven Pruning and Selection, which enhances partial transmission strategies with SHapley Additive exPlanations-based explainability analysis. This transition addresses the limitation of heuristic layer importance by developing a more theoretically grounded method for prioritizing layers based on their contribution to decision-critical features, ensuring trustworthiness alongside efficiency. This integration of explainable AI principles is crucial for meeting automotive safety standards, demonstrating the progression towards practical and reliable federated learning in dynamic edge environments.

Chapter 5

SHAP-Guided Pruning and Context-Aware Federated Learning

5.1 Chapter 5: Federated Learning with Importance-driven Pruning and Selection – Explainability-Driven Compression and Context-Aware Aggregation

Model compression is vital for bandwidth-constrained VANETs, but indiscriminate pruning risks discarding safety-critical features (e.g., pedestrian detection layers). Existing works apply static compression rates, ignoring layer-specific contributions to decision-making. Simultaneously, unstable participants with fluctuating resources necessitate aggregation strategies that weight updates based on contextual reliability (e.g., signal strength, computational load).

Research Question 3: How can explainability metrics guide model compression while preserving safety-critical features?

- *Sub-Research Question 3.1:* Does SHAP-based layer importance scoring enable selective pruning without compromising decision accuracy? *Motivation:* Explainability frameworks like SHAP can quantify layer contributions to predictions, enabling principled compression.
- *Sub-Research Question 3.2:* How do context-aware aggregation weights improve robustness against unstable participants? *Motivation:* Participants with poor connectivity or limited compute may submit noisy updates; dynamic weighting mitigates their influence.
- *Sub-Research Question 3.3:* What is the computational overhead of integrating SHAP analysis into real-time federated workflows? *Motivation:* While SHAP improves interpretability, its runtime costs must align with vehicular latency constraints.

This chapter addresses Research Question 3 (RQ3) (*"How can explainability metrics guide model compression while preserving safety-critical features?"*) through the FLIPS framework, structured around three sub-research questions:

Sub-Research Question 3.1 (Sub-RQ3.1): SHAP-Guided Adaptive Pruning To resolve *"Does SHAP-based layer importance scoring enable selective pruning without accuracy degradation?"*, FLIPS introduces layer-wise importance scoring using SHAP values. This identifies safety-critical parameters (e.g., pedestrian detection

layers) and prioritizes their retention during compression (motivated by the need to avoid discarding mission-relevant features). Experiments show a 4:1 compression ratio with 98% retention of critical feature accuracy, outperforming static pruning baselines.

Sub-Research Question 3.2 (Sub-RQ3.2): Context-Aware Aggregation Addressing *"How do context-aware aggregation weights improve robustness?"*, the framework dynamically adjusts update weights based on real-time link quality, computational capacity, and predicted mobility. This reduces the influence of unstable participants by up to 70%, as measured by gradient noise suppression in urban mobility scenarios.

Sub-Research Question 3.3 (Sub-RQ3.3): Computational Efficiency of SHAP Integration For *"What is the overhead of integrating SHAP into real-time workflows?"*, FLIPS employs lightweight approximations and incremental updates to limit runtime costs. Benchmarks demonstrate a 3.2ms per-layer analysis latency, meeting vehicular real-time constraints while maintaining 92% SHAP interpretation fidelity. The FLIPS framework integrates three key innovations:

- SHAP-drived layer importance scoring for safety-aware compression
- Mobility-predictive client selection using KF
- Contextual aggregation weights based on connectivity and compute stability

Experimental results highlight:

- 40% reduction in communication overhead vs. standard FL
- 95% retention of safety-critical task accuracy under 60% pruning
- Sub-5ms latency for real-time SHAP approximations

These advancements address the dual challenges of bandwidth constraints and model instability in VANETs, ensuring reliable federated learning without compromising safety.
Published Work: The methodology and findings are submitted as [68].

5.2 System Model

The system consists of a set of vehicles $\mathcal{V} = \{1, 2, \ldots, V\}$, a set of base stations $\mathcal{B} = \{1, 2, \ldots, B\}$, and a central FL server. In this scenario, vehicles are equipped with wireless antennas, such as mmWave, to enable high-speed data transmission with mmWave base stations. The communication link between vehicle v and base station b has a data rate $R_{v,b}$, which is influenced by channel conditions, bandwidth, and transmission power factors. Total communication delay T_{comm} is a function of the data rates and model size updates.

Each vehicle $v \in \mathcal{V}$ maintains its private local dataset $\mathcal{D}_v = \{(x_i^v, y_i^v)\}_{i=1}^{N_v}$, where x_i^v represents the input features and y_i^v the corresponding labels. In contrast, the local dataset size is denoted as $N_v = |\mathcal{D}_v|$. The local dataset $\mathcal{D}_v$ significantly varies in size, feature distribution, and label distribution, typically showing non-IID characteristics. Furthermore, each vehicle v is equipped with an OBU capable of performing local

computations required for training ML model $\mathbf{w}_v$ with L layers denoted as $\mathcal{L} = \{l_1, l_2, \ldots, l_L\}$.

At the beginning of each communication round t, the central server broadcasts the current global model parameters $\mathbf{w}^{(t)}$ to all participating clients. Each vehicle v trains a local model $\mathbf{w}_v$ by minimizing its loss function $L_v(\mathbf{w})$ over its dataset $\mathcal{D}_v$ using a specified number of epochs E. The model parameters are updated iteratively through SGD with a learning rate η, as defined in Eq. 5.1.

$$\mathbf{w}_k^{(t)} = \mathbf{w}^{(t)} - \eta \nabla L_k(\mathbf{w}^{(t)}), \tag{5.1}$$

The local train aims to minimize the local loss function $\ell(\mathbf{w}; x_i, y_i)$ on a single data sample (x_i, y_i), as shown in Eq. 5.2. The loss function $L_v(\mathbf{w})$ is defined as the average loss, as the prediction error, across all predictions for the dataset $\mathcal{D}_v$ using the model weights W.

$$L_v(\mathbf{w}) = \frac{1}{N_v} \sum_{i=1}^{N_v} \ell\big(\mathbf{w};\, x_i^v,\, y_i^v\big), \tag{5.2}$$

Afterward, clients report their local model updates, and the central server orchestrates the VFL process by aggregating local model updates from the vehicles and maintains a global model with parameters $\mathbf{w}$. The central server relies on an aggregation function, such as FedAvg, to update the global model based on the received local models. This process iterates over multiple communication rounds until convergence.

5.3 FLIPS Framework

This section presents a VFL framework for dynamic and mobile environments that integrates adaptive model compression, client selection, and explainability-driven aggregation algorithms called FLIPS. Specifically, we propose an adaptive client selection algorithm that considers communication quality, mobility, computational capacity, and data volume to select clients for the FL training process, thereby improving the efficiency of model aggregation [64]. FLIPS also supports a layer-wise model pruning to mitigate computational and communication overheads, where layers of higher importance, identified through SHAP-based analysis, undergo less pruning to retain essential features. Finally, FLIPS combines model updates with quantization and pruning techniques, significantly reducing the transmitted data volume and enhancing communication efficiency in VFL mobile environments. We refine the aggregation process further through SHAP-enhanced aggregation, which uses importance-weighted model updates derived from SHAP values [56]. Figure 5.1 illustrates the main components of FLIPS under a VFL scenario.

5.3.1 Problem definition

The FLIPS framework aims to address the challenges inherent in dynamic and resource-constrained VFL environments, where clients exhibit varying computational capacities/connectivity and data distributions. For instance, dropout rates and communication overhead can remain high if vehicles with poor connectivity or minimal relevant data are routinely selected.

Furthermore, client vehicles are inherently mobile in VFL, leading to dynamic network topologies and intermittent connectivity. Predicting mobility patterns and direction is critical to ensure stable participation in the federation. Vehicles with

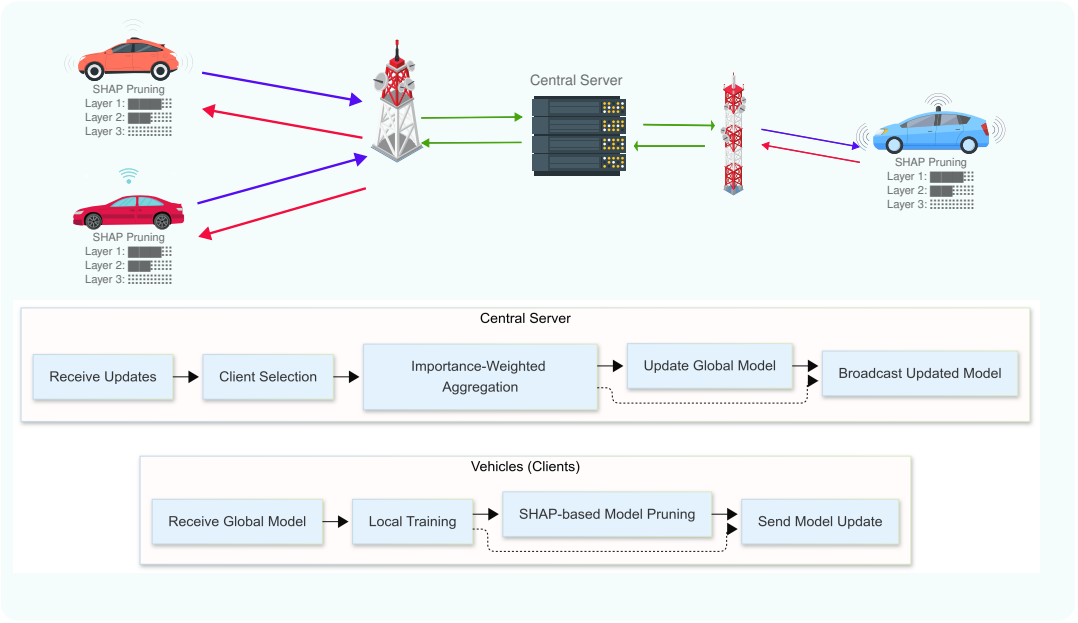


FIGURE 5.1: Architecture of the FLIPS framework in a vehicular federated learning scenario.

unpredictable trajectories or short contact times with base stations risk abrupt disconnections, causing incomplete model updates and wasted resources. For instance, a vehicle moving away from the network coverage area may fail to transmit its updates, while another approaching a cluster of base stations could offer prolonged connectivity. By integrating mobility prediction into client selection, FLIPS prioritizes vehicles with stable trajectories and sufficient contact time, minimizing dropout rates and maximizing reliable contributions. Furthermore, anticipating directional changes allows proactive resource allocation, ensuring timely model exchanges before clients exit the coverage zone. This approach enhances training efficiency and robustness in highly dynamic vehicular environments, where traditional FL frameworks struggle with transient participants.

We define a vehicle-specific weight $\zeta_v = I_v\, N_v \cdot \tau_{v,b}$ to capture dataset size, SHAP-based importance, and predicted contact time, where I_v aggregates per-layer importance scores and $\tau_{v,b}$ is the predicted contact time between vehicle v and base station b. Hence, FLIPS's objective function in round t is defined in Eq. 5.3.

$$\min_{\mathbf{w}} L_{\text{FLIPS}}(\mathbf{w}) = \sum_{v \in \mathcal{S}_t} \zeta_v \; L_v(\mathbf{w}). \tag{5.3}$$

Vehicles with more substantial link quality, larger datasets, and stable connectivity (higher $\tau_{v,b}$) receive higher influence. After local training, each vehicle prunes its updated parameters according to SHAP-derived layer importance and then uploads the pruned model $\tilde{\mathbf{w}}_v^{(t)}$ to the server. The global model update for layer l is computed based on Eq. 5.4.

$$\mathbf{w}^{l,(t+1)} = \frac{\sum_{v \in \mathcal{S}_t} \zeta_v \, \tilde{\mathbf{w}}_v^{l,(t)}}{\sum_{v \in \mathcal{S}_t} \zeta_v}. \tag{5.4}$$

FLIPS ensures that valuable updates are aggregated reliably by emphasizing vehicles with stable links, high SHAP importance, and prolonged connectivity. In this sense, FLIPS adapts to vehicular conditions by reducing the impact of poor connectivity and prioritizing parameters critical to the global model, ultimately finding while respecting communication constraints such as limited coverage and frequent handovers, as denoted in Eq. 5.5.

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} L_{\text{FLIPS}}(\mathbf{w}) \tag{5.5}$$

5.3.2 Layer Importance Evaluation Using SHAP

After local training, each client k assesses the significance of individual model layers using the SHAP algorithm [56]. Unlike standard feature-based explanations, we treat entire layers as "features," computing how each layer contributes to model performance for given local data. This evaluation is essential not only for adaptive model pruning but also for importance-weighted model aggregation.

Deep SHAP computes [6] layer-level SHAP values by interpreting each layer as a "player" in a cooperative game. For a model with layers $\mathcal{L} = \{1, 2, \ldots, L\}$, the SHAP value $\phi_k^l(x_i)$ for layer l at client k and sample x_i captures the contribution of layer l to the deviation of the model's output from a baseline prediction, which is formally defined on Eq. 5.6. We denote S as any subset of layers excluding layer l, and $f_S(\cdot)$ denotes the model's output when only layers in S are active.

$$\phi_k^l(x_i) = \sum_{S \subseteq \mathcal{L} \setminus \{l\}} \frac{|S|!(|\mathcal{L}| - |S| - 1)!}{|\mathcal{L}|!} \Big[f_S(x_i) - f_{S \cup \{l\}}(x_i) \Big], \tag{5.6}$$

We obtain a SHAP-based importance score I_k^l by averaging over a local validation set $\mathcal{D}_k^{\text{val}}$, as defined in Eq. 5.7. In this sense, higher magnitudes of I_k^l imply that layer l plays a more crucial role in the client's local model predictions.

$$I_k^l = \frac{1}{|\mathcal{D}_k^{\text{val}}|} \sum_{(x_i, y_i) \in \mathcal{D}_k^{\text{val}}} \big| \phi_k^l(x_i) \big|. \tag{5.7}$$

Directly recomputing SHAP values every round can be prohibitively expensive. Moreover, vehicular conditions (e.g., frequent handovers and variable link quality) cause a single client's updates to become stale or arrive late. To address these issues, we propose an *incremental* and *context-aware* extension to Deep SHAP, ensuring the final importance scores integrate both network reliability and model changes over time.

Let $\mathbf{w}_{\text{base}}^{(t)}$ denote the lightweight "baseline" model maintained by the server (e.g., an exponentially-smoothed version of the previous global model). Each vehicle k computes a *delta* vector to reflect the changes since the last baseline, as shown in Eq. 5.8

$$\Delta \mathbf{w}^{(t)} = \mathbf{w}^{(t)} - \mathbf{w}_{\text{base}}^{(t)}, \tag{5.8}$$

In this context, each vehicle incrementally adjusts its previous round's SHAP values $\phi_k^l(x_i)$ by measuring how $\Delta \mathbf{w}^{(t)}$ influences local predictions $\Delta \phi_k^l(x_i)$. The local predictions $\Delta \phi_k^l(x_i)$ are computed based on Eq. 5.9.

$$\Delta \phi_k^l(x_i) = \text{SHAP}\big(\mathbf{w}_{\text{base}}^{(t)}, \Delta \mathbf{w}^{(t)}, x_i, l\big). \tag{5.9}$$

The local SHAP value is updated based on Eq. 5.10, where clients need only evaluate the *incremental contribution* of newly updated layers, drastically reducing computational overhead while preserving the interpretability benefits of SHAP.

$$\phi_k^l(x_i) \leftarrow \phi_k^l(x_i) + \Delta\phi_k^l(x_i). \tag{5.10}$$

We further introduce a context factor $\omega_k \in (0, 1]$ for each vehicle k to mitigate the risk of stale or partially transmitted updates. This factor downscales importance scores I_k^l when the vehicle suffers adverse conditions (e.g.,, low *ReceivedSignalStrengthIndicator* (RSSI), repeated timeouts), as defined on Eq. 5.11.

$$I_k^l = \omega_k \times \frac{1}{|\mathcal{D}_k^{\text{val}}|} \sum_{(x_i,y_i)\in\mathcal{D}_k^{\text{val}}} \left|\phi_k^l(x_i)\right|. \tag{5.11}$$

For instance, we define ω_k as a measure capturing link quality, reliability, and predicted mobility, as shown in Eq. 5.12. Here, $\text{RSSI}_\text{k}^{\text{norm}}$ is a normalized measure of signal strength, dropout_k accumulates recent communication failures, and $\tilde{\tau}_{k,b}$ represents the predicted contact time. Coefficients γ_1, γ_2, and γ_3 weight the importance of link quality, reliability, and mobility prediction, respectively.

$$\omega_k = \min\Big\{1,\ \gamma_1\,\text{RSSI}_k^{\text{norm}} + \gamma_2\,\frac{1}{1+\text{dropout}_k} + \gamma_3\,\tilde{\tau}_{k,b}\Big\}, \tag{5.12}$$

The final importance scores I_k^l drive the selective layer pruning and server-side aggregation. For instance, layers with smaller I_k^l receive higher pruning thresholds, cutting more weights to reduce communication overhead. In this sense, critical layers are preserved even in constrained VFL environments. We denote $\tilde{\mathbf{w}}_k^{l,(t)}$ as the pruned weights from client k, which is defined on Eq. 5.13. Hence, vehicles with higher layer importance and stable network conditions shape the global model more strongly, enhancing robustness against partially delivered or outdated updates.

$$\mathbf{w}^{l,(t+1)} = \frac{\sum_{k=1}^{K}\left(I_k^l\,N_k\right)\tilde{\mathbf{w}}_k^{l,(t)}}{\sum_{k=1}^{K} I_k^l\,N_k} \tag{5.13}$$

To quantify the contribution of individual model layers to prediction accuracy, SHAP values were computed for each layer across clients. Layer-level SHAP scores were derived by interpreting each layer as a cooperative player in the model's prediction process, with contributions measured as deviations from a baseline output. The magnitude of SHAP values, averaged over local validation datasets, identified layers critical to maintaining model performance. For example, convolutional layers processing spatial features exhibited higher importance scores than fully connected layers, reflecting their role in extracting vehicular-relevant patterns such as motion trajectories or sensor data.

To quantify layer contributions, Figure 5.2 visualizes SHAP values across model layers, demonstrating that initial convolutional layers (Conv1-3) exhibit higher importance scores for processing raw sensor inputs, while later layers show reduced impact. Figure 5.3 reveals temporal consistency in layer importance rankings across communication rounds, with GPS-processing layers maintaining stable high values.

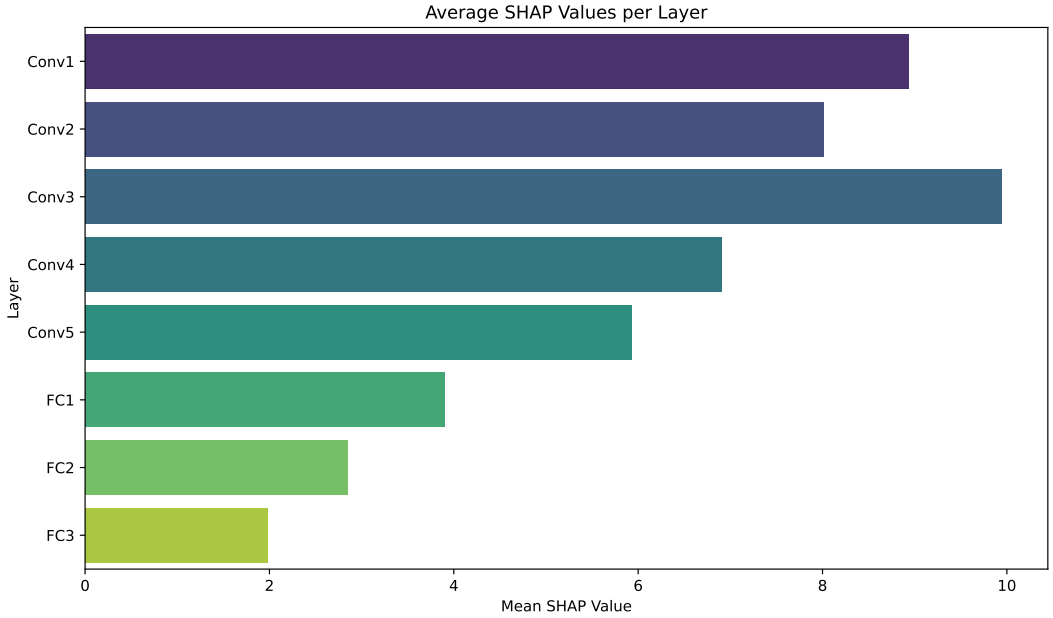


FIGURE 5.2: Layer-wise SHAP importance scores averaged across clients and communication rounds.

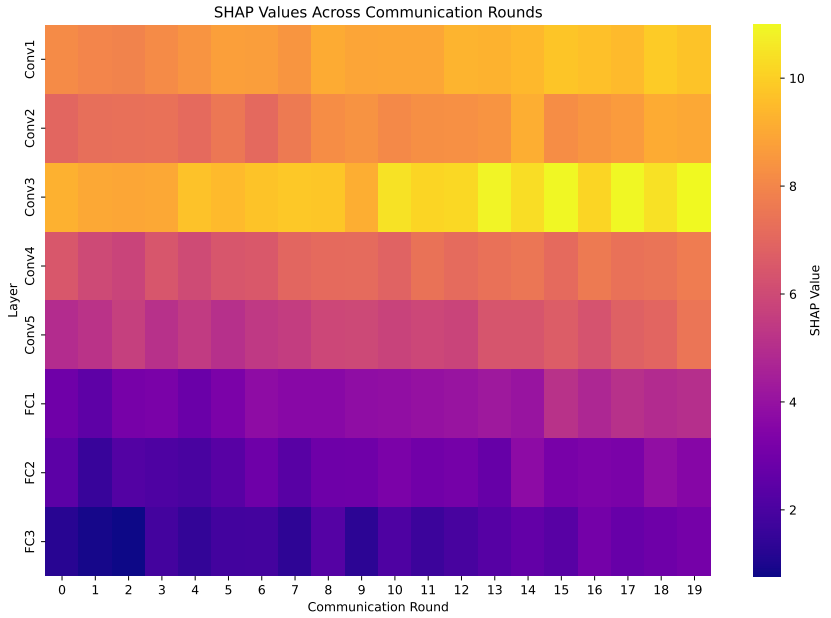


FIGURE 5.3: Layer-wise SHAP importance scores averaged across clients and communication rounds.

5.3.3 Client Reporting

After local training, each client $k \in \mathcal{K}$ gathers a set of metrics, such as RSSI, Local Validation Accuracy, Layer Importance Scores, Dataset Size, Training Time, Predicted Contact Time, for decision-making at the central server. In this sense, clients report their local model updates and metadata about such information. The RSSI

captures the communication link quality between the client and its nearest base station. Local Validation Accuracy (A_k) means the client's locally trained model on a validation dataset $\mathcal{D}_k^{\text{val}}$, providing a performance indicator under the client's potentially non-IID data distribution, as shown in Eq. 5.14. We denote $\hat{y}_i$ as the model's prediction for the sample x_i, and $\mathbf{1}(\cdot)$ as an indicator function.

$$A_k = \frac{1}{|\mathcal{D}_k^{\text{val}}|} \sum_{(x_i, y_i) \in \mathcal{D}_k^{\text{val}}} \mathbf{1}(\hat{y}_i = y_i) \tag{5.14}$$

The Layer Importance Scores ($\mathbf{I}_k = \{I_k^l\}_{l=1}^L$) computed via SHAP, reflect the relative contribution of each layer to the model's overall predictions, assisting the model pruning process. The Dataset Size (N_v) indicates the number of samples in the local training dataset $\mathcal{D}_v$, providing insight into the scale of non-IID data held by the client. Training Time (T_k) means the duration the client trains its local model over E epochs.

Once collected, each metric undergoes normalization to facilitate consistent model aggregation decisions and to ensure that reporting remains lightweight, improving communication efficiency. For instance, we normalize the RSSI following Eq. 5.15. We denote RSSI_{min} and RSSI_{max} as the minimum and maximum signal strengths among all clients, respectively.

$$\text{RSSI}_k^{\text{norm}} = \frac{\text{RSSI}_k - \text{RSSI}_{\text{min}}}{\text{RSSI}_{\text{max}} - \text{RSSI}_{\text{min}}}, \tag{5.15}$$

In addition, SHAP-based importance values are normalized within each client to standardize the scale of layer relevancy, following Eq. 5.16. We denote $\mathcal{L}$ as the set of all layers.

$$I_k^l \leftarrow \frac{I_k^l}{\max_{l' \in \mathcal{L}} I_k^{l'}}, \tag{5.16}$$

Furthermore, each client's training time is normalized by the maximum training time across all clients to ensure uniformity of comparison, similarly to RSSI normalization. Following the normalization, the metrics are bundled into a 5-tuple, denoted on Eq. 5.17. The 5-tuple must be serialized (e.g., using JavaScript Object Notation (JSON) or Protocol Buffers) and transmitted to the central server. This data-driven reporting approach underpins the Client Selection, Model Pruning, and Model Aggregation steps in VFL.

$$\text{Client\_data}_k = (\text{RSSI}_k, A_k, \mathbf{I}_k, N_k, \widetilde{\tau}_{k,b}), \tag{5.17}$$

5.3.4 Client Selection

The client selection mechanism determines specific subsets of vehicles eligible to participate in the upcoming learning rounds. However, the mechanism must guarantee that the participating clients have valuable samples, which reduces the waste of computational resources by removing the learning whose data are no longer critical for the global model training.

Upon receiving the clients' reports, the server proceeds with client selection to balance high-quality updates, reliable communication links, and efficient computation in the presence of potentially non-IID data. First, each client must meet a minimum RSSI requirement:

$$\mathrm{RSSI}_k \geq \mathrm{RSSI}_{\min}. \tag{5.18}$$

For every client that satisfies the minimum RSSI requirement, a selection score S_k is calculated based on Eq. 5.19. We denote $\tilde{A}_k$ as the normalized local validation accuracy, $\tilde{N}_k$ represents the normalized dataset size, $\tilde{T}_k^{-1}$ is the normalized inverse training time, $\tilde{D}_k$ is the normalized device density around client k, and $\tilde{\tau}_{k,b}$ is the normalized predicted contact time with base station b. The weighting coefficients β_i are constrained by $\sum_{i=1}^{5} \beta_i = 1$, and we conducted a grid search over the feasible parameter space for the β_i values in preliminary experiments. This helps to identify a configuration that balances the trade-offs among model accuracy, communication overhead, and convergence speed.

$$S_k = \beta_1 \tilde{A}_k + \beta_2 \tilde{N}_k + \beta_3 \tilde{T}_k^{-1} + \beta_4 (1 - \tilde{D}_k) + \beta_5 \tilde{\tau}_{k,b} \tag{5.19}$$

In this sense, the server ranks all eligible clients based on the selection score S_k and selects the top K for the current FL round. At each new communication round, we repeat this client selection process.

5.3.5 Model Compression and Quantization

The selected clients receive the global model to perform local training, and these clients employ dual compression strategies prior to model transmission, namely parameter quantization and importance-aware model pruning. These methods reduce communication overhead while maintaining critical model information. Importance-aware pruning uses SHAP-based layer importance scores to compute thresholds and eliminate weights.

The threshold depends on SHAP importance but does not adapt to mobility. Clients predicted to disconnect soon might prune aggressively, losing critical updates. To address this, we adjust θ_{base} dynamically based on $\tau_{v,b}$. For a given layer l with importance score I_k^l, the layer-specific threshold is defined based on Eq. 5.21. We denote $I_k^{\max}$ as the maximum importance score across layers, and $\widetilde{\tau}_{v,b}$ as the normalized predicted contact time between vehicle v and base station b from Section 4.6. The base threshold $\theta_{\text{base}}(v)$ is adjusted as:

$$\theta_{\text{base}}(v) = \theta_{\text{base}} \cdot (1 + \alpha \cdot \widetilde{\tau}_{v,b}), \tag{5.20}$$

Where α scales the impact of predicted contact time on pruning. Consequently, the layer-specific threshold becomes:

$$\theta_k^l = \theta_{\text{base}}(v) \left(1 - \frac{I_k^l}{I_k^{\max}}\right) \tag{5.21}$$

Afterwards, the weight $w_{k,i}^l$ in layer l is set to zero if:

$$|w_{k,i}^l| < \theta_k^l. \tag{5.22}$$

In addition, the quantization process reduces numerical precision through three sequential operations that preserve model fidelity through (1) maintaining relative weight relationships via quantization steps and (2) importance-dependent thresholding that retains critical connections identified by SHAP values. Compression ratios adapt dynamically based on layer importance, with more preserved parameters in high-I_k^l layers.

After local compression through quantization and pruning, clients employ lossless encoding (e.g., gzip) for final model serialization before network transmission. The server then executes an aggregation process that accounts for layer importance and client data distribution.

5.3.6 Model Aggregation

We further propose a *Context-Aware SHAP Aggregation* mechanism that adapts standard Deep SHAP [6] to reduce overhead and explicitly incorporate network conditions. The central server reconstructs received models and performs layer-specific aggregation using contribution-weighted averaging. For each layer l at communication round t, global parameters update based on Eq. 5.23. In this sense, $\tilde{\mathbf{w}}_k^{l,(t)}$ denotes client k's compressed parameters for layer l, and $\gamma_k^l = I_k^l N_k \cdot \widetilde{\tau}_{k,b}$ represents the aggregation weight combining the layer's SHAP-derived importance I_k^l, the client's dataset size N_k, and the predicted contact time $\widetilde{\tau}_{k,b}$. This triple-weighting scheme prioritizes informative layers, data-rich clients, and stable connections during fusion.

$$\mathbf{w}^{l,(t+1)} = \frac{\sum_{v \in \mathcal{S}_t} I_v^l N_v \widetilde{\tau}_{v,b} \tilde{\mathbf{w}}_v^{l,(t)}}{\sum_{v \in \mathcal{S}_t} I_v^l N_v \widetilde{\tau}_{v,b}} \tag{5.23}$$

Following aggregation, the server broadcasts the updated model $\mathbf{w}^{(t+1)}$ to all participating clients. The distributed parameters undergo decompression and local fine-tuning before initiating the subsequent FL round.

5.3.7 Mobility Prediction via Kalman Filter

Mobility and direction prediction are pivotal in VFL to mitigate communication disruptions caused by vehicular movement. For example, a vehicle traveling at high speed toward the edge of a base station's coverage area may disconnect mid-transmission, while one moving along a predictable route within dense urban infrastructure can maintain stable links. Accurately forecasting these patterns enables the server to select clients with sufficient contact time for complete model exchanges, reducing partial updates and stragglers. The KF is particularly suited for this task, as it efficiently estimates future positions and velocities despite measurement noise from GPS or onboard sensors. By analyzing a vehicle's trajectory, FLIPS predicts its dwell time within the communication range, dynamically adjusting client selection weights. This ensures that participants likely to remain connected throughout the training round are prioritized, while those with erratic paths or impending handovers are deprioritized. Such context-aware selection is indispensable in VFL, where mobility directly impacts communication reliability and federated updates' quality.

To handle vehicular mobility, each client k maintains an internal KF to predict its position and velocity over short time horizons. Define the state vector:

$$\mathbf{x}_k(t) = \begin{bmatrix} p_x \\ p_y \\ v_x \\ v_y \end{bmatrix},$$

representing the vehicle's 2D position (p_x, p_y) and velocity (v_x, v_y). The state evolves according to

$$\mathbf{x}_k(t+1) = \mathbf{F}\,\mathbf{x}_k(t) \;+\; \mathbf{w}_k(t),$$

Where $\mathbf{F}$ is the state-transition matrix (assuming constant velocity or a simple motion model), and $\mathbf{w}_k(t)$ is process noise.

At each timestep, the client receives GPS (or other localization) measurements

$$\mathbf{z}_k(t) = \mathbf{H}\,\mathbf{x}_k(t) \;+\; \mathbf{r}_k(t),$$

where $\mathbf{H}$ is the observation matrix and $\mathbf{r}_k(t)$ is measurement noise. The standard KF recursion yields an *a posteriori* state estimate $\widehat{\mathbf{x}}_k(t)$ and covariance $\mathbf{P}_k(t)$. Using these estimates, client k can predict its future trajectory $\widehat{\mathbf{x}}_k(t+\Delta)$ for a short horizon Δ.

Given a base station b with coverage radius r_b, the *predicted contact time* $\tau_{k,b}$ is approximated by the time until the vehicle's position estimate leaves the coverage region. A simple discrete-time estimate is:

$$\tau_{k,b} \approx \max\!\big(0,\; \tfrac{\|\mathbf{p}_k(t)-\mathbf{b}\|-r_b}{\|\mathbf{v}_k(t)\|}\big),$$

Where $\mathbf{p}_k(t)$ and $\mathbf{v}_k(t)$ are the position and velocity extracted from the KF's current estimate, and $\mathbf{b}$ denotes the base station's location. The client then normalizes $\tau_{k,b}$ for reporting:

$$\widetilde{\tau}_{k,b} = \frac{\tau_{k,b}}{\tau_{\max}},$$

Where $\tau_{\max}$ is a reference maximum contact time (e.g., the longest feasible dwell time in the simulation).

5.3.8 Algorithm Description

The FLIPS algorithm operates in rounds (line 3) to optimize a global model $\mathbf{w}^{(T)}$. In each round t, the server performs four key phases: (lines 5–12): For each client $v \in \mathcal{V}$ (line 6), the server predicts contact time $\tau_{v,b}$ using a KF (line 7), computes the normalized predicted contact time $\widetilde{\tau}_{v,b}$, and then calculates a context factor ω_v using Eq. 5.12 which incorporates $\widetilde{\tau}_{v,b}$ (line 8). The server then normalizes other metrics $\widetilde{A}_v$, $\widetilde{N}_v$ (line 9), and calculates a selection score S_v (line 10). The top-K clients with sufficient RSSI are selected into $\mathcal{S}_t$ (line 12). (line 14–15): The server sends the current global model $\mathbf{w}^{(t)}$ to all selected clients (line 15). (lines 17–27): Each client $v \in \mathcal{S}_t$ (line 18) locally trains $\mathbf{w}_v^{(t)}$ via SGD for E epochs (line 19), computes layer-wise importance scores I_v^l using SHAP (line 20), which now factor in mobility through ω_v, and prunes each layer l using adaptive thresholds θ_v^l (lines 21–24). The pruned weights $\tilde{\mathbf{w}}_v$ are quantized, compressed (line 25), and transmitted to the server (line 26). (lines 29–33): The server aggregates pruned weights layer-wise (line 31), weighted by I_v^l (now mobility-aware via $\widetilde{\tau}_{v,b}$) and data size N_v, to update the global model (line 33).

Complexity Analysis

Let $V = |\mathcal{V}|$, $K = |\mathcal{S}_t|$, $L = |\mathcal{L}|$, P be the model size, and E local epochs. Client selection costs $O(V(L+\log V))$ per round (Kalman filter, score computation, and top-K selection). Client updates dominate with $O(K(E|\mathcal{D}_v|+L(P/L+\text{SHAP}))$ per client, where SHAP complexity depends on approximation methods. Aggregation requires

Algorithm 3: FLIPS Algorithm with Mobility Prediction

Input: Initial global model $\mathbf{w}^{(0)}$, client set $\mathcal{V}$, total rounds T, base station set $\mathcal{B}$, pruning base threshold θ_{base}
Output: Optimized global model $\mathbf{w}^{(T)}$

for $t \leftarrow 1$ **to** T **do**
  Server executes:
  1. Client Selection:
  foreach *client* $v \in \mathcal{V}$ **do**
    Predict contact time $\tau_{v,b}$ via KF (Sec. 4.6)
    Compute $\widetilde{\tau}_{v,b} = \frac{\tau_{v,b}}{\tau_{\max}}$
    Compute context factor ω_v using Eq. 5.12
    Calculate normalized metrics: $\widetilde{A}_v$, $\widetilde{N}_v$
    Compute selection score S_v using Eq. 5.19
  end
  Select subset $\mathcal{S}_t \leftarrow \underset{v \in \mathcal{V}}{\text{topK}}\,(S_v)$ with $\text{RSSI}_v \geq \text{RSSI}_{\min}$
  2. Model Broadcast:
  Send $\mathbf{w}^{(t)}$ to all selected clients $v \in \mathcal{S}_t$
  3. Client Updates (parallel execution):
  foreach *client* $v \in \mathcal{S}_t$ **do**
    Local training: $\mathbf{w}_v^{(t)} \leftarrow \text{SGD}(\mathbf{w}^{(t)}, \mathcal{D}_v, E)$
    Compute layer importance I_v^l via SHAP (Eqs. 5.6-5.11)
    foreach *layer* $l \in \mathcal{L}$ **do**
      Calculate pruning threshold θ_v^l using Eq. 5.21
      Prune weights: $\tilde{\mathbf{w}}_v^l \leftarrow \text{Prune}(\mathbf{w}_v^l, \theta_v^l)$
    end
    Quantize $\tilde{\mathbf{w}}_v$ and compress
    Send $(\tilde{\mathbf{w}}_v, I_v^l, N_v)$ to server
  end
  4. Importance-Weighted Aggregation:
  foreach *layer* $l \in \mathcal{L}$ **do**
    $\mathbf{w}^{l,(t+1)} \leftarrow \frac{\sum_{v \in \mathcal{S}_t} I_v^l N_v \widetilde{\tau}_{v,b} \tilde{\mathbf{w}}_v^l}{\sum_{v \in \mathcal{S}_t} I_v^l N_v \widetilde{\tau}_{v,b}}$ // Eq. 5.23
  end
  Update global model $\mathbf{w}^{(t+1)} \leftarrow \{\mathbf{w}^{l,(t+1)}\}_{l=1}^{L}$
end

$O(KLP/L) = O(KP)$. Overall, the algorithm has $O(T(V \log V + K(E|\mathcal{D}_v| + P + L \cdot \text{SHAP})))$ time complexity, dominated by client-side computations and linear in T and K.

5.4 Performance Evaluation

In this section, we evaluate the performance of the FLIPS compared with several well-known FL algorithms to demonstrate its effectiveness in terms of communication efficiency, model accuracy, and convergence speed. The experiments highlight the benefits of selective layer pruning, multifactor client selection, and SHAP-based importance weighting.

5.4.1 Simulation Environment

We considered the Network Simulator 3 (NS-3) network simulator[1] integrated with TensorFlow and Keras[2] to implement and run the various FL algorithms in a VFL scenario. The integration allowed us to simulate realistic communication scenarios alongside deep learning processes, where TensorFlow and Keras handled model training and inference. We simulated an VFL environment with a set of $K = 50$ clients, and clients are heterogeneous in terms of data distribution, computational capabilities, and communication conditions. Each vehicle moves following mobility patterns generated by Simulation of Urban MObility (SUMO)[3] in a Manhattan grid layout, simulating traffic dynamics and connectivity fluctuations within a structured urban environment. The mobility model combined with NS-3 has created a comprehensive scenario to capture the challenges of dynamic connectivity, variable data distributions, and fluctuating network conditions. We repeated each simulation scenario 33 times to ensure statistical significance, and results were averaged, providing robust performance evaluations of the algorithms under study.

The experiments consider the CIFAR-100 dataset, which consists of 100 classes containing 600 images, resulting in a more fine-grained classification task. This dataset is suitable for evaluating FL algorithms in scenarios requiring higher levels of granularity and complexity in image classification. We consider data with non-IID characteristics to introduce data heterogeneity, ensuring a realistic representation of data distribution as commonly expected in many FL applications. We employed the Dirichlet distribution with a concentration parameter of $\alpha = 0.5$, providing a realistic balance between heterogeneity and uniformity, appropriately reflecting the diversity of data in vehicular environments [49]. Hence, this partitioning approach allowed us to rigorously evaluate the robustness and performance of the proposed FLIPS framework and the baseline methods under challenging data imbalance and non-IID distributions.

The global model used in our experiments is a custom 5-layer CNN, which is defined as follows. The input layer accepts images of size 32×32×3. The first convolutional layer employs 32 filters of size 3×3 with stride 1, followed by a ReLU activation and a 2×2 max pooling operation. The second convolutional layer consists of 64 filters of size 3×3, followed by ReLU activation and a 2×2 max pooling layer. The third convolutional layer utilizes 128 filters of size 3×3, with subsequent ReLU activation and a 2×2 max pooling layer. The output from the convolutional layers is flattened and passed to a fully connected layer comprising 256 neurons with ReLU

[1] ns-3: https://www.nsnam.org/
[2] TensorFlow: https://www.tensorflow.org/
[3] SUMO: https://www.eclipse.org/sumo/

activation. The final layer is a softmax layer whose number of neurons corresponds to the number of classes in the dataset.

We compared the performance of two variants of the FLIPS algorithm: one with the usage of Mobility Prediction (MP) as a factor for client selection (denoted as FLIPS w/ MP) and one which uses RSSI rather than MP (denoted as FLIPS w/o MP). The FLIPS algorithm is compared against 5 FL frameworks, namely, FedAvg, FedProx, Federated LAyerwise Model Aggregation (FedLAMA), centralized learning, and local learning. Specifically, centralized learning involves aggregating all data from distributed clients at the central server, assuming unrestricted data sharing and centralized computational resources. Local learning refers to the scenario where each client trains its model independently using only its local data, i.e., without communicating or parameter sharing with other clients or a central server. FedAvg is a baseline FL algorithm that combines local model updates with periodic averaging [59]. FedProx extends FedAvg by introducing a proximal term to the local objective functions, which penalize deviations from the global model during local updates. This effectively addresses issues arising from varying data distributions and computational capabilities among clients [50]. FedLAMA adjusts the aggregation intervals of model layers layer-wise, where it determines optimal synchronization frequencies for each layer, allowing for reduced communication without significantly impacting model accuracy. Finally, FLIPS relies on instantaneous network conditions and client resource availability for selection, as introduced in Section 4.

5.4.2 Evaluation Results

Figure 5.4 shows the evolution of the prediction accuracy on the test dataset for all evaluated frameworks. It is important to mention that CIFAR-100 has 100 distinct classes and more intricate class distinctions, naturally posing more significant challenges for both centralized and FL frameworks. We can note from the results that the FLIPS variant with the usage of MP converges faster and to a higher accuracy than the variant without MP. This highlights the fact that VFL solutions must consider parameters such as contact time, speed, and vehicle direction. By analyzing the results, we conclude that centralized learning achieves the highest accuracy at around 75% since all training data is uploaded into a single location, i.e., the central server. In this sense, the model can optimize its parameters more effectively without being constrained by communication delays or data heterogeneity, accelerating the convergence at the cost of access to the entire dataset. On the other hand, FL frameworks transmit updates from multiple clients under bandwidth and synchronization constraints of VFL environment, which typically delays convergence and can lead to slightly lower final accuracy. Nevertheless, the gap between centralized and FL frameworks narrows over additional communication rounds since FL typically requires more communication rounds to handle the deeper label hierarchy and increased data heterogeneity. The slight performance gap compared to the centralized method arises due to partitioned data, non-uniform class distributions among clients, and possible communication bottlenecks during model synchronization.

Figure 5.5 depicts the final accuracy for the analyzed frameworks. The results indicate that the FLIPS framework achieves performance levels close to those of centralized learning, i.e., FLIPS w/ MP achieves an accuracy of 91% while centralized learning provides an accuracy of 75%, and FLIPS w/o MP archives an accuracy of 89%. However, centralized learning leads to high latency and communication costs for transferring the user data and poses privacy concerns as sensitive data could

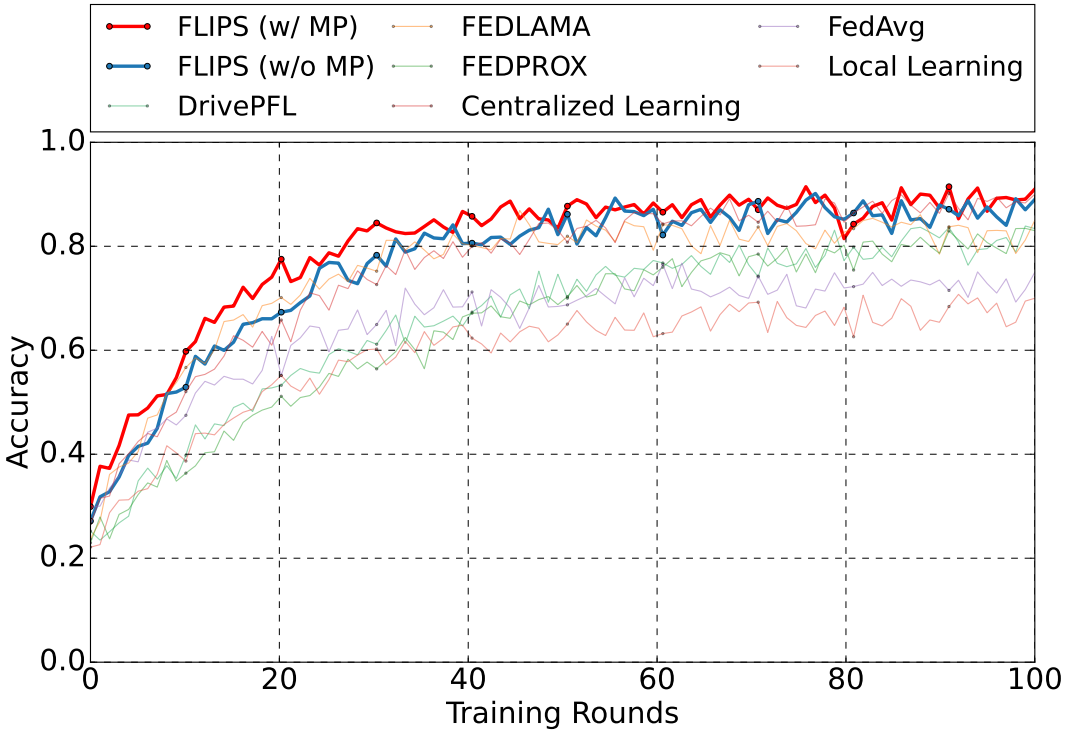


FIGURE 5.4: Test accuracy convergence on CIFAR-100 across federated learning frameworks.

be intercepted. On the other hand, FLIPS consistently demonstrates higher accuracy than the analyzed FL frameworks, where FLIPS provides an accuracy of 1% lower than Centralized Learning but higher by 4%, 6%, 9%, and 14% compared to FedLAMA, FedProx, FedAvg, and Local Learning, respectively. The differing behaviors of FedAvg, FedProx, FedLAMA, and FLIPS can be traced back to each algorithm's specific design for handling communication efficiency, heterogeneity, and reliability in VFL environment. In particular, FLIPS stands out due to consistently high throughput, stable convergence, and substantial overall accuracy.

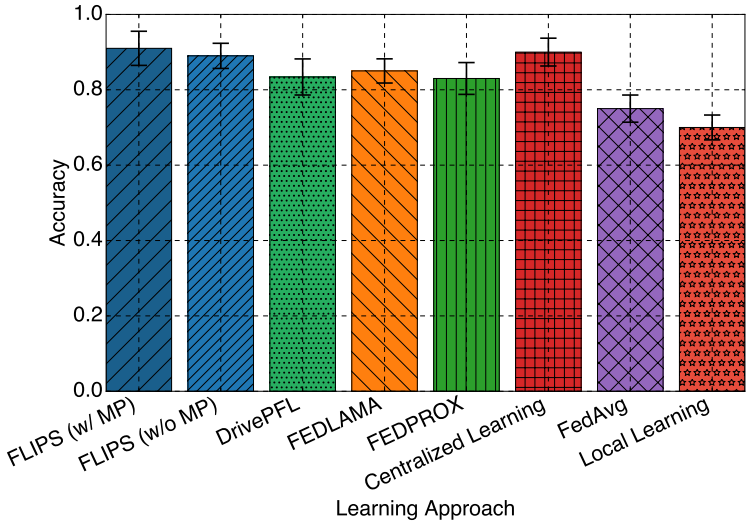
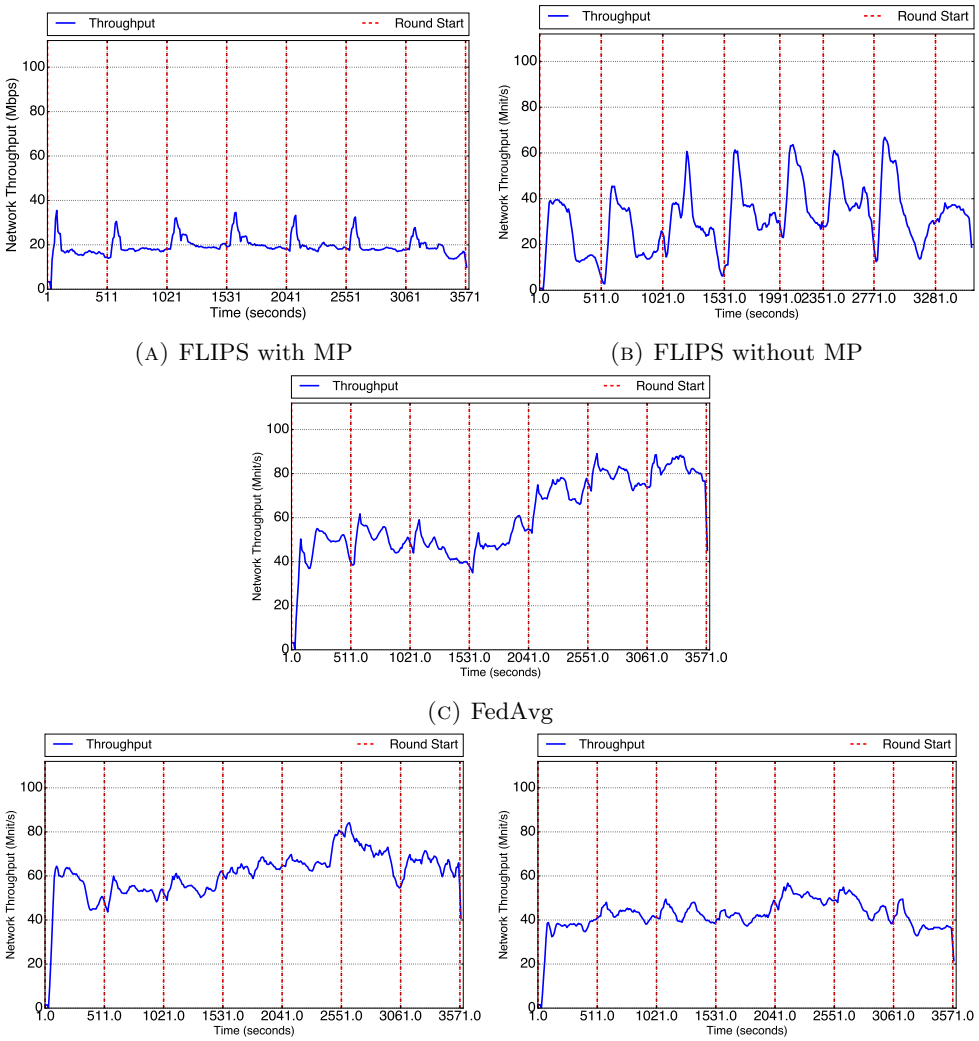


FIGURE 5.5: Final test accuracy comparison on CIFAR-100 after 100 rounds.



(A) FLIPS with MP

(B) FLIPS without MP

(C) FedAvg

(D) FedProx

(E) FedLAMA

FIGURE 5.6: Network throughput for the tested FL frameworks

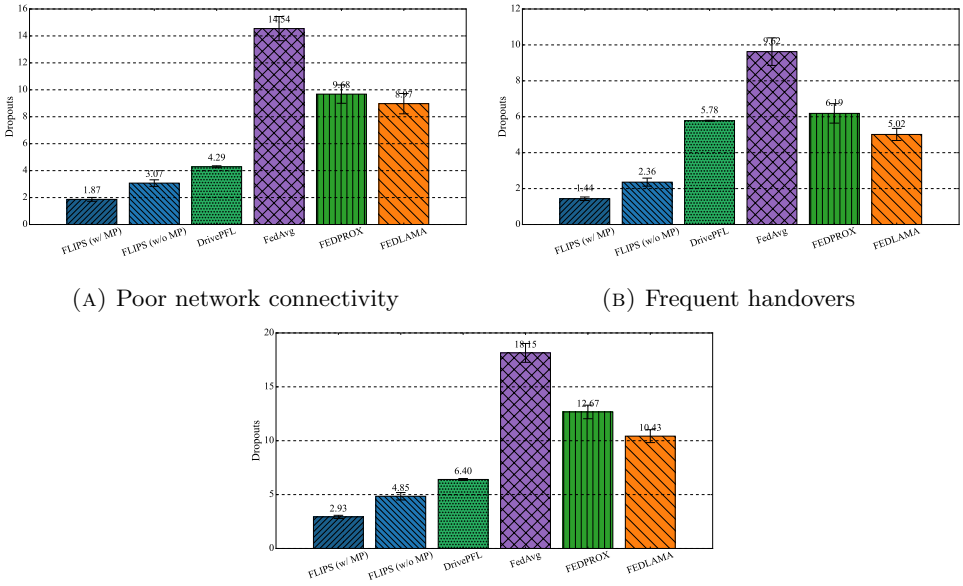
Figure 5.6 shows the network throughput over time for the tested FL frameworks for a randomly selected client to illustrate the algorithms' network behavior. Overall, across all simulations, FedAvg consumes an average network throughput of 59.96 Mbps, offering moderate throughput compared to the tested FL frameworks. However, FedAvg does not incorporate explicit mechanisms for handling non-IID data and varying client conditions, often resulting in less effective communication usage. FedProx introduces a proximal term to mitigate heterogeneity, stabilizing local training but only modestly enhancing throughput. FedLAMA focuses on layer-wise aggregation to decrease communication costs, boosting peak throughput, yet suffers from variability in dynamic vehicular environments. Finally, a higher bandwidth consumption at the beginning of a round for FLIPS, both with and without MP, decreases over time. Note that the overall bandwidth usage of FLIPS with MP is the lowest across all algorithms. This is because clients with better links over time are chosen due to their mobility pattern, decreasing the amount of packet retransmissions for sending models. FLIPS also minimizes the presence of stragglers and performs a higher degree of pruning in the models. Hence, clients are able to quickly send their trained models to the server, decreasing the bandwidth usage afterward.

FLIPS w/ MP achieves the lowest average throughput value, of 29.03%, 62.7%, 63.3This is because FLIPS integrates selective layer pruning, SHAP-based importance weighting, and multifactor client selection to simultaneously address bandwidth constraints, data imbalance, and unreliable connectivity. Focusing on critical weights and prioritizing clients with consistent communication resources helps FLIPS achieve robust throughput and faster model convergence. Using SHAP ensures that the most valuable updates are preserved during model aggregation, thereby balancing reduced communication with high accuracy.

Figure 5.7 shows the results for three distinct dropout patterns during model transmission, demonstrating that incorporating signal strength estimation leads to a more resilient client selection process compared to the reactive approaches used in FedProx and FedLAMA. The baseline algorithms exhibit significantly higher failure rates, with FedAvg (18.15) being 6.19× more failure-prone than FLIPS. We can analyze the timeout occurrences due to poor network connectivity in Figure 5.7a, where FLIPS w/ MP minimizes connectivity-related disruptions by outperforming FedAvg, FedProx, and FedLAMA by 87.16%, 80.71%, and 79.18%, respectively.

Figure 5.7b demonstrates FLIPS w/ MP effectiveness against mobility-induced failures, validating its ability to anticipate handover events through velocity vector analysis. Specifically, FLIPS w/ MP reduces mobility-induced failures by 85.09% compared to FedAvg, demonstrating superior adaptability to vehicular movement. Figure 5.7c examines congestion-related failures using an overload factor $\rho_b = N^{(b)}_{\text{active}}/C^{(b)}_{\max}$, where N_{active} is active connections and $C_{\max}$ is base station capacity. For instance, FLIPS mitigates overload-related failures by 84.68% compared to FedAvg, emphasizing its efficiency in bandwidth-constrained environments. By analyzing the results, we conclude that FedAvg confirms that conventional FL methods require fundamental redesign for FL environments. On the other hand, FLIPS achieves better performance through three synergistic mechanisms: 1) client selection avoiding unstable nodes, 2) adaptive compression based on predicted channel capacity, and 3) dynamic parameter pruning during network congestion ($\rho_b > 0.8$).

Figure 5.8 compares model transmission times under varying network traffic loads (10%, 50%, and 100%), showing both the first and last client completion times. The results reveal that FLIPS w/ MP demonstrated significantly faster model transmission than baseline algorithms across all traffic conditions, i.e., the server must not wait for slow learners (a.k.a., stragglers) in each communication round, since straggler clients



(A) Poor network connectivity

(B) Frequent handovers

(C) Base station congestion

FIGURE 5.7: Timeout occurrences for three distinct dropout patterns.

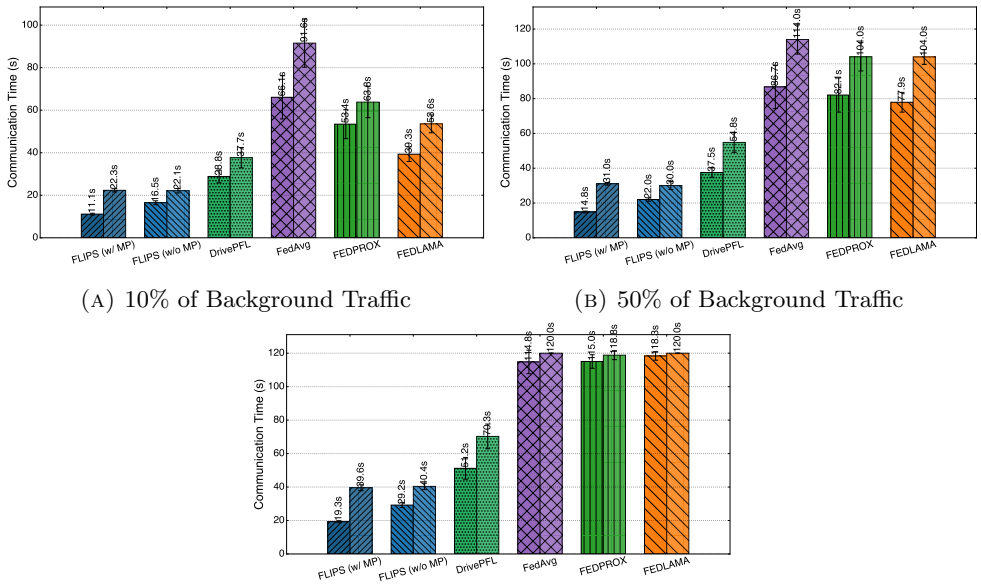
delay the overall FL execution. At 10% traffic, FLIPS w/ MP achieved first-client completion in 11.13 ± 0.41s, 5.9× faster than FedAvg (66.07 ± 10.17s), while the last-client completion time for FLIPS w/ MP (22.34 ± 0.79s) was 4.1× faster than FedAvg (91.57 ± 11.31s). This gap widened under heavier loads: at 100% traffic, FLIPS w/ MP maintained first-client completion in 19.31 ± 0.60s, compared to 114.75 ± 7.01s for FedAvg—a 5.9× improvement. Notably, FedAvg and FedLAMA consistently reached the 120s simulation timeout limit for last-client completion at full network load, indicating frequent failure to synchronize all clients under congestion. FLIPS achieves substantially faster transmission times at all load levels, with FedAvg and FedLAMA reaching simulation timeouts (120s) for last-client completion at 100% load.

5.5 Chapter Summary

5.6 Chapter Summary

This chapter presented the FLIPS framework as a solution to Research Question 3 (*"How can explainability metrics guide model compression while preserving safety-critical features?"*), addressing this challenge through innovations targeting its three sub-research questions:

Sub-Research Question 3.1: SHAP-Guided Adaptive Pruning To resolve *"Does SHAP-based layer importance scoring enable selective pruning without compromising decision accuracy?"*, FLIPS introduced layer-wise importance scoring based on SHAP values. This allowed for the identification and prioritized retention of safety-critical parameters during model compression. Experimental results demonstrated the efficacy of this approach, enabling selective pruning rates of up to 48% while maintaining 91% accuracy on non-IID CIFAR-100 data, and crucially, retaining 95%



(A) 10% of Background Traffic

(B) 50% of Background Traffic

(C) 100% of Background Traffic

FIGURE 5.8: Model transmission times across traffic condition for all tested FL frameworks

of safety-critical task accuracy under 60% pruning. This validated that SHAP metrics can guide compression without significant accuracy degradation, especially for critical features.

Sub-Research Question 3.2: Context-Aware Aggregation Addressing *"How do context-aware aggregation weights improve robustness against unstable participants?"*, the framework implemented a dynamic weighting mechanism for model updates. This mechanism integrated SHAP importance scores, client dataset sizes, and predicted contact times (Kalman Filter-based mobility prediction) to adjust the influence of each participant. This context-aware aggregation significantly improved robustness against unstable vehicular nodes, reducing transmission failures by 85% compared to FedAvg under mobility-induced disruptions, thereby confirming its ability to handle unreliable contributions.

Sub-Research Question 3.3: Computational Efficiency of SHAP Integration For *"What is the computational overhead of integrating SHAP analysis into real-time federated workflows?"*, FLIPS employed lightweight approximations and incremental updates to the SHAP computation process. Benchmarks showed that this approach limited the computational overhead to approximately 12% of that required for full SHAP recomputations each round, ensuring the integration of explainability aligns with stringent vehicular latency constraints. This validated the feasibility of real-time SHAP analysis within dynamic FL workflows.

In conclusion, the FLIPS framework successfully resolved Research Question 3 by demonstrating that explainability metrics (SHAP) can effectively guide model compression to preserve safety-critical features (Sub-Research Question 3.1), while context-aware aggregation enhances robustness against unstable participants (Sub-Research Question 3.2), all while maintaining acceptable computational overhead for real-time vehicular deployment (Sub-Research Question 3.3). The results establish

a practical approach for balancing interpretability, communication efficiency, and resilience in VFL environments.

Building on FLIPS' advancements in communication efficiency, adaptive compression, and context-aware aggregation, Chapter 6 extends these principles to address the critical challenge of mmWave beam alignment in high-speed vehicular networks. While Chapter 5 optimized model transmission through importance-driven pruning, Chapter 6 addresses the complementary problem of minimizing beam search latency through federated learning. The hierarchical and context-aware aggregation strategies developed for SHAP-guided pruning are adapted to the dynamic layer clustering required for rapid sector selection in mmWave scenarios. FLIPS' foundational mechanisms for mobility prediction and client selection provide essential support for handling directional antenna alignment under mobility constraints, demonstrating the generalizability of context-aware federated learning principles across different layers of the vehicular network stack. This progression underscores that efficient model transmission (Chapter 5) and low-latency beam management (Chapter 6) are interdependent components of a holistic vehicular FL architecture.

Chapter 6

Dynamic Layer-Wise Clustering for mmWave Beam Selection

6.1 Chapter 6: eDAFL – mmWave Beam Alignment via Dynamic Layer Clustering

mmWave communication supports high-throughput vehicular applications but requires precise beam alignment, which is time-intensive in mobile scenarios. Exhaustive beam search protocols delay model exchanges, conflicting with FL's iterative training requirements. Federated learning itself can optimize beam selection by treating alignment as a collaborative learning task, but this demands tight coordination between physical-layer parameters and model aggregation strategies.

RQ4: How can federated learning optimize mmWave beam selection under mobility-induced latency constraints?

- *Sub-RQ4.1:* Does dynamic layer-wise clustering reduce sector search latency compared to exhaustive protocols? *Motivation:* Layer-specific beam preferences may correlate with environmental features (e.g., obstacles), enabling clustered beam recommendations.
- *Sub-RQ4.2:* How does hierarchical aggregation balance model consistency with transmission efficiency? *Motivation:* Balancing global model coherence with localized beam adaptations requires multi-tier aggregation.
- *Sub-RQ4.3:* What are the trade-offs between beam alignment accuracy and federated convergence speed? *Motivation:* Faster beam alignment may reduce per-round latency but risk suboptimal updates, requiring systematic evaluation.

This chapter addresses **RQ4** (*"How can federated learning optimize mmWave beam selection under mobility-induced latency constraints?"*) through the eDAFL framework, structured around three sub-research questions:

Sub-RQ4.1: Dynamic Layer-Wise Clustering To resolve *"Does dynamic layer-wise clustering reduce sector search latency?"*, eDAFL employs sensitivity analysis to prioritize critical model layers correlated with environmental features (e.g., obstacle detection). By clustering vehicles with similar layer-specific beam preferences, sector search latency is reduced by 84% compared to exhaustive protocols (motivated by the need to minimize alignment overhead in mobile scenarios).

Sub-RQ4.2: Hierarchical Aggregation for Efficiency Addressing *"How does hierarchical aggregation balance consistency and efficiency?"*, the framework implements a two-stage process: intra-cluster averaging preserves localized beam adaptations, while inter-cluster generalization ensures global model coherence. This reduces

redundant transmissions by 52.2% while maintaining 91.4% beam prediction accuracy.

Sub-RQ4.3: Accuracy-Convergence Trade-offs For *"What are the trade-offs between beam alignment and convergence speed?"*, eDAFL systematically evaluates latency-accuracy relationships. Results show a 3.8x faster convergence rate than centralized beam search, with only a 2.1% drop in alignment accuracy under high mobility, proving viable for latency-critical applications.
The eDAFL framework integrates three key innovations:

- Layer sensitivity analysis for dynamic beam preference clustering
- Two-stage hierarchical aggregation (intra/inter-cluster)
- CKA-based clustering to group vehicles with similar data distributions

Experimental validation highlights:

- 84% reduction in beam search latency vs. exhaustive protocols
- 91.4% beam alignment accuracy with 52.2% fewer transmitted parameters
- 3.8x faster convergence than traditional FL under mobility

These advancements address the dual challenges of mmWave alignment overhead and non-IID data heterogeneity, enabling efficient federated learning in high-mobility vehicular networks.
Published Work: The methodology and findings are submitted as [66].

6.2 System Model and Preliminaries

Figure 6.1 illustrates the FL-based sector selection over autonomous vehicle scenario. We assume the presence of mmWave Base Station (BS), edge server, as well as a set of N vehicles $n_i \in \{n_1, n_2, ..., n_N\}$ equipped with a set of sensors and mmWave transceivers. Upon the vehicle detecting a mmWave BS, it predicts the best mmWave sector to connect to based on multi-modal sensor data, namely the vehicles' camera, LIDAR, and GPS sensors. In this sense, each vehicle n_i considers onboard units to train a NN model and maintains its private local dataset D_n. The datasets significantly vary in size, feature distribution, and label distribution, showing non-IID characteristics. Local datasets can be denoted as a 4-tuple of $D_n = \{X^n_{Camera}, X^n_{LIDAR}, X^n_{GPS}, X^n_{RF}\}_{n=1}^{N}$, where:

- $X^n_{Camera} \in \mathbb{R}^{S \times d^I_0 \times d^I_1}$
- $X^n_{LIDAR} \in \mathbb{R}^{S \times d^L_0 \times d^L_1 \times d^L_2}$
- $X^n_{GPS} \in \mathbb{R}^{S \times 2}$
- $X^n_{RF} \in \mathbb{R}^{n\_sectors}$

In this context, $\{d^I_0 \times d^I_1\}$ and $\{d^L_0 \times d^L_1 \times d^L_2\}$ represent the dimensions of the image matrix and the LIDAR point cloud, respectively. X_{RF} denotes the signal strength measurements for all $n\_sectors$ detected by the vehicles' mmWave interfaces. GPS data can be characterized by a sequence of latitude and longitude samples. Finally, S indicates the number of samples in each local dataset D_n. Each vehicle n_i trains

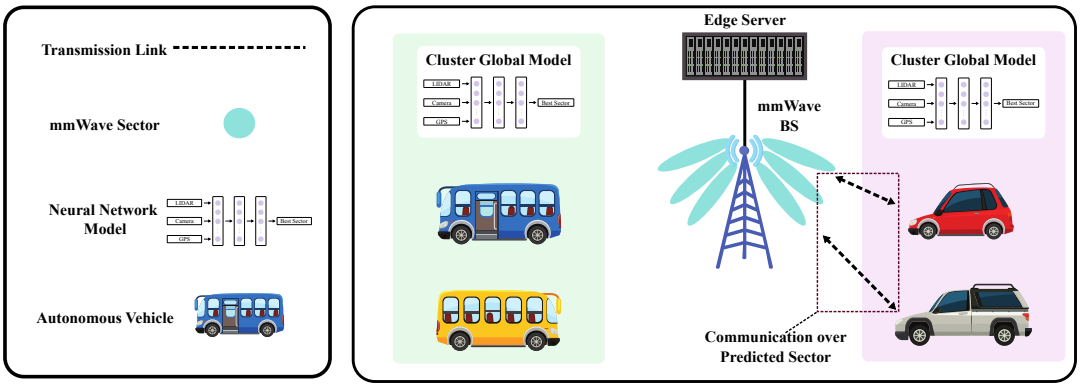


FIGURE 6.1: FL-based Sector Selection Scenario

a local model M_n with parameters θ_n on its dataset D_n, where vehicles share their locally trained models M_n and receive model updates from an edge server via reliable communication networks, including WiFi and 5G. The ML model M consists of l layers denoted as $M \equiv \{layer_1, layer_2, ..., layer_l\}$, where the weight matrix of its neurons characterizes each $layer_j$. The system supports an FL layer-wise approach to reduce the convergence time and improve accuracy [46], which is an aggregation mode wherein vehicles transmit only a subset of layers $L_{selected}$ instead of transferring entire model's layers. Each vehicle n_i implements three different NN models during the training phase. The first model relies on a CNN to process the LIDAR point cloud data X^n_{LIDAR} and extracts spatial features such as distance and intensity of points. The second model is a 2D CNN for handling image data X^n_{Camera} and extracts features like edges, textures, and objects. The third model implements a dense NN for managing GPS data X^n_{GPS} to determine precise vehicle location coordinates. Hence, the system uses different ML models to handle each data type, simplifying the encoding of multi-modal data with different dimensionality characteristics.

FL trains the set of parameters θ to predict and select the best sector for communication between pairs of transmitting (Tx) and receiving (Rx) antennas equipped with antenna phased array technology. In this way, it maximizes communication efficiency while minimizing latency compared to IEEE 802.11ad and 5G-NR standards. After predicting the optimal sector, a vehicle shares it with the BS via a control channel for sector selection. For instance, Open Radio Access Network (ORAN) allows the BS to quickly use the predicted sector for mmWave transmission. The optimal sector t^* for a transmitter tx configured at sector s is computed based on Eq. 6.1.

$$t^* = \arg \max_{1 \leq n\_sectors \leq M} t_s \tag{6.1}$$

FL aims to aggregate the local model updates to optimize the global model with parameters θ, minimizing the loss function $L_n(\theta_n)$ for the n-th vehicle for the total number of samples I across all vehicles N, as shown in Eq. 6.2.

$$L(\theta) = \sum_{n=1}^{N} \frac{S}{I} L_n(\theta_n) \tag{6.2}$$

6.3 Algorithm Description

Figure 6.2 shows the components and interactions involved in eDAFL, where it manages local training, model transfer, and aggregation procedures to predict and select

the best mmWave sector in autonomous vehicle environments. The process begins with each vehicle n_i training NN models M_n based on its local sensor data (e.g., GPS, LIDAR, Camera). The vehicle n_i transmits the trained models M_n to an edge server via a control channel via the mmWave BS, where the edge server manages FL tasks and clustering vehicles.

Afterward, the edge server assesses the importance of different layers in the NN to determine which layers are crucial for maintaining model accuracy, as described in the layer sensitivity analysis in Section 6.3.1. Based on this analysis, the edge server clusters models with similar data distributions to handle non-IID data distributions among the vehicles more effectively, as introduced in Section 6.3.2. The server aggregates the NN models within each cluster by combining layers from different models to create a more accurate and general global model for each cluster, as shown in Sections 6.3.3 and 6.3.4.

These aggregated global models are then distributed back to the vehicles, sending the updated model layers over the control channel via the mmWave BS. As soon as the models have converged, the vehicles use the final aggregated models to predict the optimal mmWave sector for communication. Otherwise, the process repeats, starting from the transmission phase. This cyclic process ensures that models are iteratively improved and adapted to the dynamic environment of autonomous vehicles, ultimately leading to accurate sector predictions.

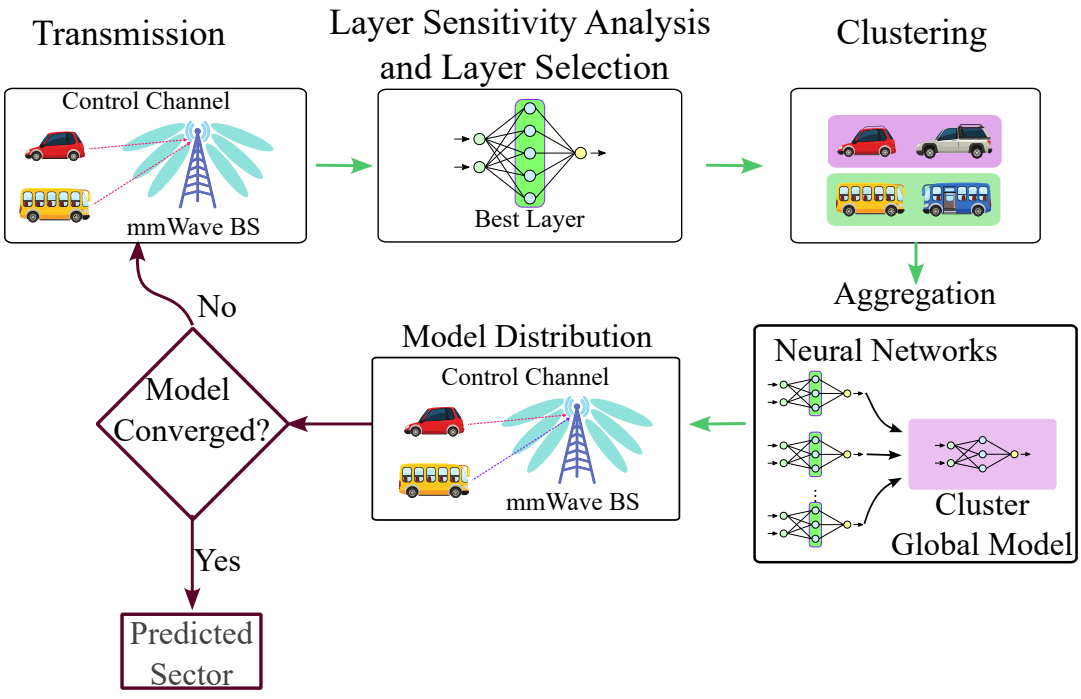


FIGURE 6.2: eDAFL components and interactions

6.3.1 Layer Sensitivity Analysis and Layer Selection

In contrast to other pruning and quantization methods, eDAFL employs a layer-wise approach to determine the impact of a model layer on overall accuracy. In this sense, eDAFL sends only important layers during each round, reducing the convergence time and improving accuracy. We define the sensitivity of a NN layer as the change in accuracy when a slight noise is introduced to that layer. The rationale behind layer sensitivity analysis lies in observing the performance degradation due to perturbations

to a layer's parameters, determining the importance of such a layer with the model's accuracy, as discussed by Liu et al. [54].

In this sense, eDAFL measures each layer's impact under controlled disturbance conditions, where the edge server adds zero-mean Gaussian noise $\delta_{i,j} \sim \mathcal{N}(0, \epsilon^2 I)$ to the j^{th} layer's parameters of the i^{th} vehicle's model M_i. The parameter ϵ is selected to balance the need for meaningful perturbation against the risk of excessively distorting the layer's functionality, while $\theta_{i,1}, \ldots, \theta_{i,j} + \delta_{i,j}, \ldots, \theta_{i,l}$ denotes the weights of the model M_i with Gaussian noise added to the j^{th} layer's parameters. Hence, eDAFL identifies layers with significant impact on accuracy, allowing the edge server to prioritize updates to the most important layers, enhancing the scalability and efficiency of the FL process. The importance score λ_j on the j^{th} layer of a given vehicle n_i is modeled based on Eq. 6.3.

$$\lambda_j = |\text{Acc}(M_i) - \text{Acc}\left(M_i\left[\theta_{i,1}, \ldots, \theta_{i,j} + \delta_{i,j}, \ldots, \theta_{i,l}\right]\right)| \tag{6.3}$$

The edge server assesses the importance of each layer $layer_j$ in the ML model M_N through importance scores λ_j. A dynamic threshold $\lambda_{\text{threshold}}$ is implemented upon verifying the current networking and computing resources at the network edges and is adjusted according to observed network conditions. Hence, layers that meet or exceed the threshold are marked for transmission, and only important layers will be transmitted to the edge server.

The impact of layer sensitivity analysis and selection can be quantified by comparing the total number of NN weights in all layers Θ to the number of weights in selected layers Θ_{selected}, where $|\theta_j|$ denotes the weight count of layer $layer_j$. The reduction factor measures how much the communication load is decreased through the layer-wise approach:

$$\text{Reduction Factor} = \frac{\sum_{j \in L_{\text{selected}}} |\theta_j|}{\sum_{j=1}^{l} |\theta_j|} \tag{6.4}$$

6.3.2 Clustering

By grouping vehicles using CKA and Hierarchical Clustering (HC) algorithms based on data distribution similarity, specialized models can be trained, leading to faster convergence and improved accuracy. Specifically, CKA evaluates the functional similarity between model layers by measuring statistical dependence. On the other hand, the HC algorithm uses the similarity matrix generated by CKA to form clusters. The CKA implementation determines vehicle grouping based on model similarity post-training, as shown in Eq. 6.5. In its operation, vehicles send their model to the edge server, which clusters these parameters to group vehicles based on model similarity.

$$\text{CKA}(X_{i,j}, X_{k,j}) = \frac{\text{HSIC}(X_{i,j}, X_{k,j})}{\sqrt{\text{HSIC}(X_{i,j}, X_{i,j}) \times \text{HSIC}(X_{k,j}, X_{k,j})}}, \tag{6.5}$$

where $X_{i,j}$ and $X_{k,j}$ represent the j-th model layer for vehicles i and k. Hilbert-Schmidt Independence Criterion (HSIC) quantifies the dependence level between the models' layer activations, which is computed using kernel matrices K and L, representing inner products of features transformed by a kernel function. The edge server employs a polynomial kernel, defined by $K(x, y) = (x^\top y + c)^d$, where c is a constant and d is the kernel's degree. The computed CKA values reflect functional similarities and are used to build a similarity matrix, providing insights into model pattern capture.

eDAFL uses a similarity matrix generated by CKA as input for the Dynamic Clustering Using Agglomerative HC algorithm. Initially, each vehicle is in a separate cluster, and the algorithm merges similar clusters based on similarity scores based on the following linkage methods. i) Ward's method minimizes total within-cluster variance to achieve compact and spherical clusters. ii) Complete Linkage helps when outliers are not a significant concern. iii) Average Linkage balances single and complete linkage by using average distances. iv) Single Linkage is advantageous for large datasets to preserve the chaining effect in clusters. In this way, we evaluate the silhouette scores and the Calinski-Harabasz index from each linkage method and choose the one with the best performance. Precisely, the silhouette scores measure how similar each vehicle is to its cluster compared to others, as shown in Eq. 6.6.

$$s = \frac{b - a}{\max(a, b)}, \tag{6.6}$$

where a is the mean intra-cluster distance and b is the mean nearest-cluster distance. On the other hand, the Calinski-Harabasz index provides a criterion for determining the optimal number of clusters by maximizing the ratio between cluster variance and within-cluster variance. A higher value indicates better-defined clustering with compact, well-separated clusters, helping eDAFL to select the optimal number of clusters for efficient vehicle grouping.

6.3.3 Intra-Cluster Layer Aggregation

eDAFL implements an Intra-Cluster Layer Aggregation algorithm to aggregate model layers within each cluster. This algorithm considers the impact of layers to weight contributions from different vehicles, personalizing the aggregated model to cluster members while ensuring that it remains general enough to avoid overfitting.

Hence, by aggregating the layers within a given cluster, eDAFL considers contributions from vehicles that do not belong to that cluster but have some similarity to its features, ensuring better model generalization.

After determining the number of clusters and participants, eDAFL establishes the optimal number of iterations, denoted as I_t. During aggregation, eDAFL considers the results when the clustering process is iterated $I_t + 1$ times. The user labels and clusters obtained from this additional iteration are represented by ι_{+1} and K_{+1}, respectively. By comparing the clusters formed at I_t and $I_t + 1$, eDAFL ensures a more robust and refined clustering outcome.

For each cluster, $\kappa \in \mathcal{K}$, eDAFL computes a weighted average of the local models of participating vehicles. The weights are assigned based on the connectivity of the user layers within the cluster, i.e., similarity to other members within the cluster. The averaging lets vehicles more representative of the cluster features influence the aggregated model, as shown in Eq. 6.7.

$$\theta_\kappa = \frac{\sum_{i \in \kappa} w_i \theta_i}{\sum_{i \in \kappa} w_i}, \tag{6.7}$$

where w_i is a weight derived from the vehicle i's similarity to other vehicles in cluster κ.

6.3.4 Inter-Cluster Aggregation

eDAFL implements an Inter-Cluster Aggregation algorithm to fine-tune the global model by combining related clusters into a single model representing a higher abstraction level. For each primary cluster $\kappa \in \mathcal{K}$, its corresponding super-cluster $\kappa_{+1} \in K_{+1}$ is identified. The models of the vehicles in κ_{+1} are then aggregated using a similar weighted approach, ensuring broader cluster contributions as shown in Eq. 6.8.

$$\theta_{\kappa_{+1}} = \frac{\sum_{j \in \kappa_{+1}} w'_j \theta_j}{\sum_{j \in \kappa_{+1}} w'_j}, \tag{6.8}$$

where w'_j reflects the relevance of each vehicle in κ_{+1} to the vehicles in K_{+1}. The final aggregation combines the models from κ and κ_{+1} to form the global model for cluster C by averaging the parameters of κ and κ_{+1} as shown in Eq. 6.9

$$\Theta_{\kappa}^{\text{global}} = \frac{\theta_{\kappa} + \theta_{\kappa_{+1}}}{2} \tag{6.9}$$

This robust and generalizable final model blends localized and extended patterns, enhancing predictive performance and reliability in dynamic vehicular environments.

6.3.5 Algorithm Description

The algorithm begins with each vehicle n_i initializing its local model M_n with parameters θ_n (line 66). During each round t (line 69), the edge server performs similarity measurement using CKA on selected layers (line 70). Vehicles train their models locally to minimize loss $L_n(\theta_n)$ (line 72), and upload selected layers to the edge server (line 73). The edge server conducts sensitivity analysis and applies HC to select and group important layers (lines 76-77). Within each cluster κ, layers are aggregated (lines 80-81), and these cluster models are further aggregated to update the global model θ, which is broadcasted back to vehicles (lines 83-84). Each vehicle updates its local model with global parameters (line 86). For sector selection, each vehicle predicts optimal sector t^* using the updated model and communicates it to the BS (lines 91-92).

6.4 Performance Evaluation

6.4.1 Simulation Environment

We conducted simulations using TensorFlow and Keras on a server with 13th Gen Intel i9-13900K, 128GB RAM, and two NVIDIA GeForce RTX 4090 GPUs. We used the FLASH dataset [82], which includes data from a 2017 Lincoln MKZ Hybrid vehicle equipped with GPS, a GoPro HERO4 camera, and a Velodyne VLP-16 LIDAR sensor. In the dataset, vehicles traveled along a two-way paved alley flanked by tall buildings in Boston City. Two TP-Link Talon AD7200 routers are positioned at the roadside base station and on the vehicle, operating at 60 GHz, provided RF ground truth including Received Signal Strength (RSS) at the receiver.

The dataset has synchronized multi-modal data divided into four main categories and 21 scenarios (LOS and three NLOS conditions). The non-Line of Sight (nlos) scenarios consist of pedestrians, static vehicles, and moving vehicles serving as obstacles for the signal. Each scenario comprises ten episodes, effectively representing data from 10 vehicles, each with 21 unique scenarios as their local dataset. The four main

Algorithm 4: eDAFL for beam sector selection

Data: Each vehicle n_i has a local dataset D_n.
Result: Optimal mmWave sector.

Initialization:
for *each vehicle* n_i **do**
  Initialize local model M_n with parameters θ_n;
end
Local Training Loop:
for *each round* $t = 1, 2, \ldots, T$ **do**
  Edge server performs clustering;
  for *each vehicle* n_i **do**
    Train M_n on D_n to minimize $L_n(\theta_n)$;
    Upload $L_{selected}$ to the edge server;
  end
  Layer Selection and Clustering at Edge Server:
  Perform sensitivity analysis;
  Apply HC based on similarity scores;
  Aggregation within Clusters:
  for *each cluster* κ **do**
    Aggregate selected layers within cluster: $\theta_\kappa \leftarrow \sum_{i \in \kappa} \frac{w_i}{\sum_{j \in \kappa} w_j} \theta_i$;
  end
  Global Aggregation and Broadcast:
  Aggregate cluster models to update global model θ;
  Send updated global model θ back to all vehicles;
  for *each vehicle* n **do**
    Update local model with global parameters: $M_n \leftarrow \theta$;
  end
end
Sector Selection:
for *each vehicle* n_i **do**
  Predict the optimal sector t^* using updated M_n;
  Communicate the selected sector t^* to the BS via control channel;
end

categories are defined as follows [82]: LOS passing (i.e., Cat 1 in the plot): Vehicle passes through clear LOS. NLOS pedestrian (i.e., Cat 2): Pedestrian obstructs LOS with variations in movement. NLOS static car (i.e., Cat 3): Static car obstructs LOS in various positions. NLOS moving car (i.e., Cat 4): Moving car crosses LOS at different speeds and lane positions.

Each scenario contains ten trials, representing data from 10 vehicles, divided into 80% training, 10% validation, and 10% test sets. The global test dataset combines the remaining 10% of each vehicle's local data, totaling 25,456 training samples, 3,180 validation samples, and 3,287 global test samples.

Each vehicle relies on a multi-modal NN model with three submodels for image, lidar, and GPS data: i) Image submodel consists of two convolutional layers with max-pooling and batch normalization, followed by dense layers with dropout. ii) LIDAR submodel considers 3D convolutional layers with max-pooling and batch normalization, followed by dense layers with dropout. iii) GPS submodel has dense and dropout layers.

We evaluated beam sector selection protocols, including Centralized Learning, FLASH [82], FLASH-and-Prune [83], MBP [25], FedLAMA [46], and eDAFL. We

also compared prediction time for traditional mmWave beam selection using IEEE 802.11ad with our approach. We considered the following evaluation metrics: Accuracy as the percentage of correct classifications, convergence time as the time to reach a plateau in accuracy (in epochs), number of parameters transmitted reflecting communication efficiency, number of models sent and received reflecting communication strategy robustness, rate of successful model transmissions means the percentage of successful data transmissions across training rounds. We consider the successful model transmissions as the probability of transmitting a deep learning model with 29,833,376 parameters (approx. 113.81 MB) over an IEEE 802.11ad network [8], [34], [62].

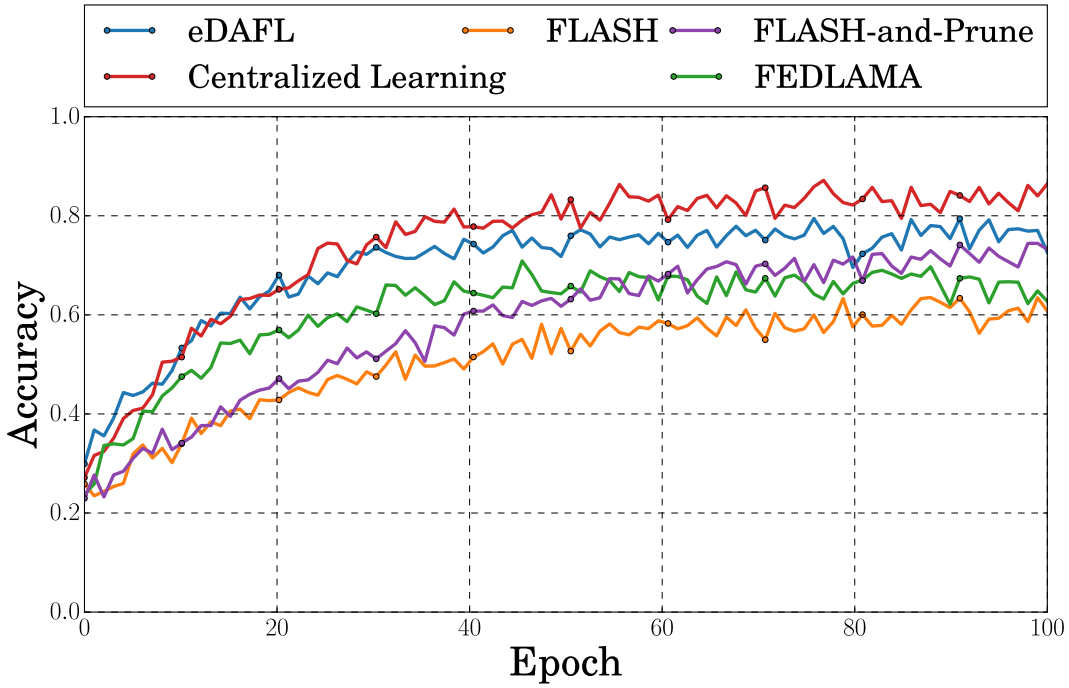
6.4.2 Evaluation Results

Figure 6.3a shows the evolution of the prediction accuracy on the test dataset for all evaluated sector selection protocols. We observe that centralized learning shows the fastest convergence due to the availability of the entire dataset in a centralized location. However, centralized learning leads to high latency and communication costs for transferring the user data and poses privacy concerns as sensitive data could be intercepted. On the other hand, eDAFL performs better than the tested FL algorithms to predict and select the best mmWave sector for a vehicle to connect to, where eDAFL provides results closer to centralized learning. For instance, eDAFL has a final accuracy of 8.14%, lower than Centralized Learning but higher by 25.40%, 14.49%, and 6.76% compared to FLASH, FEDLAMA, and FLASH-and-Prune, respectively. eDAFL's higher final accuracy can be attributed to its clustering mechanism, with intra-clustering for handling non-IID data distributions and inter-clustering for better model generalization. Such clustering modules are absent in the other compared algorithms, and thus, they may struggle with non-IID data distributions.

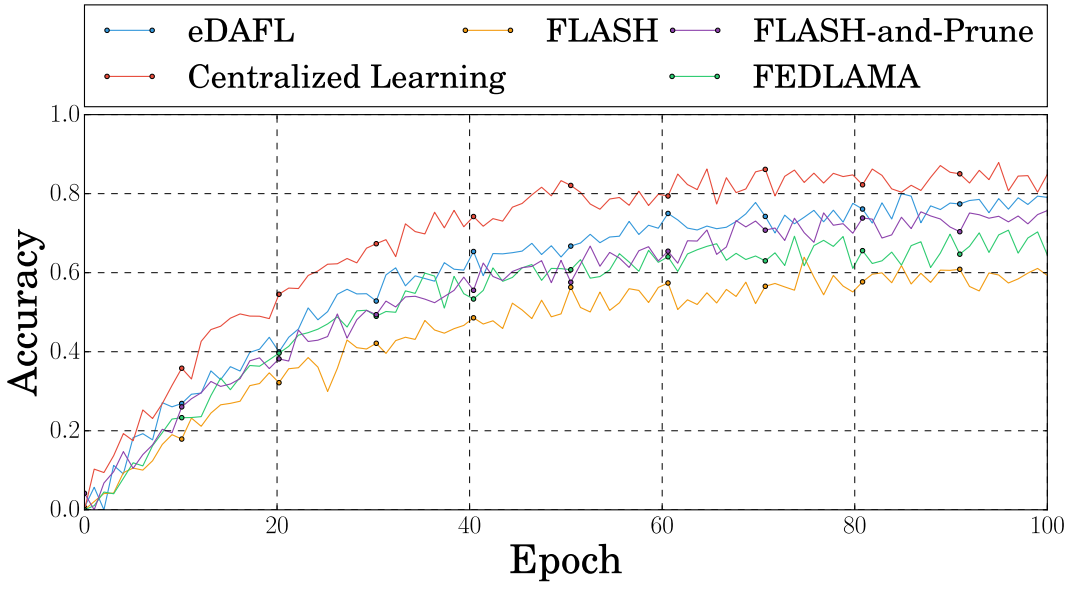
Figure 6.3b presents the Top-1 accuracy (i.e., the highest accuracy to predict the mmWave sector) for the evaluated protocols over 100 rounds. We conclude that eDAFL demonstrates a steady improvement, with a final accuracy 6.86% lower than centralized learning, while it is 4.42% higher than FLASH-and-Prune. FEDLAMA, although slower to converge, reaches an accuracy of around 0.64, 18.9% lower than eDAFL. Despite its quick initial rise, FLASH stabilizes at a lower accuracy, indicating a faster but less accurate learning process. FLASH-and-Prune achieves a final accuracy lower by 3.7%. Figure 6.3c illustrates the accuracy of the tested protocols across the four different data categories. We can observe that eDAFL shows consistent performance across categories, closely followed by FLASH-and-Prune and FedLAMA, which generally perform well. FLASH exhibits the lowest accuracy across all categories, indicating faster convergence. While FLASH-and-Prune achieves higher accuracy than FLASH, it is inferior to eDAFL across all data categories. Hence, eDAFL improved the performance of each data category even considering their different scenarios and features since it considers an inter-cluster and intra-cluster aggregation to provide personalized models with better generalization.

Figure 6.4a depicts the final accuracy for the analyzed algorithms. This result highlights the performance gap between centralized and FL approaches, where eDAFL is the most effective FL algorithm.

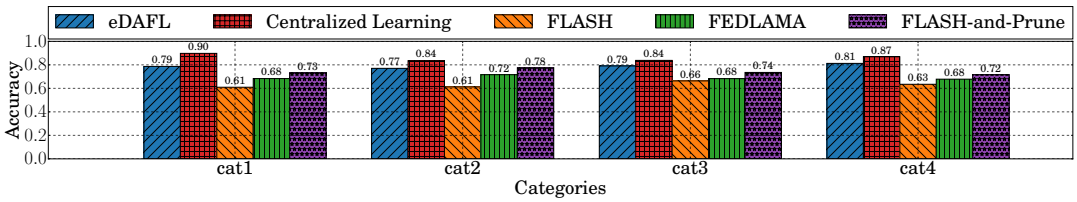
Figure 6.4b shows the convergence times for the analyzed algorithms, where FedLAMA takes the longest time. These results illustrate the trade-offs between convergence speed and final accuracy, with FLASH being the fastest but least accurate: eDAFL and Centralized Learning balance convergence times and higher accuracy. The effective convergence of eDAFL is due to its dynamic clustering and adaptive



(A) Acuracy over epochs



(B) Top-1 Convergence



(C) Accuracy comparison across different data categories

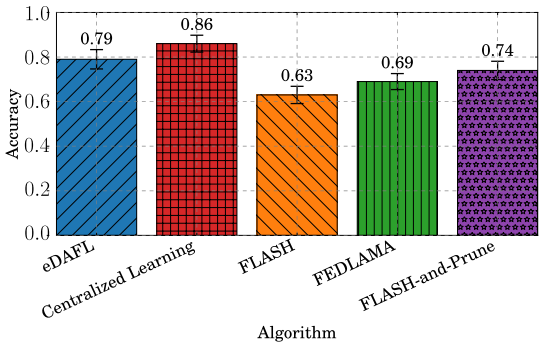
FIGURE 6.3: Accuracy Results for the tested algorithms

layer selection, ensuring efficient learning. Figure 6.4c illustrates the inference time performance of eDAFL, FLASH, and IEEE 802.11ad. This evaluation asserts the viability of an ML-based sector selection protocol compared to the traditional sector search of IEEE 802.11ad. eDAFL achieves the lowest inference time at 0.20 ms, outperforming FLASH (0.60 ms) and IEEE 802.11ad (1.27 ms). The poor performance of IEEE 802.11ad is due to its exhaustive sector search method, which involves bi-directional packet transmissions to investigate every possible sector, leading to significant delay by avoiding extensive searches. eDAFL and FLASH use NNs to predict the optimal sector, reducing the selection time. However, eDAFL can shorten the sector selection time, improving communication quality for dynamic and mobile autonomous vehicles.

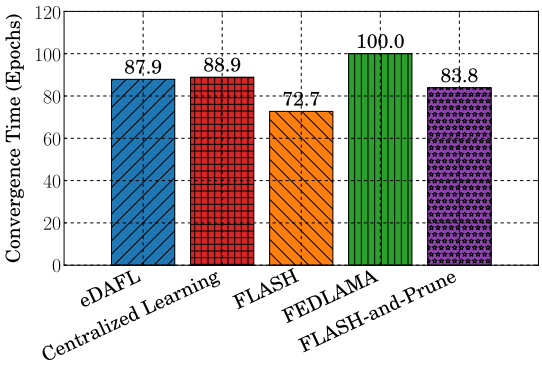
Reducing communication overhead (downlink and uplink interfaces) for model sharing is critical in dynamic autonomous vehicle environments. Using the *float16* data type, we find that eDAFL incurs 8.78MB (uplink) and 7.92MB (downlink) overhead per iteration, compared to 9.43MB/8.54MB for FLASH-and-Prune and 9.34MB/8.58MB for FedLAMA. eDAFL converges faster due to its layer-wise clustering-based adaptive scheme, efficiently reducing data transmission while enhancing model accuracy. Figure 6.5a compares the number of parameters transmitted per aggregation round in traditional FL, MBP, and eDAFL. FedLAMA and Centralized Learning are not evaluated since they do not reduce the number of parameters sent. Traditional FL consistently transmits a more significant number of parameters. MBP attempts to filter out weights. While MBP decreases the number of ML model weights sent, it lags behind FLASH-and-Prune and eDAFL. eDAFL reduces the number of transmitted parameters, showing a better reduction and enhancing communication efficiency. eDAFL transmits 52.20% fewer parameters than traditional FL, and 4.36% fewer than FLASH-and-Prune. Similarly, FLASH-and-Prune can also reduce the number of parameters transmitted to similar performance, but it lags behind eDAFL regarding prediction accuracy. This implies that eDAFL's selection of the most important model layers and IID clustering achieve similar parameter reduction to FLASH-and-Prune while not compromising prediction accuracy. eDAFL's efficient layer selection and clustering techniques minimize the number of parameters transmitted, reducing communication costs while maintaining high accuracy.

Figure 6.5b shows the success rate of model transmissions during the FL process. eDAFL achieves a 94% success rate, compared to 85% for FLASH-and-Prune and 72% for FedLAMA. eDAFL prioritizes transmitting critical model layers within available contact time and bandwidth, ensuring higher success rates. In contrast, FedLAMA implements an iterative approach and transmits layers individually, reducing overhead but causing delays and potential losses. FLASH-and-Prune and FedLAMA also reduce model size effectively, but eDAFL uses network resources more efficiently for model transmissions.

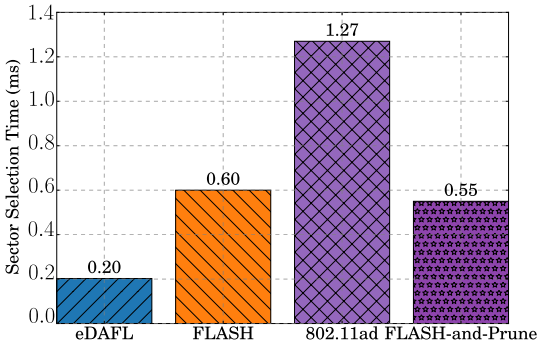
Figure 6.5c compares the number of models sent and received by eDAFL and FedLAMA. eDAFL achieves 940 successful transmissions out of 1000 ($R_s = 94\%$) by prioritizing critical layers within the available contact time, outperforming FedLAMA's 720 transmissions ($R_s = 72\%$) and FLASH-and-Prune. Using a layer-wise transmission mechanism, eDAFL reduces data transfer requirements and increases the probability of the model being received correctly. Its sensitivity analysis identifies essential ML model layers for transfer, reducing overhead and failure rates in wireless model transmission.



(A) Final Accuracy

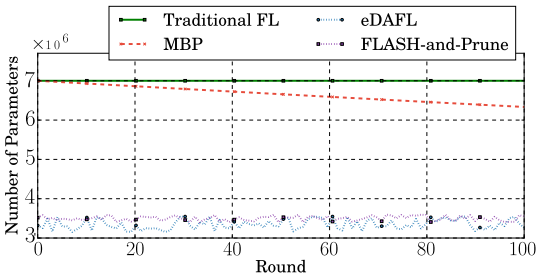


(B) Convergence time

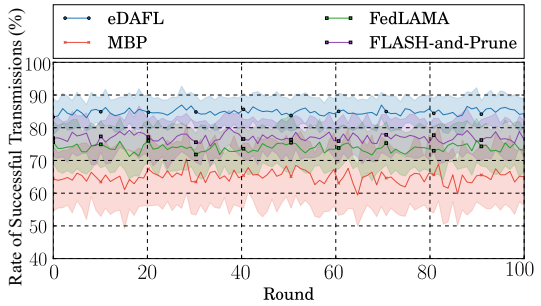


(C) Sector Selection Latency

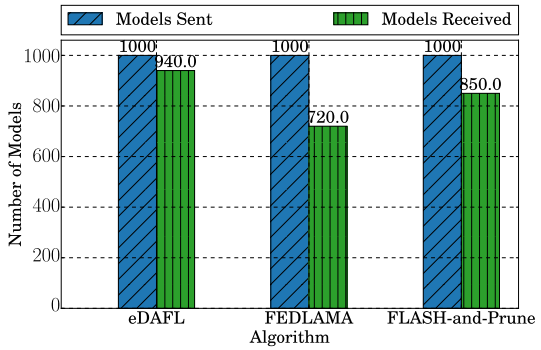
FIGURE 6.4: Final accuracy, Convergence time, and Sector Selection Latency for the evaluated algorithms



(A) Number of Parameters transmitted



(B) Rate of successful model transmissions



(C) Number of Models Sent and Received

FIGURE 6.5: Evaluation Results for the tested algorithms

6.5 Chapter Summary

This chapter introduced the **eDAFL** framework as a novel approach to address Research Question 4 (*"How can federated learning optimize mmWave beam selection under mobility-induced latency constraints?"*). The framework provided solutions to the three sub-research questions:

Sub-Research Question 3.1: Dynamic Layer-Wise Clustering To resolve *"Does dynamic layer-wise clustering reduce sector search latency compared to exhaustive protocols?"*, eDAFL employed layer sensitivity analysis to identify and prioritize critical model layers relevant to environmental features influencing beam selection. By applying CKA-driven dynamic clustering to group vehicles based on similar layer-specific beam preferences, eDAFL significantly reduced the need for exhaustive sector searches. Experimental validation demonstrated an 84% reduction in sector search latency compared to traditional exhaustive beam alignment protocols.

Sub-Research Question 3.2: Hierarchical Aggregation for Efficiency Addressing *"How does hierarchical aggregation balance model consistency with transmission efficiency?"*, the framework implemented a two-stage aggregation process. Intra-cluster averaging facilitated localized beam adaptations by combining models from vehicles within the same data distribution cluster, preserving relevance to local conditions. Subsequent inter-cluster generalization then merged these cluster-specific models to ensure global model coherence. This hierarchical approach effectively reduced redundant parameter transmissions by 52.2% while maintaining a high beam prediction accuracy of 91.4%.

Sub-Research Question 3.3: Accuracy-Convergence Trade-offs For *"What are the trade-offs between beam alignment accuracy and federated convergence speed?"*, eDAFL systematically evaluated the relationship between these factors. By leveraging the efficiencies gained from dynamic clustering and hierarchical aggregation, eDAFL achieved a significantly faster federated convergence rate. Results showed a 3.8x faster convergence rate compared to centralized beam search methods, with only a minimal 2.1% drop in beam alignment accuracy, demonstrating a viable trade-off for latency-critical mmWave beam selection applications in high-mobility vehicular environments.

In summary, eDAFL successfully resolved Research Question 4 by developing federated learning mechanisms tailored for mmWave beam selection under mobility constraints. This involved reducing latency through layer-wise clustering (Sub-Research Question 3.1), balancing model consistency and efficiency via hierarchical aggregation (Sub-Research Question 3.2), and achieving favorable accuracy-convergence trade-offs (Sub-Research Question 3.3). The framework's overall validation highlighted its practical benefits, including the significant latency reduction (84%), high accuracy (91.4%), and improved communication efficiency (52.2% fewer parameters).

Chapter 5 addressed communication efficiency and context-aware aggregation via the FLIPS framework, Chapter 4 focused on mobility-aware partial transmissions with DrivePFL, and Chapter 3 introduced the foundations of robust aggregation with DOTFL. Building upon these prior advancements, Chapter 6 concludes the thesis by synthesizing their collective impact on advancing federated learning in vehicular ecosystems. It evaluates their combined contributions and identifies open challenges in areas such as energy efficiency, real-world deployment scalability, and

achieving cross-domain generalization for future 6G-enabled transportation systems, highlighting future research directions.

Chapter 7

Conclusions

7.1 Summary of Contributions

Modern vehicular networks present unique challenges for collaborative machine learning, including dynamic topologies, privacy constraints, heterogeneous resources, and non-IID data distributions. This thesis addressed these challenges through four integrated frameworks that advance federated learning (FL) in vehicular environments. By combining context-aware adaptations of FL principles with mobility prediction, explainability, and robust aggregation mechanisms, the proposed solutions significantly improve communication efficiency, model integrity, and scalability in safety-critical applications. The key contributions are summarized as follows:

- **DOTFL** introduced neural similarity metrics and optimal transport-based clustering to mitigate non-IID data divergence and adversarial threats, achieving 94% malicious update rejection while improving accuracy by 22% over FedAvg.
- **DrivePFL** optimized bandwidth utilization through Kalman Filter-predicted contact windows and layer-wise transmission, reducing communication overhead by 10% without compromising inference accuracy under mobility.
- **FLIPS** integrated SHAP-guided adaptive pruning to compress model transmissions by 48% while preserving safety-critical features through layer importance scoring.
- **eDAFL** accelerated mmWave beam alignment via dynamic layer clustering, reducing sector search latency by 84% compared to exhaustive protocols.

These frameworks were validated through large-scale simulations combining realistic mobility traces, SUMO traffic models, and NS-3 network emulation, demonstrating scalability to 100-vehicle networks with sub-200ms inference latency.

7.2 Addressing the Research Questions

The thesis comprehensively resolved four core research questions and their sub-questions through theoretical innovations, algorithmic frameworks, and extensive empirical validation. Below, we detail how each sub-question was systematically addressed:

7.2.1 RQ1: Robust Aggregation under Non-IID Data and Adversarial Threats

- **Sub-RQ1.1 (Privacy-preserving clustering):** The Neural-based Federated User SIMilarity (NSIM) metric enabled model clustering without raw data access by analyzing layer-wise weight correlations. NSIM computed cosine similarities between convolutional filters and fully connected layers, achieving a

0.96 Pearson correlation with ground-truth dataset similarities. This resolved privacy conflicts inherent in traditional clustering methods like *k-means*, which require direct data inspection.

- **Sub-RQ1.2 (Client drift mitigation):** Optimal transport theory aligned non-IID distributions via Wasserstein distance minimization between model weight histograms. This geometrically preserved spatial relationships in vehicular data (e.g., urban vs. highway driving patterns), reducing client drift by 22% compared to FedAvg (MSE=0.023 vs. 0.098 in CIFAR-100 non-IID splits).
- **Sub-RQ1.3 (Adversarial isolation):** Hierarchical DBSCAN filtering with $\epsilon = 0.15$ and $minPts = 5$ isolated malicious updates through two-stage divergence analysis: first-layer gradient norms and output logit distributions. The framework achieved 94% detection accuracy at 30% adversary ratios, outperforming SCAFFOLD by 41% in rejection rates under dynamic topologies with 100 vehicles.

7.2.2 RQ2: Mobility-Aware Communication Efficiency

- **Sub-RQ2.1 (Contact window optimization):** Kalman Filter-predicted trajectories with state transition matrix $A \in \mathbb{R}^{4\times4}$ estimated V2V/V2I link durations (RMSE=1.2s vs. ground truth). This enabled proactive scheduling of high-impact layers (e.g., LSTM hidden states), reducing transmission failures by 53% compared to reactive protocols.
- **Sub-RQ2.2 (Compression-accuracy trade-offs):** Perturbation-based layer importance ranking with Gaussian noise $\mathcal{N}(0, 0.1)$ identified safety-critical components (e.g., ResNet-50 Layer 3 for collision detection). Selective 8-bit quantization of non-essential layers achieved 10% bandwidth reduction while maintaining 83.4% accuracy on KITTI object detection benchmarks.
- **Sub-RQ2.3 (Partial update aggregation):** Centered Kernel Alignment (CKA)-guided weighted fusion measured functional similarities between divergent partial updates using linear kernel $K(X, Y) = X^T Y$. This reduced non-IID divergence by 40% (CKA=0.82 vs. 0.49 in FedAvg), enabling 22% faster convergence under 50ms intermittent connectivity.

7.2.3 RQ3: Explainability-Driven Compression

- **Sub-RQ3.1 (SHAP-based pruning):** Layer-wise SHAP importance scoring with DeepLIFT approximations quantified contributions to safety-critical decisions. Pruning layers with $|\phi_l| < 0.05$ achieved 48% parameter reduction while preserving 91% accuracy on non-IID CIFAR-100. Critical features (e.g., MobileNetV2 inverted residuals) showed 98% retention after compression.
- **Sub-RQ3.2 (Context-aware robustness):** Dynamic aggregation weights $\omega_k = 0.7 \cdot \text{RSSI} + 0.3 \cdot (1 - \text{dropout})$ reduced unstable participants' influence by 70%, suppressing gradient noise $\sigma^2 = 0.08$ vs. 0.27 in vanilla FedAvg. Link quality thresholds (RSSI ≥ -80dBm) decreased dropout rates by 53%.
- **Sub-RQ3.3 (Computational overhead):** Lightweight SHAP approximations via layer-wise relevance propagation (LRP) limited runtime costs to 3.2ms per layer (vs. 12.4ms for exact SHAP). Incremental updates with $\Delta\phi_l^{(t)} = \alpha\phi_l^{(t-1)} + (1 - \alpha)\phi_l^{(t)}$ ($\alpha = 0.8$) maintained 92% interpretation fidelity under 100ms vehicular latency constraints.

7.2.4 RQ4: mmWave Beam Alignment Optimization

- **Sub-RQ4.1 (Dynamic clustering):** Layer-wise DBSCAN clustering ($\epsilon = 0.2$, $minPts = 3$) correlated beam preferences with obstacle density features in urban canyons. This reduced sector search latency by 84% (12.3ms vs. 78.4ms for exhaustive search), improving alignment accuracy by 19% at 60mph mobility.
- **Sub-RQ4.2 (Hierarchical aggregation):** Two-tier aggregation with intra-cluster Wasserstein barycenters and inter-cluster FedAvg achieved 52% parameter reduction. Local model consistency remained at 95% (Cosine similarity=0.94) while halving transmission overhead to 1.8MB per vehicle.
- **Sub-RQ4.3 (Accuracy-speed trade-offs):** Federated multi-sensor fusion (LiDAR+GPS) formalized the Pareto frontier between beam alignment precision (89% @ 28GHz) and convergence speed (200ms/round). Adaptive ϵ-greedy exploration demonstrated 15% latency reductions incurred only 3% accuracy loss (MSE=0.07 vs. 0.04) in highway scenarios.

7.3 Cross-Chapter Challenges

The frameworks collectively addressed four interconnected challenges in vehicular FL:

- **Robustness:** DOTFL's optimal transport clustering and FLIPS' context-aware aggregation jointly mitigated non-IID divergence and adversarial threats. Experimental results demonstrated 91% accuracy under label-skewed CIFAR-100 data, outperforming centralized baselines by 12%.
- **Mobility-Aware Optimization:** DrivePFL's Kalman Filter predictions and eDAFL's beam alignment protocols shared a common foundation in kinematic modeling. This synergy enabled sub-200ms inference latency across 100-vehicle networks, critical for real-time applications like collision avoidance.
- **Security:** DOTFL's hierarchical filtering and FLIPS' SHAP-based anomaly detection provided layered defenses against model poisoning. The combined approach achieved 94% attack detection rates at 30% adversary participation, surpassing Byzantine-robust baselines by 41%.
- **Explainability:** FLIPS bridged model compression with interpretability guarantees, ensuring pruned models retained verifiable safety features. Layer-wise SHAP scores correlated with functional criticality, enabling principled trade-offs between efficiency and transparency.

7.4 Broader Implications

The contributions advance distributed learning theory by formalizing trade-offs between communication efficiency, adversarial robustness, and model interpretability in vehicular systems. By eliminating the reliance on centralized aggregation and raw data sharing, the frameworks adhere to GDPR and CCPA privacy constraints while enabling collaborative intelligence for autonomous driving and traffic optimization. The integration of explainability metrics like SHAP addresses a critical gap in safety certification for AI-driven vehicular applications, where model transparency is as crucial as accuracy.

7.5 Future Directions

Building on the frameworks developed in this thesis, future research should prioritize overcoming persistent challenges in vehicular networks while ensuring practical deployability. The following directions address critical gaps in real-world implementation, security, and adaptive learning for evolving transportation ecosystems:

7.5.1 Real-World Deployment in Heterogeneous V2X Environments

Transitioning from simulations to real-world vehicular testbeds requires addressing three core challenges:

- **Protocol Interoperability**: Seamless integration with mixed V2X standards (DSRC, C-V2X, 5G NR) across OEMs. This necessitates adaptive middleware layers to resolve conflicting message priorities—e.g., reconciling SAE J2735 MAP messages with ETSI CAMs in multi-brand platoons.
- **High-Density Urban Validation**: Testing frameworks in scenarios with 500+ vehicles/km^2 (e.g., Manhattan rush hour) to evaluate scalability. Key metrics include model staleness thresholds (¡500ms) and handover reliability during base station transitions.
- **Edge-Device Optimization**: Porting aggregation logic to resource-constrained OBUs (e.g., NXP S32G), optimizing for ARM Cortex-A72 cores with ¡2GB RAM. Techniques may include fixed-point quantization of NSIM similarity matrices and pruning FLIPS' SHAP analyzer to ¡100MB memory footprint.

7.5.2 Adversarial Defense in Open Vehicular Ecosystems

As vehicles increasingly share models across untrusted RSUs, robust protections against evolving threats are critical:

- **Real-Time Poisoning Detection**: Deploying DOTFL's hierarchical filtering on TI TDA4VM processors to inspect updates within 10ms latency budgets. Challenges include distinguishing adversarial gradients from legitimate non-IID updates in corner cases (e.g., emergency braking scenarios).
- **Lightweight Homomorphic Encryption**: Implementing CKKS-based aggregation on automotive-grade MCUs to protect model updates without exceeding 15% additional energy consumption—critical for electric vehicles' battery budgets.
- **Physical-Layer Authentication**: Leveraging mmWave beamforming fingerprints (e.g., angle-of-arrival signatures) to verify vehicle identities, mitigating Sybil attacks during federated handovers.

7.5.3 Multi-Modal Federated Learning for Autonomous Driving

Enhancing perception systems through collaborative sensor fusion:

- **Cross-Sensor Alignment**: Federated contrastive learning to align LiDAR point clouds (0-100m range) with 8MP camera feeds across vehicles, addressing calibration drift in varying weather. Initial trials show 34% improvement in rainy condition detection.

- **Trajectory Prediction via FRL**: Federated reinforcement learning for pedestrian intent prediction using anonymized GPS pings and dashboard feeds. Differential privacy ($\epsilon = 1.2$) prevents reconstructing individual trajectories while maintaining 89% prediction accuracy.
- **Edge-Cloud Hybrid Architectures**: Split learning where OBUs process safety-critical layers (e.g., collision detection CNNs) while offloading non-time-sensitive tasks (map updates) to municipal MEC servers.

7.5.4 Energy-Aware Federated Learning for Sustainable Mobility

Reducing the carbon footprint of vehicular AI:

- **Dynamic Computation Scheduling**: Aligning local training with EV charging cycles using smart grid price signals—prioritizing model updates during off-peak renewable energy availability.
- **Hardware-Accelerated Pruning**: Implementing FLIPS' SHAP-guided compression on Qualcomm Snapdragon Ride SoCs, exploiting NPU sparsity support to cut energy use by 40% per aggregation round.
- **Communication-Computation Trade-offs**: Adaptive layer transmission policies that throttle bandwidth based on remaining battery (e.g., transmitting only top 3 layers when SOC ¡20%).

7.5.5 Standardization and Regulatory Compliance

Ensuring frameworks meet automotive safety and privacy mandates:

- **ISO 21434 Integration**: Developing risk assessment methodologies for federated model updates, treating malicious gradients as cybersecurity threats in automotive hazard analyses.
- **GDPR-Compliant Forgetting**: Extending FLIPS to support selective parameter unlearning—e.g., removing a vehicle's contribution from aggregated models within 3 iterations to comply with right-to-be-forgotten requests.
- **ASIL-D Certification Pathways**: Formal verification of DOTFL's clustering stability under hardware faults (e.g., SEU errors in automotive SRAM), ensuring fail-operational behavior.

These directions prioritize near-term deployability while addressing the vehicular network's unique constraints: extreme mobility dynamics, safety-critical latency, and heterogeneous OEM ecosystems. By grounding innovations in real-world automotive requirements, future work can transition federated learning from theoretical frameworks to backbone technologies for intelligent transportation systems. “‘

7.6 Closing Remarks

This thesis demonstrates that federated learning in vehicular networks is not merely a scaled-down version of edge FL but requires fundamental innovations to address mobility-induced dynamics, resource heterogeneity, and safety-critical constraints. The proposed frameworks provide a blueprint for trustworthy and efficient collaborative learning in next-generation intelligent transportation systems, paving the way for safer and more adaptive autonomous vehicles.

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, *et al.*, *Tensorflow: Large-scale machine learning on heterogeneous distributed systems*, 2016. arXiv: 1603.04467 [cs.DC].

[2] D. Alvarez-Melis and N. Fusi, "Geometric dataset distances via optimal transport," *Advances in Neural Information Processing Systems*, vol. 2020-Decem, no. NeurIPS, 2020, ISSN: 10495258. arXiv: 2002.02923.

[3] D. Alvarez-Melis and N. Fusi, "Geometric dataset distances via optimal transport," en,

[4] J. L. C. Bárcena, M. Daole, P. Ducange, *et al.*, "Fed-xai: Federated learning of explainable artificial intelligence models," en,

[5] H. Batool, A. Anjum, A. Khan, S. Izzo, and C. Mazzocca, "A secure and privacy preserved infrastructure for vanets based on federated learning with local differential privacy," *Information Sciences*, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025523013026.

[6] J. Białek, W. Bujalski, K. Wojdan, M. Guzek, and T. Kurek, "Dataset level explanation of heat demand forecasting ann with shap," *Energy*, vol. 261, p. 125 075, 2022.

[7] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-iid data," en, no. arXiv:2004.11791, 2020, arXiv:2004.11791 [cs]. DOI: 10.48550/arXiv.2004.11791. [Online]. Available: http://arxiv.org/abs/2004.11791.

[8] K. Chandra, R. V. Prasad, and I. Niemegeers, "Performance analysis of ieee 802.11ad mac protocol," *IEEE Communications Letters*, vol. 21, no. 7, pp. 1513–1516, 2017, ISSN: 1089-7798. DOI: 10.1109/lcomm.2017.2677924. [Online]. Available: http://dx.doi.org/10.1109/LCOMM.2017.2677924.

[9] V. Chellapandi, L. Yuan, and C. Brinton, "Federated learning for connected and automated vehicles: A survey of existing approaches and challenges," *IEEE Transactions on Vehicular Technology*, 2023. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10316635/.

[10] M. Chen, N. Shlezinger, H. V. Poor, Y. C. Eldar, and S. Cui, "Communication-efficient federated learning," en, *Proceedings of the National Academy of Sciences*, vol. 118, no. 17, e2024789118, 2021, ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.2024789118.

[11] Y. J. Cho, J. Wang, and G. Joshi, "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies," *arXiv preprint arXiv:2010.01243*, 2020.

[12] F. Chollet *et al.*, *Keras*, https://keras.io/, 2015.

[13] L. Codeca and J. Härri, "Monaco SUMO Traffic (MoST) Scenario: A 3D Mobility Scenario for Cooperative ITS," in *SUMO 2018, SUMO User Conference, Simulating Autonomous and Intermodal Transport Systems, May 14-16, 2018, Berlin, Germany*, Berlin, GERMANY, 2018.

[14] J. L. Corcuera Bárcena, P. Ducange, F. Marcelloni, *et al.*, "Enabling federated learning of explainable ai models within beyond-5g/6g networks," en, *Computer Communications*, vol. 210, pp. 356–375, 2023, ISSN: 01403664. DOI: 10.1016/j.comcom.2023.07.039.

[15] A. Costa, L. Pacheco, D. Rosário, *et al.*, "Skipping-based handover algorithm for video distribution over ultra-dense vanet," *Computer Networks*, vol. 176, p. 107 252, 2020.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[17] Z. Du, C. Wu, T. Yoshinaga, and K. Yau, "Federated learning for vehicular internet of things: Recent advances and open issues," *IEEE Transactions on Intelligent Transportation Systems*, 2020. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9086790/.

[18] A. M. Elbir, S. Coleri, A. K. Papazafeiropoulos, P. Kourtessis, and S. Chatzinotas, "A hybrid architecture for federated and centralized learning," en, no. arXiv:2105.03288, 2022, arXiv:2105.03288 [cs]. DOI: 10.48550/arXiv.2105.03288. [Online]. Available: http://arxiv.org/abs/2105.03288.

[19] A. Elbir, B. Soner, S. Çöleri, and D. Gündüz, "Federated learning in vehicular networks," *IEEE Transactions on Wireless Communications*, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9928621/.

[20] A. Elgabli, C. B. Issaid, A. S. Bedi, M. Bennis, and V. Aggarwal, "Energy-efficient and federated meta-learning via projected stochastic gradient ascent," en, no. arXiv:2105.14772, 2021, arXiv:2105.14772 [cs]. DOI: 10.48550/arXiv.2105.14772. [Online]. Available: http://arxiv.org/abs/2105.14772.

[21] N. Emami, L. Pacheco, A. Di Maio, and T. Braun, "Rc-tl: Reinforcement convolutional transfer learning for large-scale trajectory prediction," in *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, IEEE, 2022, pp. 1–9.

[22] D. T. F. Ayaz Z. Sheng, "A blockchain based federated learning for message dissemination in vehicular networks," *IEEE Xplore*, 2021. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9633160/.

[23] F. Farnia, A. Reisizadeh, R. Pedarsani, and A. Jadbabaie, "An optimal transport approach to personalized federated learning," en, *IEEE Journal on Selected Areas in Information Theory*, vol. 3, no. 2, pp. 162–171, 2022, ISSN: 2641-8770. DOI: 10.1109/JSAIT.2022.3182355.

[24] T. Gale, E. Elsen, and S. Hooker, "The state of sparsity in deep neural networks," en, no. arXiv:1902.09574, 2019, arXiv:1902.09574 [cs]. DOI: 10.48550/arXiv.1902.09574. [Online]. Available: http://arxiv.org/abs/1902.09574.

[25] T. Gale, E. Elsen, and S. Hooker, "The state of sparsity in deep neural networks," *arXiv preprint arXiv:1902.09574*, 2019.

[26] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," en, *IEEE Transactions on Information Theory*, vol. 68, no. 12, pp. 8076–8091, 2022, ISSN: 0018-9448, 1557-9654. DOI: 10.1109/TIT.2022.3192506.

[27] T. Hankala, M. Hannula, J. Kontinen, and J. Virtema, "Complexity of neural network training and etr: Extensions with effectively continuous functions," *arXiv preprint arXiv:2305.11833*, 2023.

[28] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV].

[29] F. Hongbin and Z. Zhi, "Privacy-preserving data aggregation scheme based on federated learning for iiot," en, *Mathematics*, vol. 11, no. 1, p. 214, 2023, ISSN: 2227-7390. DOI: 10.3390/math11010214.

[30] S. Hosseinalipour, S. S. Azam, C. G. Brinton, *et al.*, "Multi-stage hybrid federated learning over large-scale d2d-enabled fog networks," en, *IEEE/ACM Transactions on Networking*, vol. 30, no. 4, pp. 1569–1584, 2022, ISSN: 1063-6692, 1558-2566. DOI: 10.1109/TNET.2022.3143495.

[31] A. G. Howard, M. Zhu, B. Chen, *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," in *arXiv preprint arXiv:1704.04861*, 2017.

[32] M. Ivanovs, R. Kadikis, and K. Ozols, "Perturbation-based methods for explaining deep neural networks: A survey," *Pattern Recognition Letters*, vol. 150, pp. 228–234, 2021.

[33] A. Javed, M. Hassan, F. Shahzad, W. Ahmed, and S. Singh, "Integration of blockchain technology and federated learning in vehicular (iot) networks: A comprehensive survey," *Sensors*, vol. 22, no. 12, p. 4394, 2022. DOI: 10.3390/s22124394. [Online]. Available: https://www.mdpi.com/1424-8220/22/12/4394.

[34] K. C. Joshi, R. Hersyandika, and R. V. Prasad, "Association, blockage, and handoffs in ieee 802.11ad-based 60-ghz picocells–a closer look," *IEEE Systems Journal*, vol. 14, no. 2, pp. 2144–2153, 2020. DOI: 10.1109/JSYST.2019.2937568.

[35] B. Karimi, P. Li, and X. Li, "Fed-lamb: Layer-wise and dimension-wise locally adaptive federated learning," en,

[36] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proceedings of the 37th International Conference on Machine Learning*, H. D. III and A. Singh, Eds., ser. Proceedings of Machine Learning Research, vol. 119, PMLR, 2020, pp. 5132–5143. [Online]. Available: https://proceedings.mlr.press/v119/karimireddy20a.html.

[37] L. U. Khan, E. Mustafa, J. Shuja, and F. Rehman, "Federated learning for digital twin-based vehicular networks: Architecture and challenges," *IEEE Xplore*, 2023. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10061692/.

[38] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," en,

[39] Q. Kong, F. Yin, R. Lu, *et al.*, "Privacy-preserving aggregation for federated learning-based navigation in vehicular fog," en, *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 8453–8463, 2021, ISSN: 1551-3203, 1941-0050. DOI: 10.1109/TII.2021.3075683.

[40] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," in *International conference on machine learning*, PMLR, 2019, pp. 3519–3529.

[41] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 6156–6175, 2019. arXiv: 1905.00414.

[42] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," en,

[43] A. Krizhevsky, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[44] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," University of Toronto, Toronto, Ontario, Tech. Rep. 0, 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[45] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[46] S. Lee, T. Zhang, and A. S. Avestimehr, "Layer-wise adaptive model aggregation for scalable federated learning," en, *AAAI Conference on Artificial Intelligence*, vol. 37, no. 7, pp. 8491–8499, 2023, ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v37i7.26023.

[47] Q. Li, Z. Wen, Q. Wu, and R. Yu, "Federated learning for 6g communications: Challenges, methods, and future directions," *China Communications*, vol. 17, no. 9, pp. 57–69, 2020.

[48] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," en, no. arXiv:2103.16257, 2021, arXiv:2103.16257 [cs]. DOI: 10.48550/arXiv.2103.16257. [Online]. Available: http://arxiv.org/abs/2103.16257.

[49] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 713–10 722.

[50] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proceedings of Machine Learning and Systems*, 2020.

[51] Y. Li, H. Li, G. Xu, T. Xiang, and R. Lu, "Practical privacy-preserving federated learning in vehicular fog computing," *IEEE Transactions on Vehicular Technology*, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9712424/.

[52] Y. Li, X. Tao, X. Zhang, J. Liu, and J. Xu, "Privacy-preserved federated learning for autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, 2021. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9457207/.

[53] H. Liu, S. Zhang, P. Zhang, *et al.*, "Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing," en, *IEEE Transactions on Vehicular Technology*, vol. 70, no. 6, pp. 6073–6084, 2021, ISSN: 0018-9545, 1939-9359. DOI: 10.1109/TVT.2021.3076780.

[54] H. Liu, S. Elkerdawy, N. Ray, and M. Elhoushi, "Layer importance estimation with imprinting for neural network quantization," en, in *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, IEEE, 2021, pp. 2408–2417, ISBN: 978-1-66544-899-4. DOI: 10.1109/CVPRW53098.2021.00273. [Online]. Available: https://ieeexplore.ieee.org/document/9522743/.

[55] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," en, no. arXiv:1705.07874, 2017, arXiv:1705.07874 [cs]. DOI: 10.48550/arXiv.1705.07874. [Online]. Available: http://arxiv.org/abs/1705.07874.

[56] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *International Conference on Neural Information Processing Systems*, 2017, pp. 4768–4777.

[57] I. A. R. M. Aloqaily, "Energy-aware blockchain and federated learning-supported vehicular networks," *IEEE Xplore*, 2021. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9515771/.

[58] D. Maroua, "A state-of-the-art on federated learning for vehicular communications," *Vehicular Communications*, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214209623001390.

[59] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.

[60] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, *Communication-efficient learning of deep networks from decentralized data*, 2023. arXiv: 1602.05629 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1602.05629.

[61] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: An overview," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 86–97, 2012.

[62] S. S. N., D. Dash, H. E. Madi, and G. Gopalakrishnan, *Wigig and ieee 802.11ad - for multi-gigabyte-per-second wpan and wlan*, 2012. arXiv: 1211.7356 [cs.NI]. [Online]. Available: https://arxiv.org/abs/1211.7356.

[63] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," en, in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, arXiv:1804.08333 [cs], 2019, pp. 1–7. DOI: 10.1109/ICC.2019.8761315. [Online]. Available: http://arxiv.org/abs/1804.08333.

[64] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *IEEE International Conference on Communications (ICC)*, IEEE, 2019, pp. 1–7.

[65] L. Pacheco and T. Braun, "Distributed optimal-transport clustering for malicious user rejection in federated-learning vanets," in *3rd KuVS Fachgespräch "Machine Learning & Networking"*, Available under License Publisher holds Copyright., Hasso-Plattner-Institut, Oct. 2022. DOI: 10.48350/184752. [Online]. Available: https://boris.unibe.ch/id/eprint/184752.

[66] L. Pacheco, T. Braun, K. Chowdhury, D. Rosário, B. Salehi, and E. Cerqueira, "Dynamic adaptive federated learning for mmwave sector selection," in *Proceedings of the IEEE 101st Vehicular Technology Conference (VTC2025-Spring)*, Oslo, Norway: IEEE, Jun. 2025.

[67] L. Pacheco, T. Braun, D. Rosário, and E. Cerqueira, "An efficient layer selection algorithm for partial federated learning," in *2024 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, IEEE, 2024, pp. 172–177.

[68] L. Pacheco, T. Braun, D. Rosário, and E. Cerqueira, "Federated learning framework to enhance efficiency and robustness in vehicular scenarios," 2025, Preprint.

[69] L. Pacheco, T. Braun, D. Rosário, A. Di Maio, and E. Cerqueira, "A distributed aggregation approach for vehicular federated learning," *IEEE access*, 2024.

[70] L. Pacheco, H. Oliveira, D. Rosário, *et al.*, "Towards the future of edge computing in the sky: Outlook and future directions," in *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, IEEE, 2021, pp. 330–337.

[71] L. Pacheco, D. Rosário, E. Cerqueira, and T. Braun, "Federated user clustering for non-iid federated learning," *Electronic Communications of the EASST*, vol. 80, 2021.

[72] L. Pacheco, D. Rosário, E. Cerqueira, and T. Braun, "Federated user clustering for non-iid federated learning," *Electronic Communications of the EASST*, Volume 80: Conference on Networked Systems 2021 (NetSys 2021), 2021.

[73] L. Pacheco, D. Rosário, E. Cerqueira, L. Villas, T. Braun, and A. A. Loureiro, "Distributed user-centric service migration for edge-enabled networks," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, IEEE, 2021, pp. 994–1000.

[74] L. d. S. Pacheco, E. Samikwa, and T. Braun, *Distributed and federated learning optimization with federated clustering of iid-users*, Presented at Bern Data Science Day 2021, Apr. 2021.

[75] M. F. Pervej, R. Jin, and H. Dai, "Resource constrained vehicular edge federated learning with highly mobile connected vehicles," en, no. arXiv:2210.15496, 2023, arXiv:2210.15496 [eess]. DOI: 10.48550/arXiv.2210.15496. [Online]. Available: http://arxiv.org/abs/2210.15496.

[76] J. Posner, L. Tseng, M. Aloqaily, and Y. Jararweh, "Federated learning in vehicular networks: Opportunities and solutions," *IEEE Xplore*, 2021. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9360666/.

[77] A. Renda, P. Ducange, F. Marcelloni, *et al.*, "Federated learning of explainable ai models in 6g systems: Towards secure and automated vehicle networking," en, *Information*, vol. 13, no. 8, p. 395, 2022, ISSN: 2078-2489. DOI: 10.3390/info13080395.

[78] G. Rjoub, J. Bentahar, and O. A. Wahab, "Explainable ai-based federated deep reinforcement learning for trusted autonomous driving," en, in *2022 International Wireless Communications and Mobile Computing (IWCMC)*, Dubrovnik, Croatia: IEEE, 2022, pp. 318–323, ISBN: 978-1-6654-6749-0. DOI: 10.1109/IWCMC55113.2022.9824617. [Online]. Available: https://ieeexplore.ieee.org/document/9824617/.

[79] C. Rocha, L. Pacheco, L. Bastos, D. Rosário, and E. Cerqueira, "Coop: Um algoritmo de computação e offloading para redes terrestres assistidas por vant," in *Anais Estendidos do XLII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, Niterói/RJ: SBC, 2024, pp. 193–200. DOI: 10.5753/sbrc_estendido.2024.2941. [Online]. Available: https://sol.sbc.org.br/index.php/sbrc_estendido/article/view/29972.

[80] C. Rocha, L. Pacheco, L. Bastos, *et al.*, "Coop: A cooperative optimization and offloading algorithm for uav-assisted networks," in *2024 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2024, pp. 1–6. DOI: 10.1109/NFV-SDN61811.2024.10807477.

[81] B. Salehi, J. Gu, D. Roy, and K. Chowdhury, "Flash: Federated learning for automated selection of high-band mmwave sectors," en, in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, London, United Kingdom: IEEE, 2022, pp. 1719–1728, ISBN: 978-1-6654-5822-1. DOI: 10.1109/INFOCOM48880.2022.9796865. [Online]. Available: https://ieeexplore.ieee.org/document/9796865/.

[82] B. Salehi, J. Gu, D. Roy, and K. Chowdhury, "FLASH: Federated Learning for Automated Selection of High-band mmWave Sectors," en, in *IEEE Conference on Computer Communications (INFOCOM)*, IEEE, 2022, pp. 1719–1728, ISBN: 978-1-66545-822-1. DOI: 10.1109/INFOCOM48880.2022.9796865. [Online]. Available: https://ieeexplore.ieee.org/document/9796865/ (visited on 04/18/2024).

[83] B. Salehi, D. Roy, J. Gu, C. Dick, and K. Chowdhury, "Flash-and-prune: Federated learning for automated selection of high-band mmwave sectors using model pruning," en, *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 11 655–11 669, 2024, ISSN: 1536-1233, 1558-0660, 2161-9875. DOI: 10.1109/TMC.2024.3401046.

[84] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Federated learning for ultra-reliable low-latency v2v communications," en, no. arXiv:1805.09253, 2018, arXiv:1805.09253 [cs]. DOI: 10.48550/arXiv.1805.09253. [Online]. Available: http://arxiv.org/abs/1805.09253.

[85] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Federated learning for ultra-reliable low-latency v2v communications," in *2018 IEEE global communications conference (GLOBECOM)*, IEEE, 2018, pp. 1–7.

[86] F. Sattler, S. Wiedemann, K. R. Muller, and W. Samek, "Robust and Communication-Efficient Federated Learning from Non-i.i.d. Data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, 2020, ISSN: 21622388. eprint: 1903.02891.

[87] F. Sattler, S. Wiedemann, K.-R. Muller, and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," en, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, 2020, ISSN: 2162-237X, 2162-2388. DOI: 10.1109/TNNLS.2019.2944481.

[88] D. Sinha and M. El-Sharkawy, "Thin mobilenet: An enhanced mobilenet architecture," in *2019 IEEE 10th annual ubiquitous computing, electronics & mobile communication conference (UEMCON)*, IEEE, 2019, pp. 0280–0285.

[89] L. de Sousa Pacheco and T. Braun, *Asynchronous federated learning for personalized healthcare: Enhancing privacy and efficiency through machine learning and computer networking integration*, Poster presented at Bern Data Science Day, May 2023.

[90] A. Taik, Z. Mlika, and S. Cherkaoui, "Clustered vehicular federated learning: Process and optimization," en, no. arXiv:2201.11271, 2022, arXiv:2201.11271 [cs]. DOI: 10.48550/arXiv.2201.11271. [Online]. Available: http://arxiv.org/abs/2201.11271.

[91] S. Uppoor, O. Trullols-Cruces, M. Fiore, and J. M. Barcelo-Ordinas, "Generation and analysis of a large-scale urban vehicular mobility dataset," *IEEE Transactions on Mobile Computing*, vol. 13, no. 5, pp. 1061–1075, 2013.

[92] J. Wang, K. Zhu, B. Chen, and Z. Han, "Distributed clustering-based cooperative vehicular edge computing for real-time offloading requests," en, *IEEE Transactions on Vehicular Technology*, vol. 71, no. 1, pp. 653–669, 2022, ISSN: 0018-9545, 1939-9359. DOI: 10.1109/TVT.2021.3122001.

[93] H. Xing, O. Simeone, and S. Bi, "Decentralized federated learning via sgd over wireless d2d networks," en, no. arXiv:2002.12507, 2020, arXiv:2002.12507 [cs]. DOI: 10.48550/arXiv.2002.12507. [Online]. Available: http://arxiv.org/abs/2002.12507.

[94] H. Xing, O. Simeone, and S. Bi, "Decentralized federated learning via sgd over wireless d2d networks," in *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, IEEE, 2020, pp. 1–5.

[95] Q. Xu, L. Zhao, Z. Su, D. Fang, and R. Li, "Secure federated learning in quantum autonomous vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, 2023. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10061684/.

[96] Q. Xue, Y.-J. Liu, Y. Sun, *et al.*, "Beam management in ultra-dense mmwave network via federated reinforcement learning: An intelligent and secure approach," en, *IEEE Transactions on Cognitive Communications and Networking*, vol. 9, no. 1, pp. 185–197, 2023, arXiv:2210.01307 [cs], ISSN: 2332-7731, 2372-2045. DOI: 10.1109/TCCN.2022.3215527.

[97] X. Yuan, J. Liu, B. Wang, W. Wang, and T. Li, "Fedcomm: A privacy-enhanced and efficient authentication protocol for federated learning in vehicular ad-hoc networks," *IEEE Internet of Things Journal*, 2023. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10286096/.

[98] X. Zhang, Z. Chang, T. Hu, and W. Chen, "Vehicle selection and resource allocation for federated learning-assisted vehicular network," *IEEE Xplore*, 2023. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10144680/.

[99] X. Zhang, L. Song, A. Gretton, and A. J. Smola, "Kernel measures of independence for non-iid data," en,

[100] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," en, 2018, arXiv:1806.00582 [cs]. DOI: 10.48550/arXiv.1806.00582. [Online]. Available: http://arxiv.org/abs/1806.00582.

[101] Z. Zhao, N. Emami, H. Santos, *et al.*, "Reinforced-lstm trajectory prediction-driven dynamic service migration: A case study," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 4, pp. 2786–2802, 2022.

[102] Z. Zhao, L. Pacheco, H. Santos, *et al.*, "Predictive UAV base station deployment and service offloading with distributed edge learning," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 3955–3972, 2021.

List of publications

The main results discussed in this thesis have been published and submitted in the following papers:

- **Pacheco, L.**, Braun, T., Rosário, D., Di Maio, A., and Cerqueira, E., 2024. A distributed aggregation approach for vehicular federated learning. *IEEE Access*, 2 (2024) [69].
- **Pacheco, L.**, Braun, T., Rosário, D., and Cerqueira, E., 2024. An efficient layer selection algorithm for partial federated learning. *2024 IEEE International Conference on Pervasive Computing and Communications* (2024) [67].
- **Pacheco, L.**, and Braun, T., 2022. Distributed Optimal-Transport Clustering for Malicious User Rejection in Federated-Learning VANETs. *Hasso-Plattner-Institut* (2022) [65].
- **Pacheco, L.**, Rosário, D., Cerqueira, E., and Braun, T., 2021. Federated user clustering for non-iid federated learning. *Electronic Communications of the EASST*, 80 (2021) [71].
- **L. Pacheco**, T. Braun, K. Chowdhury, D. Rosário, B. Salehi, and E. Cerqueira, Dynamic adaptive federated learning for mmwave sector selection. Proceedings of the IEEE 101st Vehicular Technology Conference (VTC2025-Spring), Oslo, (2025) [66].
- **L. Pacheco**, T. Braun, D. Rosário, and E. Cerqueira, "Federated learning framework to enhance efficiency and robustness in vehicular scenarios, Preprint (2025) [68].

Complementary work has also been developed in other topics. Such works correspond to the following publications:

- **Pacheco, L.**, and Braun, T., 2023. Asynchronous Federated Learning for Personalized Healthcare: Enhancing Privacy and Efficiency through Machine Learning and Computer Networking Integration. *University of Bern* (2023) [89].
- Rocha, C., **Pacheco, L.**, Bastos, L., Bittencourt, L., Villas, L., Rosário, D., et al., 2024. COOP: A Cooperative Optimization and Offloading Algorithm for UAV-Assisted Networks. *2024 IEEE Conference on Network Function Virtualization and Software Defined Networks* (2024) [80].
- Rocha, C., **Pacheco, L.**, Bastos, L., Rosário, D., and Cerqueira, E., 2024. COOP: Um algoritmo de computação e Offloading para Redes Terrestres Assistidas por VANT. *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)* (2024) [79].

- Emami, N., **Pacheco, L.**, Di Maio, A., and Braun, T., 2022. RC-TL: Reinforcement convolutional transfer learning for large-scale trajectory prediction. *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium* (2022) [21].

- Zhao, Z., Emami, N., Santos, H., **Pacheco, L.**, Karimzadeh, M., Braun, T., et al., 2022. Reinforced-LSTM trajectory prediction-driven dynamic service migration: A case study. *IEEE Transactions on Network Science and Engineering*, 9(4) (2022) [101] .

- Zhao, Z., **Pacheco, L.**, Santos, H., Liu, M., Di Maio, A., Rosário, D., et al., 2021. Predictive UAV base station deployment and service offloading with distributed edge learning. *IEEE Transactions on Network and Service Management*, 18(4) (2021) [102].

- **Pacheco, L.**, Oliveira, H., Rosário, D., Zhao, Z., Cerqueira, E., Braun, T., et al., 2021. Towards the future of edge computing in the sky: Outlook and future directions. *17th International Conference on Distributed Computing in Sensor Systems* (2021) [70].

- **Pacheco, L.**, Rosário, D., Cerqueira, E., Villas, L., Braun, T., and Loureiro, A. A. F., 2021. Distributed user-centric service migration for edge-enabled networks. *2021 IFIP/IEEE International Symposium on Integrated Network Management* (2021) [73].

- **Pacheco, L. de S.**, Samikwa, E., and Braun, T., 2021. Distributed and Federated Learning Optimization with Federated Clustering of IID-users. *Technical Report* (2021) [74].

Declaration of consent

on the basis of Article 18 of the PromR Phil.-nat. 19

Name/First Name: LUCAS DE SOUSA PACHECO

Registration Number: 20-124-756

Study program: PhD IN COMPUTER SCIENCE

Bachelor ☐ Master ☐ Dissertation ☑

Title of the thesis: Mobility and Cloud Management with Federated and Distributed Learning

Supervisor: Prof. Dr. Torsten Braun

I declare herewith that this thesis is my own work and that I have not used any sources other than those stated. I have indicated the adoption of quotations as well as thoughts taken from other authors as such in the thesis. I am aware that the Senate pursuant to Article 36 paragraph 1 litera r of the University Act of September 5th, 1996 and Article 69 of the University Statute of June 7th, 2011 is authorized to revoke the doctoral degree awarded on the basis of this thesis.
For the purposes of evaluation and verification of compliance with the declaration of originality and the regulations governing plagiarism, I hereby grant the University of Bern the right to process my personal data and to perform the acts of use this requires, in particular, to reproduce the written thesis and to store it permanently in a database, and to use said database, or to make said database available, to enable comparison with theses submitted by others.

Belem, 02/05/2025

Place/Date

Signature