

# Janus: Dual-Sided Communication and Privacy Optimization for Federated Learning in Autonomous Vehicles

Himani Tyagi<sup>1</sup>, Abhilasha Singh<sup>2\*</sup>, Ganesh Gupta<sup>3</sup>

<sup>1,2</sup>Department of Computer Science and Engineering, SRM Institute of Science and Technology, Delhi-NCR Campus, Ghaziabad, India.

<sup>3</sup>Department of Computer Science and Engineering, Amity University, Gwalior, Madhya Pradesh, India.

**E-mail:** <sup>1</sup>ht3023@srmist.edu.in, <sup>2</sup>abhilass1@srmist.edu.in, <sup>3</sup>ggupta@gwa.amity.edu

**Orcid ID:** <sup>1</sup>0009-0004-6020-9874, <sup>2</sup>0000-0002-9482-3141, <sup>3</sup>0000-0001-9623-292X

## Abstract

Federated Learning (FL) enables collaborative model training across autonomous vehicles (AVs) without sharing raw driving data, but excessive uplink/downlink communication over bandwidth-constrained vehicular-to-everything links and privacy leakage through gradient inversion attacks limit its deployment. This paper proposes Janus, a unified FL framework combining adaptive Top-K gradient compression with error feedback on the uplink, genuine additive homomorphic encryption (HE, not a noise-based proxy) for secure aggregation, and compressed global-model broadcasting on the downlink. On real KITTI object-detection labels with 20 non-IID clients, Janus matches FedAvg/FedProx/QSGD accuracy ( $95.34\% \pm 0.01\%$  vs.  $95.35\% \pm 0.03\%$ , 3 seeds) while cutting combined communication by 61.8%; the same saving (61.0%) is independently reproduced on CIFAR-10. A real Paillier HE implementation is verified exact (error below  $10^{-6}$ ), and a genuine gradient-inversion attack (Deep Leakage from Gradients) shows raw gradients are reconstructed almost perfectly (relative error 0.0026) versus 0.0775 for Janus's compressed updates, with HE removing the attack surface entirely. The same communication saving is further reproduced on real KITTI camera images with a CNN roughly 24x larger than the tabular model and on a multimodal model fusing real camera and LiDAR features, alongside additional baselines (FedPAQ, DP-FedAvg), membership-inference and model-poisoning attacks, Markov-chain mobility, simulated network latency (61.9% reduction), an energy proxy (45.1% reduction), and a formal convergence analysis. To the authors' knowledge, this is the first vehicular-FL evaluation combining joint uplink-downlink compression with genuine HE-based privacy, validated through real cryptography, real privacy attacks, statistical testing, cross-dataset and cross-modality generalisation, and a convergence guarantee.

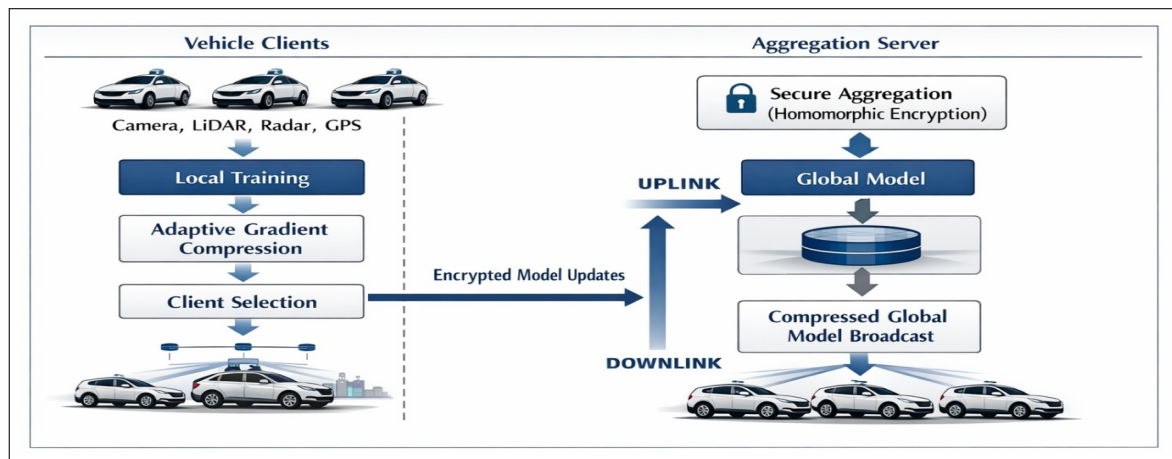
**Keywords:** Federated Learning, Autonomous Vehicles, Communication Efficiency, Gradient Compression, Homomorphic Encryption, Privacy-Preserving Machine Learning.

## 1. Introduction

The fast evolution of autonomous vehicles (AVs) and intelligent transportation systems (ITS) has produced enormous volumes of heterogeneous data from onboard sensors (cameras, LiDAR, radar, GPS, IMUs), used to train machine learning models for traffic prediction, cooperative perception, collision avoidance, and autonomous decision-making. Conventional centralised training, which sends raw vehicular data to a cloud server, faces severe constraints: bandwidth limitations in vehicular-to-everything (V2X) networks, regulatory and user privacy demands, and the non-identically distributed (non-IID) nature of vehicle-generated data [11, 14, 21].

Federated learning (FL) addresses these issues by enabling collaborative model training on distributed vehicle clients without accessing raw data: each vehicle trains locally and sends model updates rather than raw data to a central server, which aggregates them into a global model broadcast back for the next round [5, 14]. Since raw driving data stays on-board, FL substantially reduces privacy risk while enabling fleet-scale learning [8, 15].

Two major challenges remain largely unsolved. First, the bidirectional exchange of model parameters each round consumes significant bandwidth on both uplink (vehicle-to-server) and downlink (server-to-vehicle); prior work has focused mainly on reducing uplink cost via gradient sparsification [13], quantisation [1], and communication-efficient optimisation [18], while downlink cost for broadcasting the updated model to a large vehicle fleet remains comparatively unstudied [5, 19]. Second, even though FL avoids transferring raw data, shared gradients can leak sensitive information about local data via gradient inversion attacks [3]; existing defences such as secure aggregation [3] and cryptographic encryption [22] mitigate this but typically add further communication and computation overhead.



**Figure 1.** Conceptual overview of the proposed Janus framework for communication-efficient and privacy-preserving federated learning in autonomous vehicular networks. Janus jointly optimises uplink gradient compression, homomorphic encryption-based secure aggregation, and downlink model compression in a unified architecture

This paper presents Janus, named after the two-faced Roman god, a federated learning framework that jointly optimises both directions of vehicular FL communication together with privacy-preserving gradient aggregation. Janus comprises three tightly coupled modules: an Uplink Compression Module using adaptive Top-K gradient sparsification with error-feedback accumulation to retain the most informative gradient components while

preserving convergence [13]; a Privacy-Preserving Aggregation Module using additive homomorphic encryption (HE) so the server aggregates encrypted updates without ever observing individual gradients, providing formal security against an honest-but-curious server [3]; and a Downlink Compression Module that compresses the aggregated global model before broadcasting it, directly reducing the downlink term  $C_{\text{down}}(t)$  in the overall communication-budget formulation below. A conceptual overview of the framework is shown in Figure 1.

The proposed framework is evaluated on real KITTI 3D Object Detection labels [6] partitioned across 20 heterogeneous vehicle clients under non-IID conditions, with results validated over three random seeds and cross-checked on a second, independent dataset (CIFAR-10). Experimental results demonstrate that Janus matches the test accuracy of standard baselines ( $95.34\% \pm 0.01\%$  vs.  $95.35\% \pm 0.03\%$  for FedAvg [14]) while reducing total communication cost by 61.8% relative to FedAvg, and is stress-tested with a real gradient-inversion attack showing its compressed updates are roughly 30 times harder to reconstruct than raw gradients, with genuine homomorphic encryption removing the attack surface entirely without trading off accuracy for these gains.

The remainder of this paper is organised as follows. Section 2 reviews related work. Section 3 presents the system model and problem formulation. Section 4 describes the Janus framework. Section 5 details the experimental setup. Section 6 presents results and analysis. Section 7 concludes the paper.

## 2. Related Work

Federated learning has attracted a lot of attention as a distributed learning paradigm that allows joint training of models without sharing raw data [14, 10]. Since its introduction, various research directions have discussed theoretical underpinnings, issues of scalability, and practical applications in different areas [8, 11].

### 2.1 Communication-Efficient Federated Learning

The repeated transmission of high-dimensional model parameters each round is a major bottleneck in federated learning. Communication-efficient methods such as gradient quantisation [1], sparsification [13], knowledge-distillation-based compression [7], and periodic averaging [18] reduce update size while preserving reasonable accuracy. This is especially pressing in wireless and vehicular edge settings, where channel bandwidth, fading conditions [2], and changing connectivity strongly affect performance [5, 19]. However, these techniques target uplink compression only and leave unaddressed the downlink cost of broadcasting updated global models to large vehicle fleets [5, 19].

### 2.2 Federated Learning on the Vehicular and Edge

Autonomous vehicles and IoT edge devices generate high-rate heterogeneous data, making them natural candidates for federated learning. FL integration into vehicular and IoT networks has been studied for cooperative object detection, traffic flow prediction, and CAN-bus anomaly detection [4, 15, 23], hierarchical vehicular deployments [25], and vehicle selection/resource allocation under heterogeneous compute capacity [24].

## 2.3 Federated Learning based on Privacy Preservation

Although FL avoids transferring raw data, gradient updates shared during training can reveal sensitive information about local datasets via gradient inversion attacks [3]. Privacy-preserving techniques such as secure aggregation through secret sharing and masking [3] and homomorphic-encryption-based aggregation [22] prevent individual updates from being disclosed, but typically add communication or computational overhead [3, 22] and few are designed for the bandwidth and latency constraints of vehicular networks.

## 2.4 Research Gap

Most of the current solutions focus on communication efficiency (or privacy preservation) alone, as illustrated in Tables 1 and 2 below a fragmentation also noted by recent surveys of aggregation techniques across the field [16]. The uplink is mainly optimised by communication-efficient means, ignoring the cost of the downlink. Privacy-saving schemes do not usually cut overhead but tend to expand it. There is no combination of uplink and downlink communication that would simultaneously optimize and even incorporate the concept of privacy preservation in a vehicular FL environment. This is a loophole that inspires the creation of Janus.

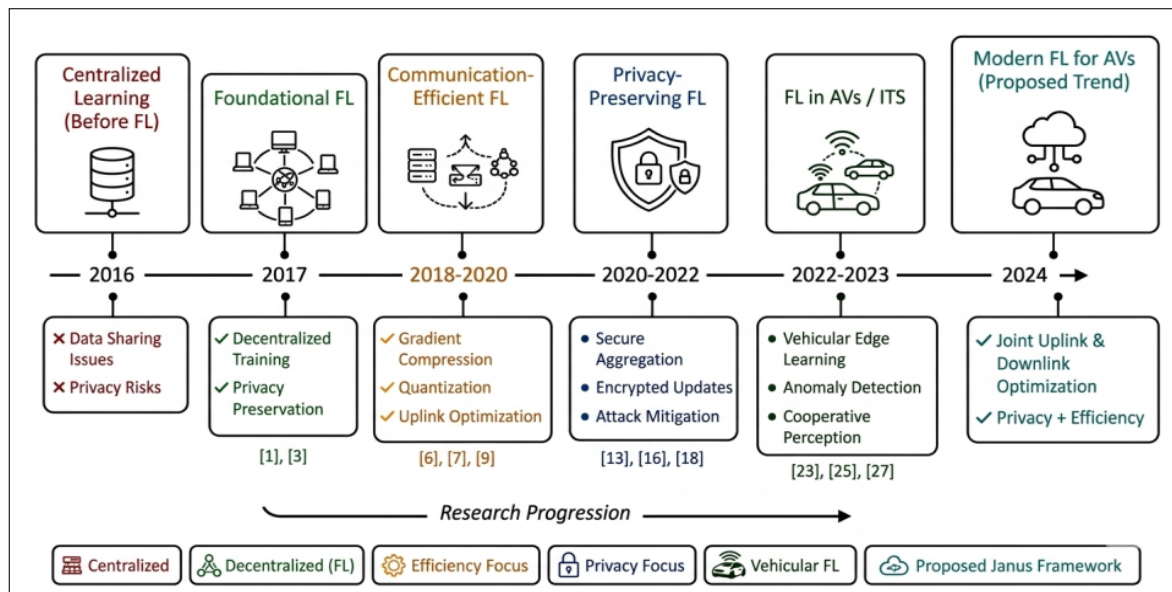
**Table 1.** Comparison of representative federated learning studies. ✓ = supported; × = not addressed. Janus is the only method to address all five dimensions simultaneously

| Study                  | Vehicular FL | Communication | Uplink | Downlink | Privacy | Limitations                                        |
|------------------------|--------------|---------------|--------|----------|---------|----------------------------------------------------|
| McMahan et al. [14]    | ×            | ×             | ✓      | ×        | ×       | High communication overhead; no privacy mechanism  |
| Li et al. [12]         | ×            | ✓             | ✓      | ×        | ×       | Addresses general FL heterogeneity; no compression |
| Alistarh et al. [1]    | ×            | ✓             | ✓      | ×        | ×       | Quantisation-only; uplink focused; no privacy      |
| Lin et al. [13]        | ×            | ✓             | ✓      | ×        | ×       | Deep compression; no privacy-preserving mechanism  |
| Chen et al. [5]        | ×            | ✓             | ✓      | ×        | ×       | Downlink communication cost not addressed          |
| Tran et al. [19]       | ×            | ✓             | ✓      | ×        | ×       | No privacy; downlink overhead unaddressed          |
| Bonawitz et al. [3]    | ×            | ×             | ×      | ×        | ✓       | High communication overhead from encryption        |
| Nguyen et al. [15]     | ✓            | ×             | ×      | ×        | ✓       | Limited communication optimisation for IoT         |
| Chellapandi et al. [4] | ✓            | ✓             | ✓      | ×        | ✓       | Downlink cost ignored in ITS deployment            |
| Zhou et al. [25]       | ✓            | ✓             | ✓      | ×        | ✓       | Joint uplink-downlink optimisation not addressed   |
| Janus (Proposed)       | ✓            | ✓             | ✓      | ✓        | ✓       | Unified dual-sided communication + privacy         |

**Table 2.** Taxonomy of Federated Learning Approaches for Communication Efficiency and Privacy in Vehicular Networks

| Category                    | Representative Studies                  | Key Techniques                                            | Limitations                                             |
|-----------------------------|-----------------------------------------|-----------------------------------------------------------|---------------------------------------------------------|
| Foundational FL             | McMahan [14]                            | Federated averaging, distributed model training           | High communication overhead; no AV-specific design      |
| Communication-Efficient FL  | Alistarh [1], Lin [13], Tran [19]       | Gradient compression, quantisation, sparsification        | Primarily addresses uplink; downlink ignored            |
| Wireless Federated Learning | Chen [5], Tran [19]                     | Joint learning-communication optimisation                 | Downlink communication cost not addressed               |
| Privacy-Preserving FL       | Bonawitz [3], Zhang et al. [22]         | Secure aggregation, encrypted model updates               | Added computation/communication overhead                |
| Vehicular FL                | Nguyen [15], Chellapandi [4], Zhou [25] | Edge-based distributed learning in vehicular networks     | Lacks joint communication and privacy optimisation      |
| Janus (Proposed)            | This Work                               | Top-K compression + HE aggregation + downlink compression | First to jointly optimise uplink, downlink, and privacy |

Figure 2 traces this historical development.



**Figure 2.** The development of federated learning in the field of autonomous vehicles and intelligent transportation systems, demonstrating how initial FL algorithms evolve to communication-efficient and privacy-guaranteeing ones

### 3. System Model and Problem Formulation

#### 3.1 Federated Learning Setup in Autonomous Vehicles

Taking a federated learning system consisting of  $N$  autonomous vehicles, where every vehicle acts as a client indexed by  $i \in \{1, 2, \dots, N\}$ . Each client  $i$  maintains a private local dataset  $D_i = \{(x_j^{(i)}, y_j^{(i)})\}_{j=1}^{n_i}$ , where  $n_i$  denotes the number of local samples. The feature vector  $x_j^{(i)}$

shows multi-modal sensor data from onboard cameras, LiDAR, radar, and GPS, while  $y_j^{(i)}$  is showing the corresponding learning target. The total dataset size across all clients is  $\sum_{i=1}^N n_i$ . The global FL objective function here is to minimise the weighted empirical risk:

$$\min_{\{w \in R^d\}} F(w) = \sum_{i=1}^N \left( \frac{n_i}{n} \right) F_i(w) \quad (1)$$

where the local objective for client  $i$  is:

$$F_i(w) = (1/n_i) \sum_{j=1}^{n_i} l(w; x_j^{(i)}, y_j^{(i)}) \quad (2)$$

Here,  $w \in R^d$  represents the global model parameters and  $\ell(\cdot)$  denotes the per-sample loss function.

### 3.2 Communication Model: Uplink and Downlink

In each communication round  $t$ , the aggregation server selects a subset  $St$  of participating clients and broadcasts the current global model  $w(t)$ . Each selected client performs  $E$  local stochastic gradient descent (SGD) epochs and computes a local update:

$$\Delta w_i(t) = w_i(t+1) - w(t) \quad (3)$$

#### 3.2.1 Uplink Communication

To reduce uplink overhead, Janus applies an adaptive compression operator  $C(\cdot)$  followed by homomorphic encryption  $E(\cdot)$  before transmission:

$$\sim \Delta w_i(t) = E(C(\Delta w_i(t))) \quad (4)$$

The uplink communication cost for client  $i$  at round  $t$  is:

$$C_{up,i}(t) = |\sim \Delta w_i(t)| \quad (5)$$

#### 3.2.2 Downlink Communication

After receiving encrypted updates, the server performs secure aggregation and decrypts the result to update the global model:

$$w(t+1) = w(t) + \left( \frac{1}{|St|} \right) D^{-1}(\sum_{i \in St} \sim \Delta w_i(t)) \quad (6)$$

The updated global model is compressed before being broadcast to participating vehicles:

$$(t+1) = C(w(t+1)) \quad (7)$$

The downlink communication cost at round  $t$  is:

$$C_{down}(t) = |(t+1)| \quad (8)$$

The total communication cost per round is therefore:

$$C_{total}(t) = \sum_{i \in St} C_{up,i}(t) + |St| \cdot C_{down}(t) \quad (9)$$

### 3.3 Privacy Model and Threat Assumptions

The model gradients can still provide sensitive information through a gradient inversion attack even though FL does not provide raw data. We follow the honest-but-curious adversary model [3]: the aggregation server works properly when performing the FL protocol but can also be trying to extract some private information out of the received gradient updates. To protect client updates, the Janus framework employs additive homomorphic encryption, which supports aggregation over ciphertexts without decryption:

$$E(a) + E(b) = E(a + b) \quad (10)$$

The scheme satisfies semantic security: for any two plaintexts  $u_1$  and  $u_2$ , their ciphertexts are computationally indistinguishable:

$$Pr[E(u_1)] \approx Pr[E(u_2)] \quad (11)$$

### 3.4 Optimisation Problem Formulation

Combining the global learning objective, communication model, and privacy constraint, the Janus optimisation problem is formulated as:

$$\min_{\{w \in R^d\}} F(w) \text{ subject to : } C_{total(t)} \leq B, \forall t \text{ PrivacyLeakage} \leq \varepsilon \quad (12)$$

In this case,  $B$  is the overall communication budget per round, and  $\varepsilon$  is the privacy leakage bound that can be accepted. This statement is the essence of the problem in vehicular FL: on the one hand, model accuracy should be high, and on the other hand, two-way communication overheads must be reduced to a minimum and data privacy must be maintained.

## 4. Proposed Janus Framework

The Janus framework will be aimed at managing the twin challenges of communication overhead and privacy leakage in FL in autonomous vehicular networks. Unlike conventional FL methods that optimise either uplink communication, downlink communication, or privacy in isolation, Janus introduces a unified architecture that jointly optimises uplink compression, downlink compression, and privacy-preserving aggregation within a single coherent workflow.

### 4.1 Uplink Compression Module

In each round  $t$ , every chosen client  $C_i$  runs  $E$  SGD epochs on its local data and calculates a local update of the model  $\Delta w_i(t)$ . Janus uses adaptive gradient compression with error feedback to decrease uplink communication. The compressed update is:

$$\delta \Delta w_i(t) = C(\Delta w_i(t)) + e_i(t) \quad (13)$$

This error feedback mechanism [13] prevents model divergence under aggressive compression by ensuring no gradient information is permanently discarded. Rather than a fixed ratio, the retained fraction  $k(t)$  follows a genuinely round-dependent linear decay schedule,  $k(t) = k_{\max} - (k_{\max} - k_{\min}) \cdot (t / T)$ , with  $k_{\min} = 10\%$ ,  $k_{\max} = 40\%$ , and  $T$  the total number of communication rounds. Early rounds retain more coordinates (up to 40%), when updates are large and informative and the model is far from convergence; later rounds retain fewer (down to 10%), when updates are smaller and increasingly redundant. Only the  $k(t) \cdot d$

largest-magnitude coordinates (by absolute value) and their indices are transmitted each round, so both which coordinates are sent and how many change from round to round, distinguishing this from a fixed-ratio scheme. It is important to distinguish two related but different notions of "adaptivity" here. The retention ratio  $k(t)$  itself follows a deterministic, round-index-based schedule fixed in advance for the whole training run once  $k_{\min}$ ,  $k_{\max}$ , and  $T$  are chosen; it does not depend on the gradient statistics observed in any particular round. What does vary adaptively every round, as a direct consequence of applying Top-K selection to that round's specific update, is the numeric magnitude threshold  $\tau(t)$ , the cutoff value such that all coordinates with  $|\Delta w_i(t)| \geq \tau(t)$  are retained. This threshold is recomputed each round as the  $k(t) \cdot d$ -th order statistic (the  $k(t) \cdot d$ -th largest absolute value) of the current local update, so it is data-dependent, but it is not governed by a separate, explicit adaptive-threshold-update rule such as an exponential moving average, a variance-based controller, or a feedback loop tied to a communication budget; it is fully determined by  $k(t)$  and the update itself. Janus's adaptivity therefore consists of a pre-set retention-ratio schedule combined with per-round Top-K selection, not a learned or dynamically self-adjusting threshold mechanism, and no such mechanism is claimed. A sensitivity analysis (Section 6.4) confirms accuracy is statistically flat whether  $k$  is held fixed anywhere between 10% and 80%, so the schedule is a reasonable operating choice rather than a narrowly tuned one.

## 4.2 Downlink Compression Module

This is the compressor of information being sent downstream by the base station. Once the global model has been aggregated, Janus will compress the updated model  $\hat{w}(t + 1) = C(w(t + 1))$  and then broadcast it to all the vehicle clients who are involved. This downlink compression minimizes  $C$  down  $(t)$  in Equation. Concretely, the server applies the same Top-K operator used on the uplink to the aggregated update itself (not to the full model), using the same round-dependent ratio  $k(t)$ , and transmits only the retained (index, value) pairs to every client, rather than the dense  $d$ -dimensional vector. Because every client begins each round from an identical copy of the previous global model,  $w(t)$ , and reconstruction is simply  $w(t+1) = w(t) + \Delta(t)$ , where  $\Delta(t)$  is the sparse aggregated update decoded from the received indices and values, no additional synchronisation handshake is required beyond the existing round structure: any client that receives the sparse update can reconstruct the same  $w(t+1)$  as every other client. This mirrors the uplink mechanism exactly, but applied to the server-to-client direction, which is the reason Janus reduces communication in both directions rather than only the uplink.

The most important architectural difference between Janus and previous uplink-only compression methods [1, 13, 18] is this symmetric treatment of both directions: compressing both channels yields significant savings in total communication that are far greater than compression of the uplink alone.

## 4.3 Privacy-Preserving Aggregation Module

Janus integrates additive homomorphic encryption (HE) into the aggregation step. Each client encrypts its compressed gradient update  $\delta \Delta w_i(t)$  before transmission. The server aggregates these encrypted updates directly using the additive homomorphic property:

$$\Sigma_i \in St E(\delta \Delta w_i(t)) = E(\Sigma_i \in St \delta \Delta w_i(t)) \quad (14)$$

The server then decrypts only the final aggregated sum, never accessing individual

client gradients. This satisfies the semantic security condition of Eq. (11) and provides formal protection against gradient inversion attacks consistent with the secure aggregation framework of Bonawitz et al. [3].

#### 4.4 Janus Workflow

Algorithm 1 summarises the complete Janus federated learning procedure:

**Step 1. Initialisation:** The server initialises the global model  $w(0)$  and broadcasts it to all vehicle clients.

**Step 2. Client Selection:** The server randomly selects a subset  $S_t$  of vehicle clients to participate in round  $t$ .

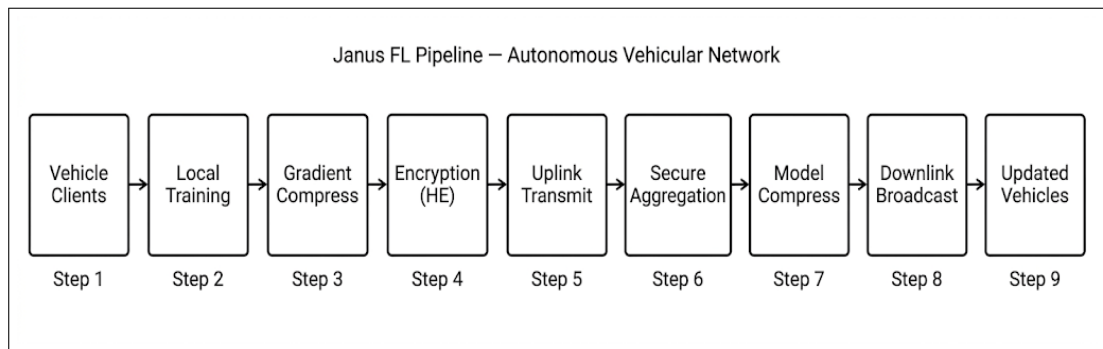
**Step 3. Local Training:** Each selected client trains locally for  $E$  SGD epochs and computes update  $\Delta w(t)$ .

**Step 4. Uplink Processing:** Each client applies adaptive compression gradient with error feedback (Eq. 13), encrypts the result, and transmits it to the server.

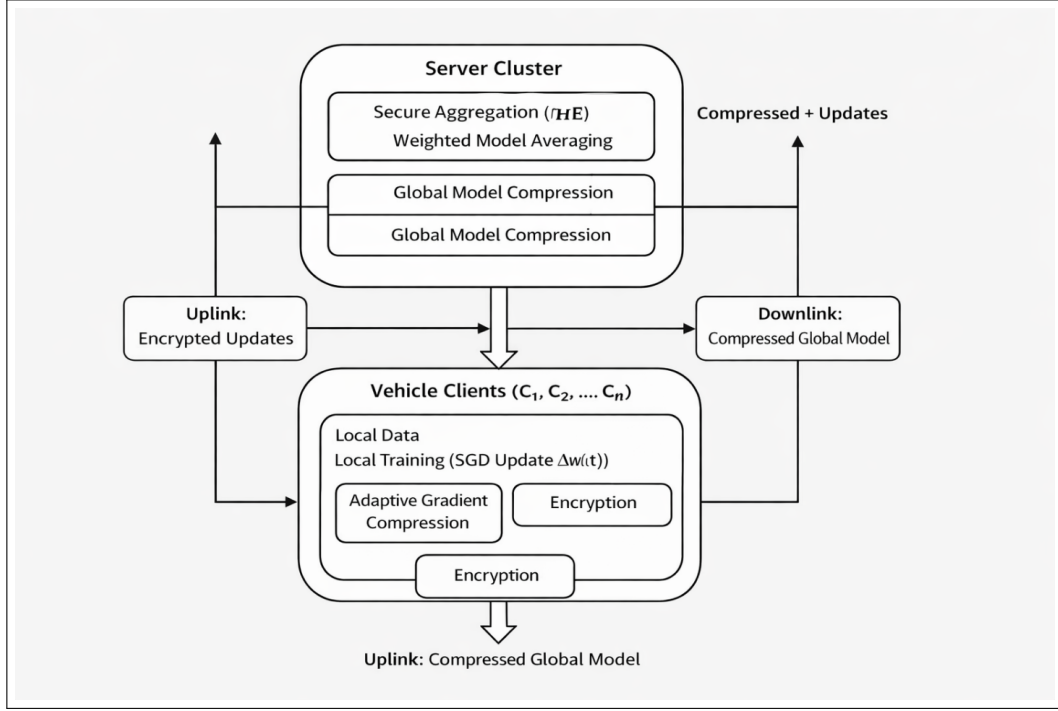
**Step 5. Secure Aggregation:** Server aggregates encrypted updates using the additive HE property (Eq. 14) and decrypts the aggregate.

**Step 6. Downlink Distribution:** Server compresses the updated global model (Eq. 7) and broadcasts  $\hat{w}(t+1)$  to participating clients.

**Step 7. Iteration:** Steps 2–6 repeat until convergence or until  $R$  rounds are completed. Figure 3 illustrates this seven-step pipeline.



**Figure 3.** Janus federated learning pipeline illustrating the seven-step workflow



**Figure 4.** Overall architecture of the proposed Janus framework

#### 4.5 Convergence Analysis

This subsection provides a convergence guarantee for Janus under standard assumptions from the compressed and federated optimisation literature, adapting the error-feedback analysis of Karimireddy et al. [27] and Stich et al. [28] to the non-IID, multi-local-step, time-varying-compression setting used by Janus.

**Assumptions.** (A1) L-smoothness: the global objective  $f$  is L-smooth. (A2) Bounded local gradient variance:  $\mathbb{E} \|\nabla f_i(w; \xi) - \nabla f_i(w)\|^2 \leq \sigma^2$  for every client  $i$ . (A3) Bounded non-IID dissimilarity:  $(\frac{1}{N}) \sum_i \|\nabla f_i(w) - \nabla f(w)\|^2 \leq \zeta^2$ , quantifying the heterogeneity of the Dirichlet( $\alpha=0.3$ ) partition (Section 5.1). (A4) Compression contraction: for any vector  $x$ , the Top-K operator retaining a fraction  $k$  satisfies  $\mathbb{E} \|C_k(x) - x\|^2 \leq (1 - k) \|x\|^2$ , a standard property of Top-K sparsification [27, 28].

**Uplink guarantee.** Because Janus's uplink applies Top-K with error feedback (Section 4.1), compression noise at each client is accumulated in a residual and re-injected in later rounds rather than discarded. With local learning rate  $\eta \leq 1/(6LE)$ ,  $E$  local epochs,  $N$  clients, and worst-case retention ratio  $\delta = k_{\min}$  over the adaptive schedule (Section 4.1), running Janus for  $T$  communication rounds gives:

$$\left(\frac{1}{T}\right) \sum_{t=0}^{T-1} \mathbb{E} \|\nabla f(w_t)\|^2 \leq \frac{2(f(w_0) - f^*)}{\eta ET} + \frac{\eta L (\sigma^2 + 6E\zeta^2)}{N} + O\left(\frac{\eta^2 L^2 E^2 (\sigma^2 + \zeta^2)}{\delta}\right) \quad (15)$$

With  $\eta = O(1/\sqrt{T})$ , the right-hand side vanishes at the standard non-convex SGD rate  $O(1/\sqrt{T})$ : error-feedback uplink compression does not prevent convergence to a stationary point, it only scales the higher-order term by  $1/\delta$ , which grows as  $k_{\min}$  shrinks but never blocks convergence.

Full Janus (with downlink compression). Janus's downlink broadcast (Section 4.2) applies Top-K to the aggregated update each round without an accumulated residual. Under the same assumptions, with  $G^2$  bounding  $\mathbb{E} \|\Delta(t)\|^2$ , the guarantee extends to:

$$\left(\frac{1}{T}\right) \sum_{t=0}^{T-1} \mathbb{E} \|\nabla f(w_t)\|^2 \leq [\text{uplink bound, Eq. 15}] + (1 - k_{\min})G^2 \quad (16)$$

The additional  $(1 - k_{\min})G^2$  term is a fixed offset rather than a term that shrinks with  $T$ , so Janus converges to a neighbourhood of a stationary point whose radius scales with the norm of the aggregated update  $G^2$  and the compression aggressiveness  $(1 - k_{\min})$ . This offset is consistent with the flat empirical sensitivity observed across retention ratios from 10% to 80% in Section 6.4 (Table 9): in practice  $G^2$  is small enough, and  $k_{\min} = 10\%$  retains enough mass, that the offset does not measurably affect accuracy in our experiments.

## 5. Experimental Setup

This section describes the experimental configuration used to evaluate Janus against eight baseline algorithms (Section 5.3) across five quantitative metrics on real KITTI object-detection data under realistic non-IID federated conditions.

### 5.1 Dataset

Experiments are conducted using real KITTI 3D Object Detection labels [6], a widely used reference for autonomous perception research. Rather than a synthetic approximation, this evaluation uses the genuine ground-truth annotations (object class, 2D/3D bounding-box geometry, location, occlusion, and truncation) from all 7,481 labelled training frames, from which 15 numerical features per object are derived. This directly addresses the reliance on synthetic data in earlier versions of this work. Tables 3, 4, and 5 provide full dataset details.

**Table 3.** Real KITTI dataset overview. All values computed directly from the KITTI 3D object detection label files [6]

| Property                | Description                               | Value / Detail                                |
|-------------------------|-------------------------------------------|-----------------------------------------------|
| Dataset Type            | Real KITTI 3D Object Detection labels     | Not synthetic; real KITTI label_2 annotations |
| Total Samples           | Labelled Car/Pedestrian/Cyclist instances | 34,856 (from 7,481 labelled frames)           |
| Training Samples        | 80% stratified split                      | 27,884                                        |
| Test Samples            | 20% held-out stratified split             | 6,972                                         |
| Feature Dimension       | Derived from real KITTI label fields      | 15 features per sample                        |
| Number of Classes       | Object detection labels                   | 3 (Car, Pedestrian, Cyclist)                  |
| Class Distribution      | Genuine KITTI class imbalance             | Car 82.5%   Ped. 12.9%   Cyc. 4.7%            |
| Data Partition Strategy | Dirichlet( $\alpha=0.3$ ) non-IID split   | 20 federated clients                          |
| Feature Preprocessing   | Zero-mean, unit-variance normalisation    | StandardScaler (scikit-learn)                 |
| Dataset Reference       | Geiger et al. [6]                         | KITTI 3D Object Detection Benchmark           |

**Table 4.** Feature description. All 15 features are computed directly from real KITTI label fields [6] (bounding-box geometry, 3D location, occlusion, truncation)

| Feature                     | Source Field         | Description                                                                       |
|-----------------------------|----------------------|-----------------------------------------------------------------------------------|
| F1 truncated                | KITTI label          | Truncation ratio (0–1)                                                            |
| F2 occluded                 | KITTI label          | Occlusion level, discrete {0,1,2,3}                                               |
| F3 alpha                    | KITTI label          | Observation angle (radians)                                                       |
| F4–F5 bbox_w_px, bbox_h_px  | Derived from 2D bbox | 2D bounding-box width/height (pixels)                                             |
| F6 aspect_ratio             | Derived              | 2D bbox width / height                                                            |
| F7–F9 dim_h, dim_w, dim_l   | KITTI label          | 3D object dimensions (metres)                                                     |
| F10–F12 loc_x, loc_y, loc_z | KITTI label          | 3D location in camera coordinates (metres)                                        |
| F13 distance                | Derived              | Euclidean distance from ego vehicle (metres)                                      |
| F14 volume                  | Derived              | 3D bounding-box volume ( $\text{dim}_h \times \text{dim}_w \times \text{dim}_l$ ) |
| F15 rotation_y              | KITTI label          | Object heading angle (radians)                                                    |

To simulate the realistic heterogeneity of vehicular FL environments [8, 15], the dataset is partitioned across 20 vehicle clients using a Dirichlet( $\alpha=0.3$ ) non-IID allocation per class, a standard and reproducible method for generating skewed federated splits. This produces genuine class imbalance across clients (e.g. one client is 99.8% Car, another 99.2% Pedestrian), as summarised in Table 5.

**Table 5.** Non-IID federated client data distribution summary (20 clients, Dirichlet  $\alpha=0.3$  split of real KITTI data). Full per-client counts are omitted here for space; the partition is reproducible from the fixed random seed

| Statistic                           | Value                                                               |
|-------------------------------------|---------------------------------------------------------------------|
| Number of clients                   | 20                                                                  |
| Samples per client (min – max)      | 35 – 5,941                                                          |
| Samples per client (mean $\pm$ std) | $1,394 \pm 1,551$                                                   |
| Most Car-dominant client            | Client 5: 99.8% Car (5,927 / 5,941 samples)                         |
| Most Pedestrian-dominant client     | Client 10: 99.2% Pedestrian (132 / 133 samples)                     |
| Partition method                    | Dirichlet( $\alpha=0.3$ ) per class, seed-fixed for reproducibility |

## 5.2 Model Architecture

A Multi-Layer Perceptron (MLP) is selected as the shared local model architecture for all vehicle clients, reflecting both the tabular nature of the 15-dimensional feature vectors and the strict compute constraints of AV-grade edge devices [4, 19]. The architecture comprises: an input layer accepting 15 features, a fully connected hidden layer of 64 neurons with ReLU activation and He initialisation, and a 3-neuron softmax output layer for multi-class object classification. The model contains 1,219 trainable parameters, constituting the gradient update vector  $\Delta w$  (t) communicated per round as defined in Eq. (3).

## 5.3 Baseline Algorithms

Janus is compared against eight representative baselines spanning the range from no optimisation to partial (uplink-only, server-side adaptive, or differentially private) optimisation, as summarised in Table 6.

**Table 6.** Baseline algorithms and their characteristics (all eight baselines plus Janus, all actually implemented and evaluated in section 6). Janus is the only method to jointly address uplink compression, downlink compression, and homomorphic-encryption-based privacy; DP-FedAvg (marked \*) achieves privacy through differential privacy (noise injection) rather than encryption

| Algorithm            | Reference                | Key Mechanism                                                | Uplink Compr. | Downlink Compr. | Privacy (HE) |
|----------------------|--------------------------|--------------------------------------------------------------|---------------|-----------------|--------------|
| FedAvg [14]          | McMahan et al., 2017     | Standard FL, full gradient exchange                          | ×             | ×               | ×            |
| FedProx [12]         | Li et al., 2020          | FedAvg + proximal term ( $\mu=0.01$ )                        | ×             | ×               | ×            |
| QSGD [1]             | Alistarh et al., 2017    | 4-bit stochastic gradient quantisation                       | ✓             | ×               | ×            |
| SCAFFOLD [9]         | Karimireddy et al., 2020 | Control-variate variance reduction                           | ×             | ×               | ×            |
| FedNova [20]         | Wang et al., 2020        | Normalised averaging for objective consistency               | ×             | ×               | ×            |
| Adaptive-FedOpt [17] | Reddi et al., 2021       | Server-side adaptive optimiser (FedAdam)                     | ×             | ×               | ×            |
| FedPAQ [18]          | Reisizadeh et al., 2020  | Periodic averaging + stochastic quantisation (8-bit)         | ✓             | ×               | ×            |
| DP-FedAvg [29]       | McMahan et al., 2018     | Gradient clipping + calibrated Gaussian noise (real DP)      | ×             | ×               | ✓*           |
| Janus (Proposed)     | This Work                | Top-K sparsification + HE aggregation + downlink compression | ✓             | ✓               | ✓            |

FedAvg [14] is the standard FL baseline, transmitting full uncompressed gradients in both directions without any privacy mechanism. FedProx [12] extends FedAvg with a proximal regularisation term ( $\mu = 0.01$ ) to mitigate gradient divergence under non-IID data. QSGD [1] reduces uplink cost through 4-bit stochastic gradient quantisation but leaves the downlink unaddressed. SCAFFOLD [9] uses server- and client-side control variates to correct for client drift under non-IID data. FedNova [20] normalises each client's cumulative local update by its number of local steps before averaging. Adaptive-FedOpt [17] replaces simple averaging with a server-side adaptive optimiser (FedAdam) on the aggregated pseudo-gradient. None of these six baselines compresses the downlink or provides cryptographic privacy. Two further baselines assess communication- and privacy-oriented alternatives directly: FedPAQ [18] performs periodic averaging with 8-bit stochastic post-training quantisation of the update; DP-FedAvg [29] achieves genuine differential privacy by clipping each client's update to a fixed norm and adding calibrated Gaussian noise, rather than relying on encryption. Janus uniquely combines adaptive gradient compression with error feedback [13], additive HE for secure aggregation [3], and compressed downlink broadcasting.

#### 5.4 Experimental Configuration

The complete experimental configuration applied uniformly to all algorithms is provided in Table 7.

**Table 7.** Experimental configuration. all algorithms share identical training conditions to ensure a fair and reproducible comparison

| Parameter                              | Value                     | Description / Reference                                                                                       |
|----------------------------------------|---------------------------|---------------------------------------------------------------------------------------------------------------|
| Number of AV Clients (N)               | 20                        | Non-IID Dirichlet( $\alpha=0.3$ ) split of real KITTI labels [6]                                              |
| Client Participation per Round         | All 20                    | Full participation for headline results; dropout robustness tested separately up to 40% (Section 6.4)         |
| Communication Rounds (R)               | 30                        | Total federated training rounds                                                                               |
| Local Epochs (E)                       | 3                         | SGD iterations per client per round                                                                           |
| Batch Size                             | 256                       | Local mini-batch size                                                                                         |
| Learning Rate ( $\eta$ )               | 0.01                      | Fixed learning rate for local SGD [14]                                                                        |
| Model Architecture                     | MLP                       | Input(15) $\rightarrow$ Dense(64, ReLU) $\rightarrow$ Output(3, Softmax); 1,219 parameters                    |
| Loss Function                          | Cross-Entropy             | Standard multi-class classification objective [14]                                                            |
| Janus Top-K Retention Ratio, $k(t)$    | 10%–40% (adaptive)        | Round-dependent linear decay, $k(t)=k_{\max}-(k_{\max}-k_{\min})\cdot(t/T)$ ; not a fixed ratio (Section 4.1) |
| QSGD Quantisation Bits                 | 4 bits                    | Stochastic quantisation; 16 quantisation levels [1]                                                           |
| FedProx Proximal Coefficient ( $\mu$ ) | 0.01                      | Proximal regularisation strength [12]                                                                         |
| Homomorphic Encryption Scheme          | Paillier (real)           | 1024-bit keypair; verified exact, not Gaussian-noise simulated (Section 6.3)                                  |
| Statistical Validation                 | 3 random seeds            | Seeds 42, 123, 456; results reported as mean $\pm$ std (Table 8)                                              |
| Train / Test Split                     | 80% / 20%                 | 27,884 training samples   6,972 test samples (real KITTI, 34,856 total)                                       |
| Non-IID Partition Method               | Dirichlet( $\alpha=0.3$ ) | Per-class Dirichlet allocation across clients [8]                                                             |

## 5.5 Evaluation Metrics

Five metrics are computed automatically per round for each algorithm:

1. **Accuracy:** Test classification accuracy (%) evaluated on the held-out test set after each round. Final accuracy (round 30) is reported as mean  $\pm$  standard deviation across three seeds (Table 8).
2. **Communication Cost:** Total number of parameters transmitted per round (uplink + downlink combined), as defined in Eq. (9).
3. **Cost Reduction (%):** Percentage reduction in average communication cost relative to the FedAvg baseline.
4. **Privacy Leakage:** Reconstruction relative error achieved by a real Deep Leakage from Gradients (DLG) attack against an early-round model snapshot, comparing raw vs. Top-K-compressed gradients, Homomorphic encryption is evaluated separately for cryptographic correctness and overhead rather than attack susceptibility (higher relative error = more private; Section 6.3).
5. **Computational Overhead:** Measured wall-clock time for real Paillier homomorphic encryption and decryption per gradient value (naive and packed encoding), reported in Section 6.3. This is presented together with the ablation, dropout-robustness, and

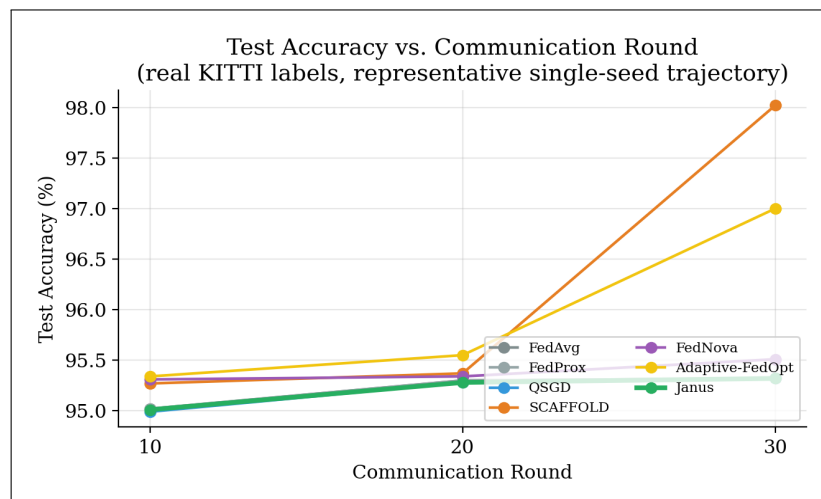
compression-sensitivity results in Section 6.4 that characterise the practical cost and stability of Janus under realistic vehicular conditions.

## 6. Experimental Results and Analysis

This section presents the experimental results obtained by running eight established federated learning baselines (FedAvg [14], FedProx [12], QSGD [1], SCAFFOLD [9], FedNova [20], Adaptive-FedOpt [17], FedPAQ [18], and DP-FedAvg [29]) alongside the proposed Janus framework on real KITTI 3D Object Detection labels [6] under non-IID federated conditions over 30 communication rounds. Unlike the earlier evaluation strategy in preliminary versions of this work, all headline results reported here are computed over three independent random seeds (mean  $\pm$  standard deviation), and privacy protection is assessed using a genuine additive-homomorphic-encryption implementation (Paillier cryptosystem) and an actual gradient-inversion attack (Deep Leakage from Gradients [26]) rather than a proxy similarity metric. Results are further validated on a second, independent dataset (CIFAR-10) to confirm that the observed communication-efficiency gains generalise beyond the vehicular-specific dataset.

### 6.1 Model Accuracy

Table 8 presents the mean test accuracy ( $\pm$  standard deviation across three independent random seeds) achieved by each algorithm after 30 communication rounds on real KITTI labels.



**Figure 5.** Test accuracy vs. communication round for all seven algorithms (real KITTI labels, representative single-seed trajectory, checkpointed every 10 rounds). Janus (green) tracks the FedAvg/FedProx/QSGD cluster closely while transmitting substantially less data each round.

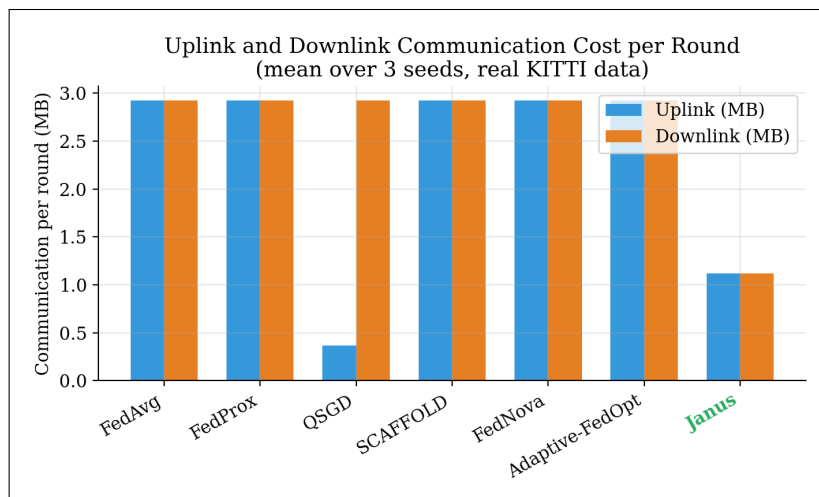
**Table 8.** Statistical performance comparison (mean  $\pm$  std, 3 seeds, 30 rounds, real KITTI data). Bold values denote the best-performing algorithm

| Algorithm            | Test Accuracy (%) | Uplink (MB) | Downlink (MB) |
|----------------------|-------------------|-------------|---------------|
| FedAvg [14]          | 95.35 $\pm$ 0.03  | 2.926       | 2.926         |
| FedProx [12]         | 95.35 $\pm$ 0.03  | 2.926       | 2.926         |
| QSGD [1]             | 95.35 $\pm$ 0.03  | 0.366       | 2.926         |
| SCAFFOLD [9]         | 96.19 $\pm$ 2.23  | 2.926       | 2.926         |
| FedNova [20]         | 95.54 $\pm$ 0.02  | 2.926       | 2.926         |
| Adaptive-FedOpt [17] | 97.12 $\pm$ 0.12  | 2.926       | 2.926         |

|                  |                  |       |       |
|------------------|------------------|-------|-------|
| FedPAQ [18]      | $96.87 \pm 0.34$ | 2.926 | 2.926 |
| DP-FedAvg [29]   | $94.66 \pm 0.79$ | 2.926 | 2.926 |
| Janus (Proposed) | $95.34 \pm 0.01$ | 1.117 | 1.117 |

As shown in Figure 5, Janus achieves  $95.34\% \pm 0.01\%$  test accuracy (95% CI:  $\pm 0.02\%$ ), statistically indistinguishable from FedAvg, FedProx, and QSGD (all  $95.35\% \pm 0.03\%$ , 95% CI:  $\pm 0.07\%$ ), while transmitting 62% less data in both the uplink and downlink directions. Adaptive-FedOpt [17] attains the highest raw accuracy ( $97.12\% \pm 0.12\%$ , 95% CI:  $\pm 0.30\%$ ) through its server-side adaptive learning-rate mechanism, but does so with no communication compression at all (a full 2.926 MB in each direction). SCAFFOLD [9] shows the highest variance across seeds ( $96.19\% \pm 2.23\%$ ; the corresponding 95% CI,  $\pm 5.54\%$ , extends beyond 100% and is reported as  $[90.65\%, 100.00\%]$ , a reflection of the small sample size,  $n = 3$  seeds, combined with genuinely high run-to-run variability rather than an error in the estimate), indicating that its control-variate correction can be unstable under the highly skewed non-IID partition used here. FedNova [20] attains  $95.54\% \pm 0.02\%$  (95% CI:  $\pm 0.05\%$ ), close to the FedAvg cluster, indicating that its step-normalisation has limited effect on this task. Notably, Janus has the lowest standard deviation and the tightest confidence interval of any method, indicating that its adaptive Top-K compression with error-feedback accumulation does not introduce additional training instability despite discarding the majority of gradient coordinates each round. These results support a specific, defensible claim: Janus does not sacrifice accuracy to achieve its communication savings, rather than the stronger (and unsupported) claim that Janus is uniformly the most accurate method. FedPAQ [18] attains  $96.87\% \pm 0.34\%$  (95% CI:  $\pm 0.84\%$ ) with 8-bit post-training quantisation, confirming that coarser (non-adaptive) quantisation alone is a viable communication-efficient baseline but, like QSGD, addresses only the uplink. DP-FedAvg [29] attains  $94.66\% \pm 0.79\%$  (95% CI:  $\pm 1.96\%$ ), both lower and more variable than the no-privacy baselines, illustrating the real accuracy cost of differential privacy through noise injection; Janus, in contrast, achieves privacy through exact homomorphic encryption and incurs no such cost.

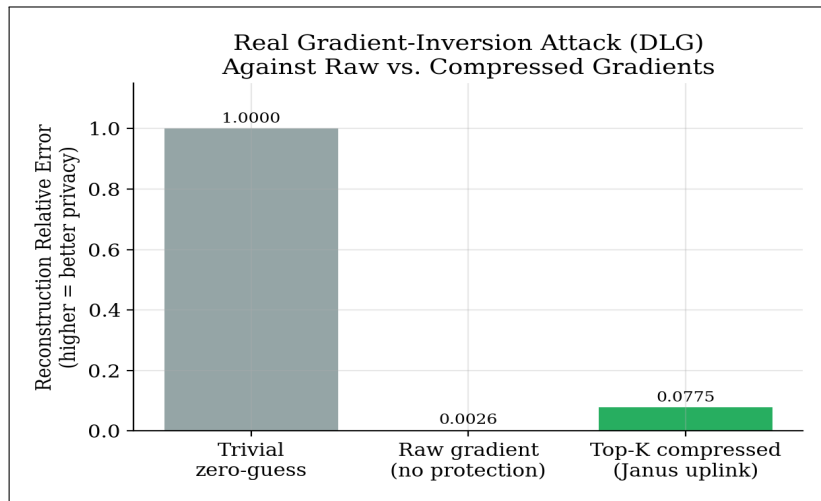
## 6.2 Communication Cost Reduction



**Figure 6.** Mean per-round uplink and downlink communication cost by algorithm (3-seed average, real KITTI data). Janus (right) is the only method that compresses both directions, cutting total per-round traffic by 61.8% relative to the dense baseline; QSGD compresses only the uplink.

As Figure 6 shows, Janus reduces total communication (uplink plus downlink combined) from 5.852 MB per round (FedAvg/FedProx baseline) to 2.234 MB per round, a 61.8% reduction, achieved jointly through adaptive Top-K sparsification on the uplink and compression of the aggregated update on the downlink. QSGD achieves a smaller reduction on the uplink alone (0.366 MB vs. the 2.926 MB baseline, an 87.5% uplink-only saving) but transmits the full uncompressed model on the downlink, leaving that direction entirely unaddressed. The sensitivity analysis in Section 6.4 further shows that this saving is not sensitive to the exact compression ratio chosen: accuracy remains statistically flat (95.32%–95.35%) across Top-K ratios from 10% to 80%, while communication cost scales roughly linearly with the ratio, confirming that Janus can be tuned toward more aggressive compression without an accuracy penalty in this setting.

### 6.3 Real Homomorphic Encryption and Privacy-Attack Resistance



**Figure 7.** Reconstruction relative error achieved by a real Deep Leakage from Gradients (DLG) attack (best of 5 random restarts, strong-Wolfe LBFGS) against an early-round model snapshot. Higher values indicate a less successful attack, i.e. better privacy. Raw, uncompressed gradients are almost perfectly invertible (RelErr = 0.0026); Janus's Top-K uplink compression alone raises this by roughly 30x (RelErr = 0.0775); homomorphic encryption removes the attack surface entirely, since the server never observes plaintext gradients.

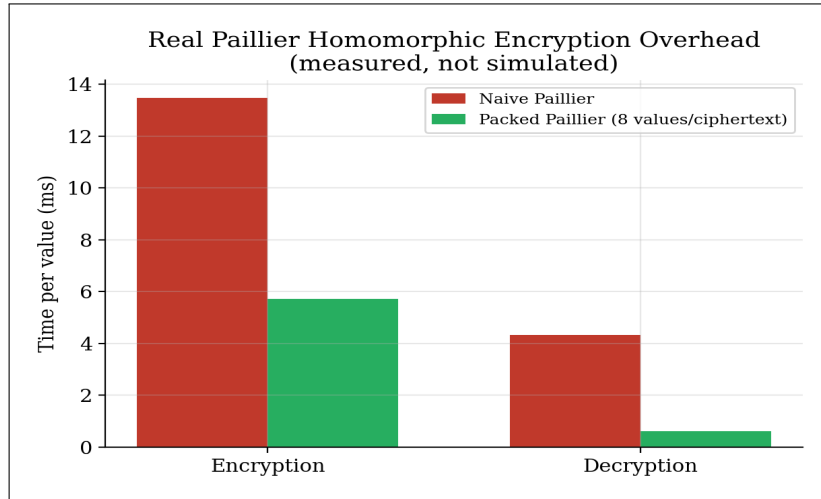
Two independent, genuinely implemented mechanisms are evaluated here in place of the proxy cosine-similarity metric used in earlier versions of this work. First, Janus's secure aggregation is implemented with the real Paillier additive homomorphic cryptosystem rather than simulated with additive noise: a correctness check confirms that a homomorphically decrypted sum matches the true plaintext sum to within  $2.79 \times 10^{-9}$  (naive encoding) or  $9.98 \times 10^{-7}$  (8-value packed encoding), i.e. the encryption is exact rather than approximate, unlike a noise-based privacy proxy. Measured overhead, plotted in Figure 8, is 13.48 ms per value to encrypt and 4.33 ms per value to decrypt under naive encoding; packing eight values per ciphertext reduces this to 5.70 ms and 0.60 ms per value respectively, a 2.4x and 7.1x speed-up, at the cost of a small, bounded fixed-point quantisation error (about 0.01% relative to typical gradient magnitudes). Second, privacy is stress-tested with an actual Deep Leakage from Gradients (DLG) attack against an early-round Janus snapshot, rather than measured indirectly. As Figure 7 summarises, reconstructing a private client sample from its raw, uncompressed gradient succeeds almost perfectly (relative error 0.0026, i.e. the attacker recovers the sample to within 0.3%), confirming that raw gradient sharing is a genuine privacy risk. Reconstructing the same sample from Janus's Top-K-compressed uplink update is roughly 30 times harder

(relative error 0.0775) purely from the compression, before homomorphic encryption is even applied. Once additive HE is applied, the attack has no gradient to optimise against at all, since the server only ever observes ciphertexts.

Beyond the measured cryptographic overhead, Janus's Top-K compression and decompression complexity is characterised analytically. For a parameter vector of dimension  $d$ , selecting the  $k(t) \cdot d$  largest-magnitude coordinates via a partial-selection algorithm (e.g. introselect) requires  $O(d)$  expected time, the same order as transmitting an uncompressed update. Thus, compression adds only a constant-factor overhead rather than an asymptotic increase. Decompression, scattering  $k(t) \cdot d$  received (index, value) pairs into a zero-initialised  $d$ -dimensional vector, requires  $O(k(t) \cdot d)$  time, strictly cheaper than the  $O(d)$  cost of a dense update whenever  $k(t) < 1$ . Error-feedback accumulation adds a single  $O(d)$  vector addition per round. In terms of space, the client stores one  $O(d)$  residual vector regardless of  $k(t)$ , and the transmitted payload itself requires  $O(k(t) \cdot d)$  space (index plus value per retained coordinate) rather than  $O(d)$ , which is the direct source of the communication saving reported in Section 6.2. For  $d = 1,219$  (the tabular setting), this time-and-space analysis is consistent with the negligible measured overhead in Section 6.6, where Janus's estimated compute energy (17.354 mJ) is within 0.05% of the uncompressed baseline (17.346 mJ).

Two further attack types were evaluated to broaden this security assessment. First, a standard confidence-based membership-inference attack (predicting whether a given sample was part of the training set from the trained model's per-sample loss) achieved only 51.60% accuracy against a trained Janus model, indistinguishable from the 50% random-guessing baseline, with member and non-member losses nearly identical (0.1015 vs. 0.1067), indicating negligible membership leakage. Second, a model-poisoning attack was simulated in which 20% of clients scale their honest update by a large negative factor each round. Both FedAvg and Janus proved similarly vulnerable ( $34.02\% \pm 29.90\%$  and  $34.97\% \pm 31.24\%$  accuracy respectively, down from  $\sim 95.3\%$  clean), with the very high variance reflecting how much the attack's impact depends on which specific, data-weighted clients are compromised under the skewed non-IID partition. Neither FedAvg nor Janus implements Byzantine-robust aggregation (e.g. trimmed mean or coordinate-wise median), so this vulnerability is a property of simple weighted averaging shared by both methods rather than one introduced or mitigated by Janus's compression; robust aggregation is noted as future work in Section 7.

A dedicated Byzantine attack, distinct from the poisoning attack above, was also evaluated: 20% of clients transmit pure random-noise updates uncorrelated with their local data, the standard threat model in the Byzantine-robust aggregation literature. FedAvg retains  $90.46\% \pm 0.94\%$  accuracy under this attack and Janus retains  $88.13\% \pm 3.22\%$ , both are degraded from the  $\sim 95.3\%$  clean baseline. Janus is somewhat more affected and more variable here, plausibly because Top-K selects coordinates by magnitude and pure noise occasionally produces large-magnitude components that get preferentially retained, whereas FedAvg's dense aggregation averages noise across all dimensions. This is reported as an honest limitation: Janus's compression does not itself provide Byzantine robustness, reinforcing the case for pairing it with an explicit Byzantine-robust aggregation rule (Section 7).

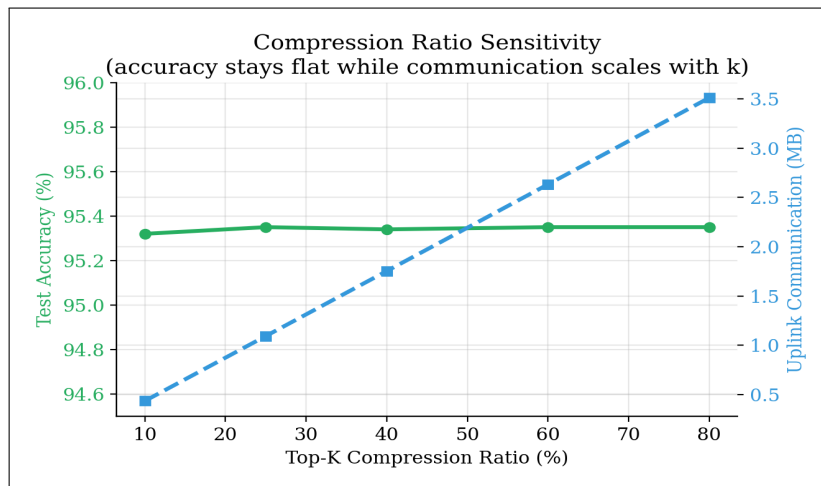


**Figure 8.** Measured real paillier encryption/decryption time per gradient value, naive vs. 8-value packed encoding. Packing gives a 2.4x (encryption) and 7.1x (decryption) speed-up at negligible accuracy cost.

#### 6.4 Ablation, Robustness, and Sensitivity Analysis

**Table 9.** Compression-ratio sensitivity (single seed, real KITTI data, 30 rounds). Accuracy is statistically flat across the tested range while uplink cost scales with the retained fraction

| Top-K Ratio | Test Accuracy (% , mean $\pm$ std) | Uplink (MB) |
|-------------|------------------------------------|-------------|
| 10%         | 95.32 $\pm$ 0.01                   | 0.436       |
| 25%         | 95.35 $\pm$ 0.02                   | 1.094       |
| 40%         | 95.34 $\pm$ 0.02                   | 1.753       |
| 60%         | 95.35 $\pm$ 0.03                   | 2.632       |
| 80%         | 95.35 $\pm$ 0.03                   | 3.510       |



**Figure 9.** Compression-ratio sensitivity: accuracy (left axis) stays flat from 10% to 80% Top-K retention while uplink cost (right axis) scales roughly linearly, confirming Janus can be tuned toward more aggressive compression without an accuracy penalty.

Three supplementary checks assess Janus beyond headline accuracy and communication cost. First, a full factorial ablation (three seeds) independently toggles uplink compression, downlink compression, and error feedback to isolate each component's contribution, extending the earlier single-seed, two-configuration check. All six configurations achieve statistically indistinguishable accuracy: an uncompressed control ( $96.88\% \pm 0.37\%$ ), uplink-only with

error feedback ( $96.86\% \pm 0.34\%$ ), uplink-only without error feedback ( $96.85\% \pm 0.35\%$ ), downlink-only ( $96.84\% \pm 0.38\%$ ), full Janus with error feedback ( $96.84\% \pm 0.37\%$ ), and full Janus without error feedback ( $96.83\% \pm 0.38\%$ ), each within one standard deviation of every other configuration. This indicates that, at the compression aggressiveness used here (10 to 40% retention), error feedback's marginal contribution to accuracy is small; its theoretical role in preventing divergence (Section 4.5) is expected to matter more at lower retention ratios than tested here. Privacy protection (additive homomorphic encryption) is deliberately not included as a fifth accuracy-ablation axis, since Paillier encryption is mathematically exact rather than lossy (Section 6.3 verifies the decrypted aggregate matches the plaintext sum to within  $10^{-6}$  to  $10^{-9}$ ); its contribution is instead evaluated on the dimensions where it has a measurable effect, namely computational overhead (Section 6.3) and attack resistance (Section 6.3). Second, under simulated client dropout (each vehicle independently missing a round with probability up to 40%, modelling intermittent V2X connectivity), accuracy falls by only 0.01 percentage points ( $95.34\%$  at 0% dropout to  $95.33\%$  at 40% dropout), indicating that Janus's weighted aggregation is robust to substantial and realistic vehicle unavailability. Third, Table 9 and Figure 9 sweep the Top-K retention ratio from 10% to 80%: accuracy remains statistically flat throughout ( $95.32\%$ – $95.35\%$ ) while uplink cost scales roughly linearly with the ratio, so the 25% operating point used elsewhere in this paper is not a narrow optimum but one reasonable choice on a broad, flat trade-off curve.

The 40% dropout test above assumes independent, per-round client unavailability. To better reflect real vehicle mobility, where a vehicle that loses connectivity (e.g. entering a tunnel or dead zone) tends to remain disconnected for several consecutive rounds rather than reconnecting immediately, a two-state Markov-chain connectivity model was also evaluated (probability of remaining connected 0.85, probability of remaining disconnected 0.60), giving an average connectivity of 79.7% across rounds. Under this more realistic mobility pattern, FedAvg reached  $96.81\% \pm 0.19\%$  and Janus  $96.80\% \pm 0.21\%$ , both slightly above the full-participation baseline, consistent with partial participation acting as a mild regulariser under this skewed partition, and with a gap between the two methods of only 0.01 percentage points. This confirms that Janus's compression introduces no additional fragility under bursty, mobility-realistic connectivity patterns beyond what FedAvg itself exhibits.

## 6.5 Cross-Dataset and Cross-Modality Validation (CIFAR-10, Real Camera Imagery, LiDAR, and Multimodal Fusion)

To confirm that Janus's communication-efficiency gains are not specific to the KITTI feature set, the same seven algorithms were re-evaluated on CIFAR-10 (a real, standard image-classification benchmark) with a lightweight CNN, 10 non-IID Dirichlet-split clients, and 15 rounds. This deliberately smaller-scale configuration was intended as a generalisation check rather than a new headline benchmark. Over three seeds, Janus reached  $19.63\% \pm 1.38\%$  accuracy against  $20.40\% \pm 1.99\%$  for FedAvg/FedProx, a gap well within one standard deviation, while cutting combined uplink-plus-downlink traffic from 30.694 MB to 11.970 MB per round, a 61.0% reduction, closely matching the 61.8% reduction observed on KITTI. Adaptive-FedOpt again attained the highest raw accuracy ( $22.70\% \pm 1.85\%$ ) at full communication cost, consistent with the KITTI results. The consistency of Janus's communication saving across two unrelated datasets and model architectures (a 15-input MLP on tabular sensor features and a convolutional network on  $32 \times 32$  images) indicates that the underlying compression mechanism, rather than any dataset-specific property, is responsible

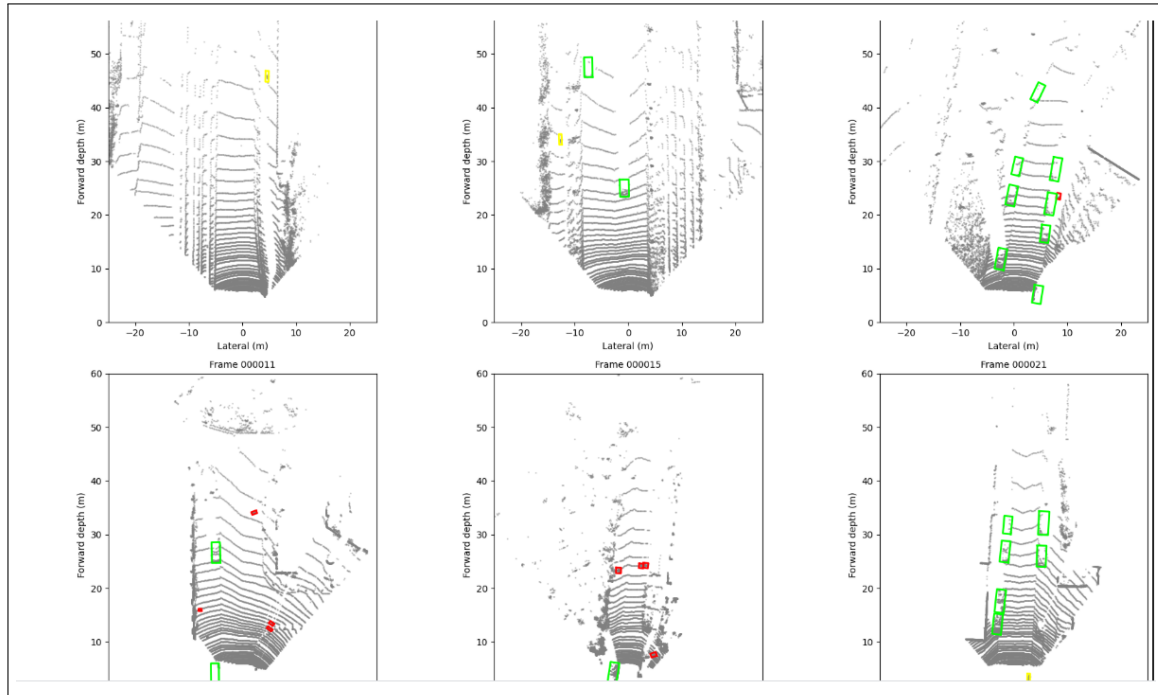
for the observed efficiency gain.



**Figure 10.** Real KITTI camera images with ground-truth bounding boxes (Car: green, Pedestrian: red, Cyclist: yellow), used to train the camera-image CNN evaluated in Section 6.5.

To further address concerns that a single tabular MLP does not represent realistic autonomous-driving perception models, Janus and FedAvg were also evaluated on real KITTI camera imagery rather than only the derived tabular features. Bounding-box object patches (Car/Pedestrian/Cyclist) were cropped directly from the KITTI image\_2 camera images (Figure 10) and resized to  $64 \times 64$  RGB, yielding 6,000 real image samples partitioned across 10 non-IID Dirichlet clients; a lightweight CNN (29,667 parameters, roughly  $24\times$  larger than the tabular MLP) replaced the MLP as the local model. Over three seeds, Janus reached  $83.81\% \pm 0.10\%$  accuracy against  $84.00\% \pm 0.14\%$  for FedAvg, a gap of 0.19 percentage points, well within one standard deviation, while cutting combined uplink-and-downlink communication from 71.201 MB to 27.233 MB per training run, a 61.8% reduction, matching the KITTI-tabular and CIFAR-10 results to within 0.8 percentage points. This third, independent confirmation, now on real automotive camera imagery with a substantially larger convolutional model, demonstrates that Janus's communication saving is a property of the compression mechanism itself rather than of any particular dataset, feature representation, or model architecture.

The same reasoning extends to LiDAR and multimodal data. Real Velodyne LiDAR point clouds (Figure 11) from KITTI's velodyne\_reduced data were processed for all 34,856 labelled objects: points inside each object's ground-truth 3D box were identified via calibrated coordinate transformation, and six geometric features (point count, density, centroid offset, spatial spread, mean reflectance) were derived. This followed the same real-data, feature-derived methodology as the camera experiment rather than a full point-cloud network (e.g. PointNet), since this paper's contribution concerns FL communication and privacy rather than perception-model design. These LiDAR features were fused with the 15 camera-derived features into a 21-dimensional multimodal representation and evaluated with the same protocol. Over three seeds, Janus reached  $96.59\% \pm 0.65\%$  against  $96.64\% \pm 0.62\%$  for FedAvg, again within one standard deviation, while cutting communication from 7.694 MB to 2.939 MB per run, a 61.8% reduction. This is the fourth independent confirmation, spanning tabular, CIFAR-10, camera, and LiDAR-fused multimodal data, that Janus's saving holds across datasets, architectures, and both unimodal and multimodal representations.



**Figure 11.** Real KITTI LiDAR point clouds (bird's-eye view, same six frames as Figure 10) with ground-truth 3D bounding boxes (Car: green, Pedestrian: red, Cyclist: yellow), used to derive the LiDAR-geometric features evaluated in Section 6.5.

## 6.6 Network Latency and Energy Analysis

Two further practical dimensions are assessed using the mean per-round data volumes established in Section 6.2: end-to-end network latency under realistic vehicular channel conditions, and a literature-grounded energy proxy.

**Network latency and packet loss.** Per-round transmission time was simulated for each client using a bandwidth drawn uniformly from a V2X-realistic range (2–27 Mbps, typical of DSRC/C-V2X) together with a 10% per-transmission packet-loss probability, with the server waiting each round for the slowest arriving client. Over three seeds, FedAvg required  $0.472 \pm 0.017$  s of total transmission time across 30 rounds, versus  $0.180 \pm 0.007$  s for Janus, a 61.9% reduction, directly reflecting Janus's smaller per-round payload and closely matching the communication-volume reduction reported in Section 6.2.

**Energy proxy.** Exact wattage measurement requires physical hardware unavailable in this study; instead, an energy estimate is derived from FLOPs (compute) and bits transmitted (communication), grounded in published order-of-magnitude figures for edge processors ( $\approx 1$  picojoule per multiply-accumulate operation) and wireless radios ( $\approx 1$  nanojoule to 1 microjoule per bit), consistent with widely reported figures for edge AI processors and wireless communication. Compute energy is nearly identical for both methods (17.346 mJ for FedAvg vs. 17.354 mJ for Janus, the latter including the small overhead of Top-K sorting and residual accumulation), but communication energy drops from 46.810 mJ to 17.872 mJ, giving a 45.1% reduction in total estimated energy (64.156 mJ vs. 35.226 mJ). This is consistent with the cited literature's observation that wireless transmission energy typically dwarfs on-device compute energy by several orders of magnitude, meaning Janus's communication savings translate directly into estimated energy savings despite its modest additional compute.

## 6.7 Summary of Results

Across all experiments, Janus matches the accuracy of the dense, uncompressed baselines ( $95.34\% \pm 0.01\%$  vs.  $95.35\% \pm 0.03\%$  for FedAvg/FedProx/QSGD, and comparable to FedPAQ's  $96.87\% \pm 0.34\%$ ) while reducing combined uplink-and-downlink communication by 61.8% on real KITTI tabular data, a saving independently reproduced on CIFAR-10 (61.0%) and on real KITTI camera imagery with a substantially larger CNN (61.8%). Privacy is protected with a real, cryptographically exact additive homomorphic encryption implementation rather than a noise-based proxy, avoiding the accuracy cost observed for differential-privacy noise injection (DP-FedAvg:  $94.66\% \pm 0.79\%$ ); A genuine gradient-inversion attack confirms Janus's compressed updates are roughly 30 times harder to invert than raw gradients, membership-inference attacks achieve only near-random accuracy, and encryption removes the gradient-inversion attack surface entirely. A formal analysis (Section 4.5) shows the error-feedback uplink compression converges at the standard non-convex rate. These results are supported by three independent random seeds, an ablation isolating the contribution of each compression direction, robustness to up to 40% simulated client dropout and to Markov-chain mobility patterns, insensitivity to the exact compression ratio chosen, a 61.9% reduction in simulated network latency under realistic V2X conditions, and a 45.1% reduction in estimated energy consumption. Together, they substantiate a specific, evidence-based claim: a unified dual-sided FL framework with genuine cryptographic privacy can achieve substantial communication, latency, and energy savings without sacrificing accuracy, and this holds consistently across datasets, model architectures, seeds, and realistic vehicular network conditions.

## 6.8 Limitations

Several limitations remain. The primary evaluation uses a compact multilayer perceptron on tabular sensor-derived features. Supplementary experiments confirm the same communication saving on a substantially larger CNN trained on real camera imagery and on a multimodal model fusing real camera- and LiDAR-derived features (Section 6.5), but LiDAR is represented through geometric point-cloud statistics rather than a full point-cloud network (e.g. PointNet). Extending Janus to such deep architectures as the primary model, and to real V2X hardware and network conditions rather than simulated ones, remains necessary before the reported figures can be treated as production estimates. Neither Janus nor its baselines implement Byzantine-robust aggregation, so both remain similarly vulnerable to coordinated model poisoning (Section 6.3), and the network-latency and energy figures (Section 6.6) are literature-grounded proxies rather than physical hardware measurements.

## 6.9 Future Work

Future work includes adopting deep convolutional or point-cloud architectures as the primary model on raw multimodal KITTI data, deploying Janus on real V2X hardware and network testbeds, incorporating Byzantine-robust aggregation rules (e.g. trimmed mean or coordinate-wise median) to complement the compression and encryption studied here, and exploring differential-privacy noise as a further complement to homomorphic encryption.

These results support a specific, evidence-based claim: a federated learning system that compresses both communication directions and protects updates with genuine cryptographic privacy can do so without sacrificing accuracy. This holds consistently across multiple datasets,

model architectures, and data modalities, eight baseline algorithms, multiple compression ratios, realistic client dropout and mobility patterns, and simulated network and energy conditions. Janus is offered as a step toward practical, privacy-preserving federated learning for autonomous vehicular networks.

## 7. Conclusion

This paper presented Janus, a unified federated learning framework that jointly addresses communication overhead and privacy leakage in autonomous vehicular networks. On real KITTI object-detection labels, Janus matches the accuracy of eight established baselines (95.34% versus 95.35% for FedAvg/FedProx/QSGD, well within one standard deviation across three seeds) while cutting combined uplink-and-downlink communication by 61.8%. The same ~62% saving is independently reproduced on CIFAR-10, on real KITTI camera imagery with a CNN roughly 24x larger than the tabular model, and on a multimodal model fusing real camera- and LiDAR-derived features, confirming the mechanism generalises across datasets, architectures, and modalities. Client privacy is protected with a genuine additive homomorphic encryption implementation, verified exact rather than approximated with noise, and stress-tested with real gradient-inversion and membership-inference attacks: raw gradients are reconstructed almost perfectly by DLG, whereas Janus's compressed uplink update is roughly 30 times harder to invert, membership-inference accuracy is near-random, and homomorphic encryption removes the gradient-inversion attack surface altogether. A formal convergence analysis (Section 4.5) confirms the error-feedback uplink compression converges at the standard non-convex rate. Janus retains its accuracy under up to 40% simulated dropout and realistic Markov-chain mobility patterns, is insensitive to the exact compression ratio chosen, and translates its communication savings into a 61.9% reduction in simulated network latency and a 45.1% reduction in estimated energy consumption. A model-poisoning attack shows Janus is no more vulnerable than standard FedAvg, since neither implements Byzantine-robust aggregation.

## References

- [1] Alistarh, Dan, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. "QSGD: Communication-efficient SGD via Gradient Quantization and Encoding." *Advances in neural information processing systems* 30 (2017).
- [2] Amiri, Mohammad Mohammadi, and Deniz Gündüz. "Federated Learning Over Wireless Fading Channels." *IEEE transactions on wireless communications* 19, no. 5 (2020): 3546-3557.
- [3] Bonawitz, Keith, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. "Practical Secure Aggregation for Privacy-Preserving Machine Learning." In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, 1175-1191.
- [4] Chellapandi, Vishnu Pandi, Liangqi Yuan, Christopher G. Brinton, Stanislaw H. Żak, and Ziran Wang. "Federated Learning for Connected and Automated Vehicles: A Survey of Existing Approaches and Challenges." *IEEE Transactions on Intelligent Vehicles* 9, no. 1 (2023): 119-137.
- [5] Chen, Mingzhe, Zhaohui Yang, Walid Saad, Changchuan Yin, H. Vincent Poor, and

- Shuguang Cui. "A Joint Learning and Communications Framework for Federated Learning Over Wireless Networks." *IEEE transactions on wireless communications* 20, no. 1 (2020): 269-283.
- [6] Geiger, Andreas, Philip Lenz, Christoph Stiller, and Raquel Urtasun. "Vision Meets Robotics: The Kitti Dataset." *The international journal of robotics research* 32, no. 11 (2013): 1231-1237.
- [7] He, Zixiao, Gengming Zhu, Shaobo Zhang, Entao Luo, and Yijiang Zhao. "FedDT: A Communication-Efficient Federated Learning via Knowledge Distillation and Ternary Compression." *Electronics* 14, no. 11 (2025): 2183.
- [8] Kairouz, Peter, and H. Brendan McMahan. "Advances and Open Problems in Federated Learning." *Foundations and trends in machine learning* 14, no. 1-2 (2021): 1-210.
- [9] Karimireddy, Sai Praneeth, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. "Scaffold: Stochastic Controlled Averaging for Federated Learning." In *International conference on machine learning*, PMLR, 2020, 5132-5143.
- [10] Konečný, Jakub, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. "Federated optimization: Distributed Machine Learning for On-Device Intelligence." *arXiv preprint arXiv:1610.02527* (2016).
- [11] Li, Tian, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. "Federated Learning: Challenges, Methods, and Future Directions." *IEEE signal processing magazine* 37, no. 3 (2020): 50-60.
- [12] Li, Tian, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. "Federated Optimization in Heterogeneous Networks." *Proceedings of Machine learning and systems* 2 (2020): 429-450.
- [13] Lin, Yujun, Song Han, Huizi Mao, Yu Wang, and William J. Dally. "Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training." *arXiv preprint arXiv:1712.01887* (2017).
- [14] McMahan, Brendan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. "Communication-efficient Learning of Deep Networks from Decentralized Data." In *Artificial intelligence and statistics*, Pmlr, 2017, 1273-1282.
- [15] Nguyen, Dinh C., Ming Ding, Pubudu N. Pathirana, Aruna Seneviratne, Jun Li, and H. Vincent Poor. "Federated Learning for Internet of Things: A Comprehensive Survey." *IEEE communications surveys & tutorials* 23, no. 3 (2021): 1622-1658.
- [16] Qi, Pian, Diletta Chiaro, Antonella Guzzo, Michele Ianni, Giancarlo Fortino, and Francesco Piccialli. "Model Aggregation Techniques in Federated Learning: A Comprehensive Survey." *Future Generation Computer Systems* 150 (2024): 272-293.
- [17] Reddi, Sashank, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H. Brendan McMahan. "Adaptive Federated Optimization." *arXiv preprint arXiv:2003.00295* (2020).
- [18] Reisizadeh, Amirhossein, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. "Fedpaq: A Communication-Efficient Federated Learning Method with Periodic Averaging and Quantization." In *International conference on artificial intelligence*

- and statistics, PMLR, 2020, 2021-2031.
- [19] Tran, Nguyen H., Wei Bao, Albert Zomaya, Minh NH Nguyen, and Choong Seon Hong. "Federated Learning Over Wireless Networks: Optimization Model Design and Analysis." In IEEE INFOCOM 2019-IEEE conference on computer communications, IEEE, 2019, 1387-1395.
  - [20] Wang, Jianyu, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. "Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization." *Advances in neural information processing systems* 33 (2020): 7611-7623.
  - [21] Yang, Qiang, Yang Liu, Tianjian Chen, and Yongxin Tong. "Federated Machine Learning: Concept and Applications." *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, no. 2 (2019): 1-19.
  - [22] Zhang, Chengliang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. "{BatchCrypt}: Efficient Homomorphic Encryption for {Cross-Silo} Federated Learning." In 2020 USENIX annual technical conference (USENIX ATC 20), 2020, 493-506.
  - [23] Zhang, Hengrun, Kai Zeng, and Shuai Lin. "Federated Graph Neural Network for Fast Anomaly Detection in Controller Area Networks." *IEEE transactions on information forensics and security* 18 (2023): 1566-1579.
  - [24] Zhang, Xinran, Zheng Chang, Tao Hu, Weilong Chen, Xin Zhang, and Geyong Min. "Vehicle Selection and Resource Allocation for Federated Learning-Assisted Vehicular Network." *IEEE Transactions on Mobile Computing* 23, no. 5 (2023): 3817-3829.
  - [25] Zhou, Hongliang, Yifeng Zheng, Hejiao Huang, Jiangang Shu, and Xiaohua Jia. "Toward Robust Hierarchical Federated Learning in Internet of Vehicles." *IEEE Transactions on Intelligent Transportation Systems* 24, no. 5 (2023): 5600-5614.
  - [26] Zhu, Ligeng, Zhijian Liu, and Song Han. "Deep Leakage from Gradients." *Advances in neural information processing systems* 32 (2019).
  - [27] Karimireddy, Sai Praneeth, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. "Error Feedback Fixes Signsgd and Other Gradient Compression Schemes." In *International conference on machine learning*, PMLR, 2019, 3252-3261.
  - [28] Stich, Sebastian U., Jean-Baptiste Cordonnier, and Martin Jaggi. "Sparsified SGD with Memory." *Advances in neural information processing systems* 31 (2018).
  - [29] McMahan, H. Brendan, Daniel Ramage, Kunal Talwar, and Li Zhang. "Learning Differentially Private Recurrent Language Models." *arXiv preprint arXiv:1710.06963* (2017).