Centralized vs. Decentralized Federated Learning for Autonomous Driving

Amin Armat
College of Engineering
The Pennsylvania State University
State College, PA
aja6920@psu.edu

Varun Damarla
College of Engineering
The Pennsylvania State University
State College, PA
vfd5089@psu.edu

Sankalp Subhendu
College of Engineering
The Pennsylvania State University
State College, PA
svs7191@psu.edu

***Abstract*—Autonomous driving systems rely heavily on machine learning models to process sensory data for real-time decision-making. Traditional centralized machine learning, while accurate, poses significant privacy risks and struggles with scalability in the pursuit of continuous learning. This paper evaluates the application of federated learning to predict steering angles from visual inputs while preserving data privacy. The research compares centralized and decentralized federated learning frameworks, focusing on their performance, scalability, and application in real-world environments. These findings highlight the potential of federated learning to enable scalable, privacy-preserving, and robust autonomous driving solutions in new, dynamic environments.**

1. Introduction

The rise of autonomous vehicles has revolutionized modern transportation, relying on advanced machine learning systems to process sensory data for navigation and decision-making. Traditional centralized learning paradigms, which aggregate massive datasets to train global models, introduce critical privacy risks and incur substantial communication costs. Federated learning emerges as a compelling alternative, enabling collaborative model training across distributed edge devices while preserving data locality.

This research evaluates the application of federated learning in steering angle prediction tasks using the SullyChen Driving Dataset collection. Drawing inspiration from Zhang et al. (2021), this paper compares centralized and decentralized federated learning frameworks while employing Dual-Stream and Spatio-Temporal neural architectures. Main goals include assessing model performance, scalability, and real-time feasibility across the two different federated learning frameworks.

2. Background & Related Work

2.1. Federated Learning for Autonomous Driving

Federated learning decentralizes model training by aggregating locally trained updates instead of raw data [5]. Centralized federated learning utilizes a parameter server to coordinate updates, while decentralized federated learning relies on peer-to-peer communication, enhancing scalability and fault tolerance. Studies such as Nguyen et al. (2022) and Zhang et al. (2021) highlight federated learning's potential in autonomous driving by addressing privacy concerns and mitigating bandwidth constraints.

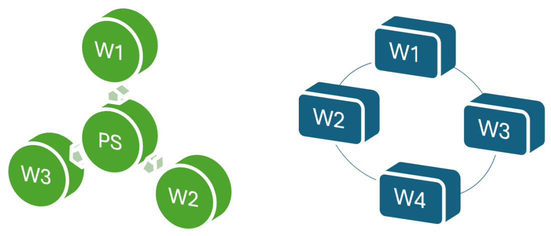

Figure 1. On the left, a centralized federated learning framework. On the right, an example decentralized framework with four workers.

2.2. Model Architectures for Steering Angle Prediction

Dual-stream models process RGB frames and optical flow streams independently, offering robustness in scenarios requiring motion understanding [6]. In contrast, spatio-temporal models employ 3D convolutions to capture temporal dependencies in consecutive frames [7]. Prior research establishes the suitability of these architectures for federated learning, attributing their efficacy to compact design and real-time inference capabilities. Recent studies have also explored hybrid architectures, such as combining recurrent layers with convolutional designs, to improve temporal modeling without sacrificing computational efficiency.

3. Methodology

3.1. Dataset and Preprocessing

This study utilized the SullyChen Driving Dataset, comprising around 110,000 labeled images reflecting diverse driving conditions [3]. The dataset was partitioned into 45,500 images for base model pretraining and 63,800 images for the federated learning experiments. Images were resized to 256×455 pixels and normalized to ensure uniform input across models.

3.2. Model Architectures

Two model architectures were implemented, designed specifically to handle the challenges of steering angle prediction:

Dual-Stream Model

This architecture consists of two parallel convolutional streams. The first processes three contiguous frames of raw RGB images, utilizing two convolutional layers with kernel sizes of 3x3, followed by max-pooling layers. The second stream utilizes optical flow values, calculated via OpenCV's Gunnar-Farneback algorithm, using the three consecutive frames and processes them with identical convolutional layers. Outputs from both streams are concatenated and passed through a fully connected network with three layers of sizes 256, 64, and 1. Dropout layers with a rate of 0.2 were added to mitigate overfitting, and ReLU activation was used to introduce non-linearity. This model's lightweight structure emphasizes computational efficiency, making it suitable for resource-constrained environments.

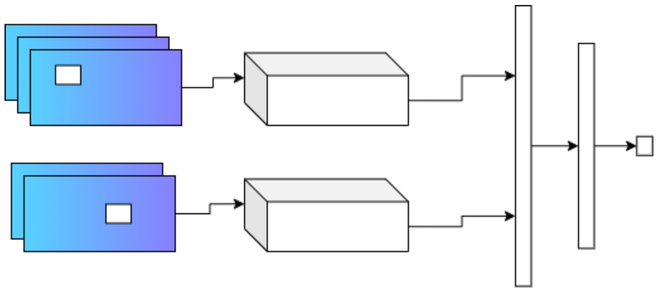
Figure 2. A simplified view of the dual stream model architecture. The top stream processes the three consecutive RGB frames in a CNN, while the bottom processes the two optical flow maps computed from the three frames in a CNN. The results are concatenated into a fully connected neural network to output a single driving angle.

Spatio-Temporal Model

This architecture applies 3D convolutions to three consecutive RGB frames to capture both spatial and temporal dependencies. It begins with two convolutional layers using 3D kernels of size 3x3x3, followed by 3D max-pooling layers to down-sample the input while retaining temporal information. The resulting features are flattened and passed through a fully connected network with layers of sizes 512, 128, and 1. Dropout with a rate of 0.2 was incorporated after each fully connected layer to regularize training.

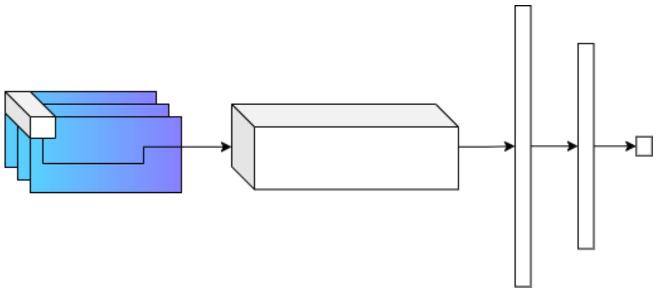
Figure 3. A simplified view of the spatio-temporal model architecture. The three consecutive RGB frames are processed by a 3D CNN and the results are fed into a fully connected neural network to output a single driving angle.

3.3. Example Visualization of Model Inputs

Both the models take in the three consecutive frame inputs as in Figure 4, with only the Dual Stream model taking the additional optical flow map as input. The Spatio-Temporal model does without the optical flow input due to the nature of the model and its ability to capture spatial dependence within the consecutive frames. It should be mentioned that the optical flow is calculated as: $O_t = f(A_{t-1}, A_t)$. Using only the three consecutive frames, the optical flow is calculated, and the result is two optical flow frames that are passed into the Dual Stream model. For N consecutive RGB frames $(A_{t-(n-1)}, A_{t-(n-2)}, \ldots, A_t)$, there are N-1 optical flow frames passed to the model $(O_{t-(n-2)}, O_{t-(n-3)}, \ldots, O_t)$.

3.4. Federated Learning Frameworks

As shown in Figure 5, centralized federated learning utilizes a server to aggregate model updates from vehicles and distribute the global model. This framework offers simplicity but is prone to bottlenecks in high-density network scenarios.

Figure 5 also illustrates an outline of the decentralized federated learning framework used in our study. Decentralized federated learning facilitates direct inter-vehicle communication, enabling peer-to-peer exchange of model updates. This approach enhances robustness and reduces reliance on centralized infrastructure, making it ideal for distributed vehicular networks. In a real-world environment, workers would connect and exchange model weights with nearby neighbors. Given the prior datasets at hand, workers needed to simulate this connection process during each aggregation step. To achieve this, each worker would randomly connect to between one and four of its peer neighbors each round. This encapsulates the idea of changing environments and the location of each worker at a given time.

3.5. Training and Evaluation

The training process was divided into two phases:

Base Model Training

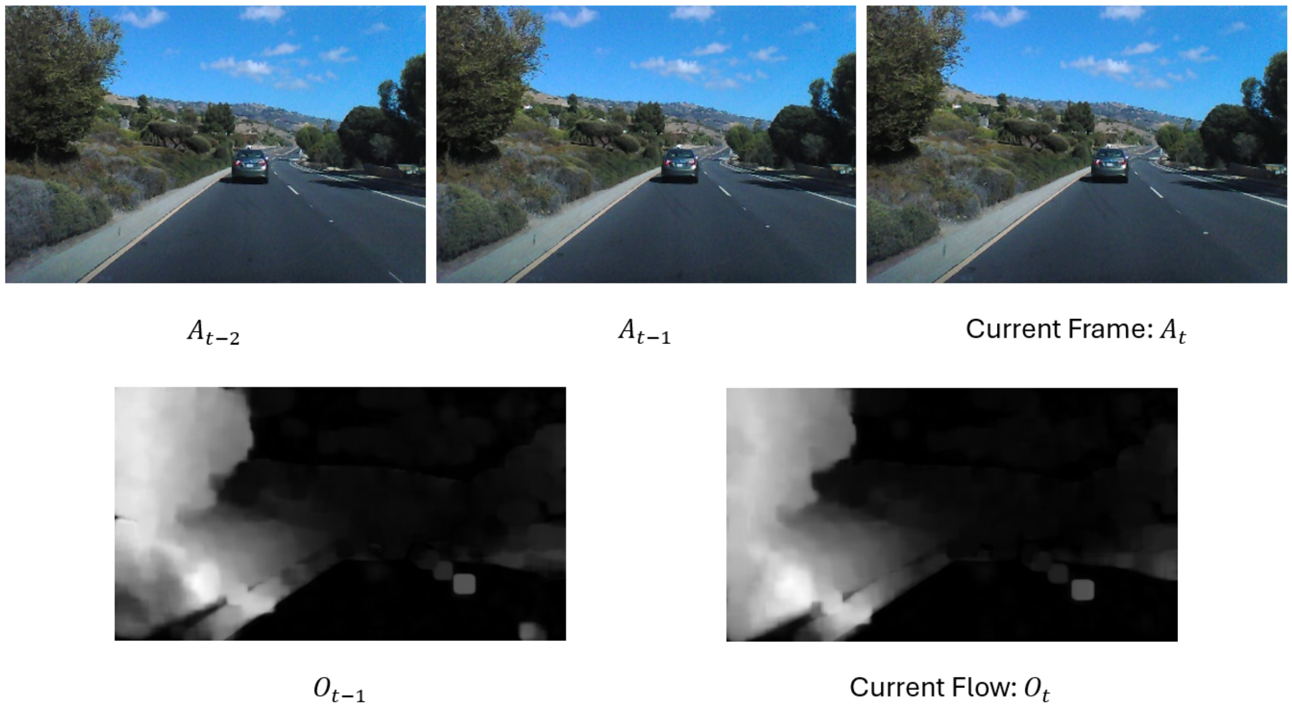

Figure 4. Example visualizations of both the frame inputs and optical flow maps that are fed into the model

Each architecture (Dual-Stream and Spatio-Temporal) was trained on the first dataset of the SullyChen driving dataset collection for 40 epochs to determine which model was better suited for federated learning. The training used the following hyperparameters:

- Batch size: 64
- Learning rate: $2 * 10^{-4}$
- Optimizer: Adam with weight decay of 10^{-5}
- Loss function: Mean Squared Error (MSE)

Federated Learning Training

Federated learning experiments were conducted over 50 training rounds with the following setup utilizing the second dataset of the SullyChen dataset collection:

- Batch size per worker: 32
- Learning rate: $2 * 10^{-5}$
- Optimizer: Adam with weight decay of 10^{-5}
- Number of workers: Varied (4, 8, 16)
- Loss function: Mean Squared Error (MSE)
- Training rounds per worker per global iteration: 5

For the federated learning training, each worker, regardless of the total number of workers in the experiment, was trained on 1,700 samples from their individual training dataset. This was done for consistency across the variable number of workers in the experiment. This choice was additionally made to capture a real-world scenario where workers wouldn't have the capability to train for long segments of time on all the raw sensor data they have captured. A further breakdown of the dataset is shown in Figure 6.

Analysis metrics included training/testing loss and Root Mean Squared Error (RMSE) on worker test data. Results were also analyzed qualitatively by plotting ground truth steering angles vs. models' predicted angle.

Base models and FL models were trained on an RTX A4500 GPU as provided by the Penn State CSE Lab machines for parallelization purposes. In a real-world setting, hardware for training the federated learning models would vary, although results would remain similar as the GPU provides faster training times.

4. Results

4.1. Base Model Training

During the base model training phase, both the Dual-Stream and Spatio-Temporal models were trained for 40 epochs. Training loss and test loss were recorded per epoch to evaluate convergence and generalization, as shown in Figure 7.

Key metrics from the evaluation are summarized below in Table 1.

Analysis of test loss plots showed that Dual-Stream demonstrated faster convergence and significantly stabler performance across epochs compared to Spatio-Temporal.

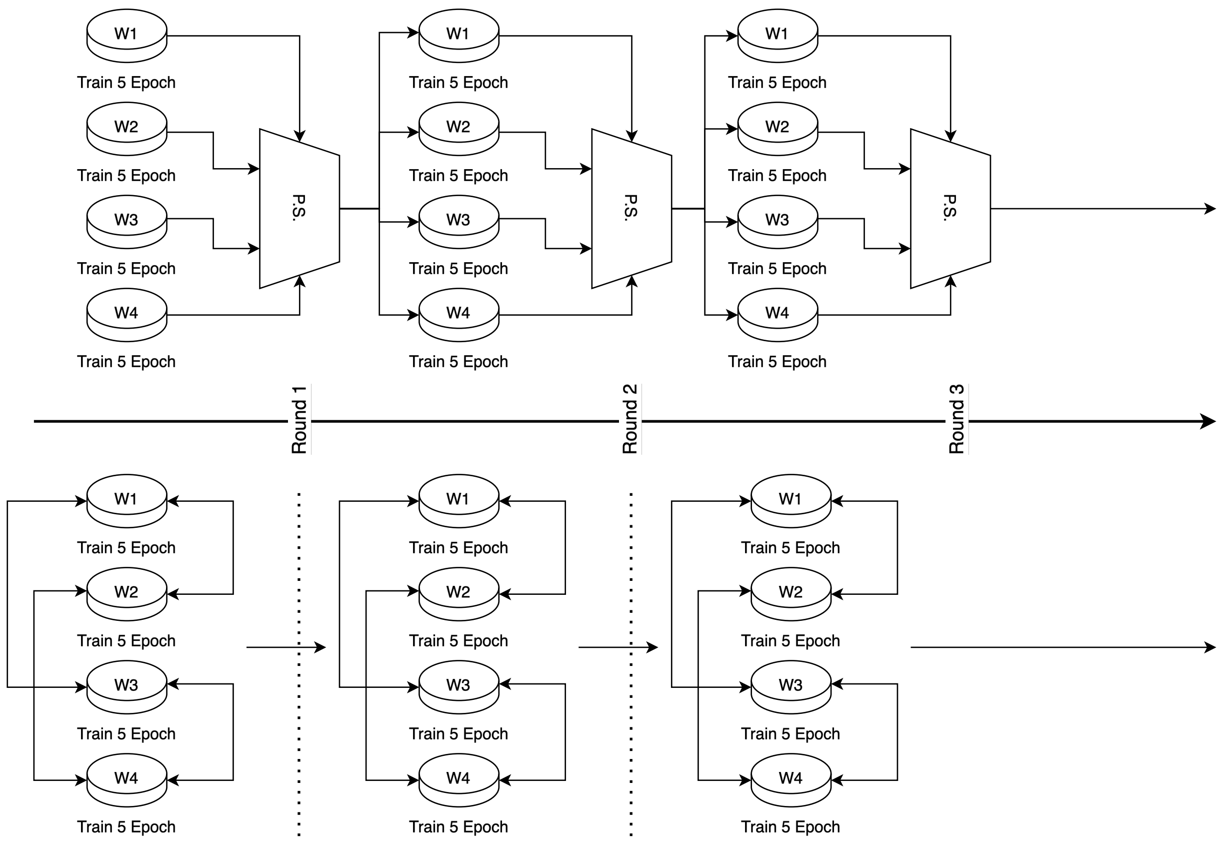

Figure 5. An outline of the federated learning training section. The top portion illustrates the centralized FL situation, while the bottom outlines the simultaneous decentralized FL situation.

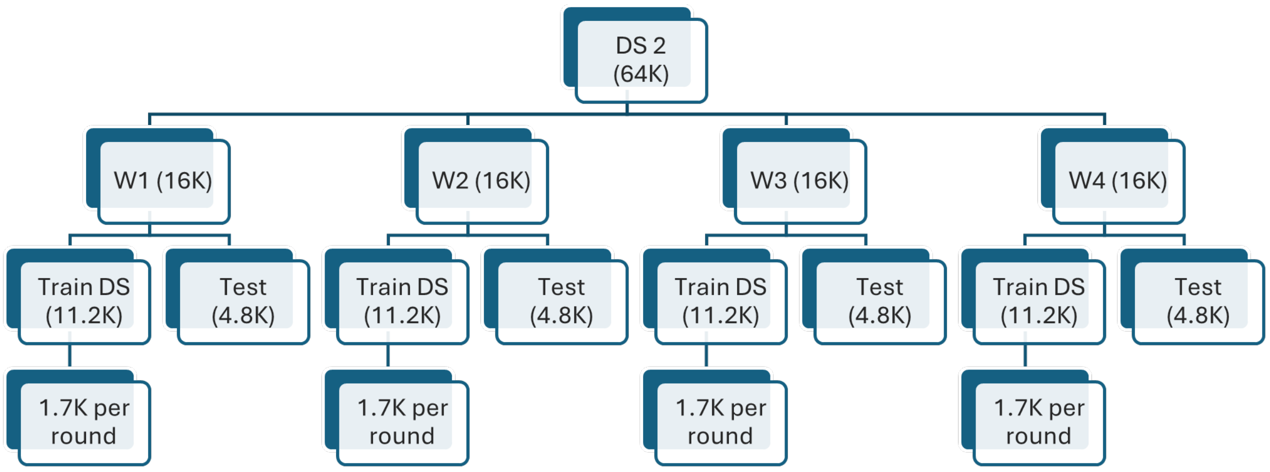

Figure 6. A breakdown of how the second dataset is used for the federated learning training per worker

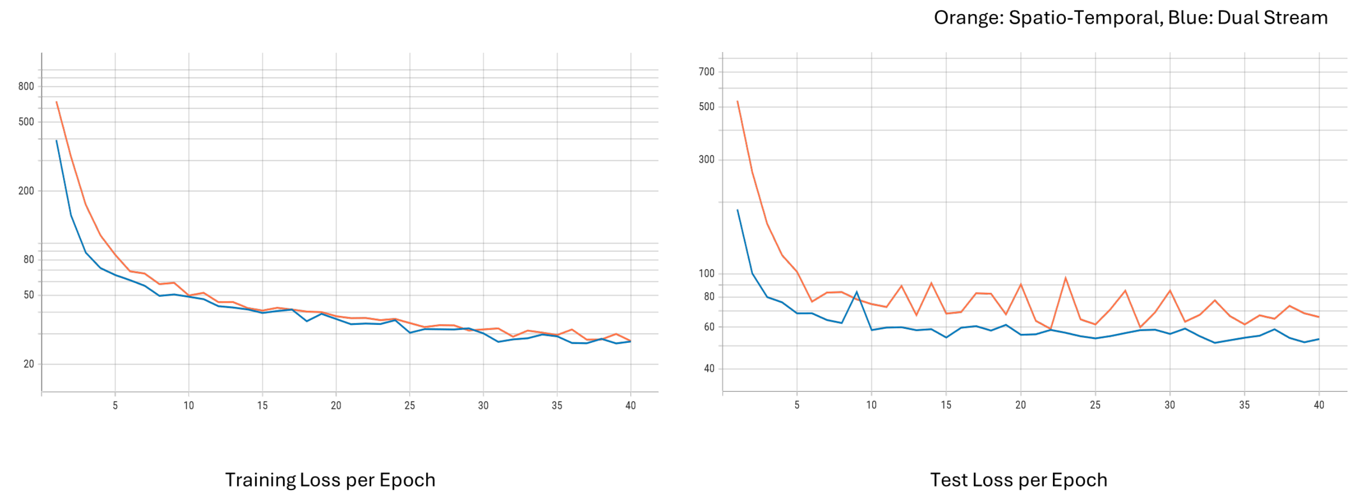

Figure 7. Training and testing loss per epoch, comparing both the Dual Stream and Spatio-Temporal models

TABLE 1. MODEL EVALUATION METRICS ($\sim$ 6400 UNSEEN FRAME INPUTS) W/ CPU INFERENCE

Model	Dual Stream	Spatio-Temporal
RMSE	**8.4729**	10.8527
Avg. Inference Time	**0.0061 s**	0.0111 s
Avg. FPS	**162.66**	90.15
Training Time (40 ep)	$\sim$ 5 hr	$\sim$ **2.5 hr**
Model Size	**81 MB**	171 MB

Additionally, Dual-Stream's smaller size and faster inference time (Table 1) made it more suitable for deployment in federated learning scenarios. These advantages led to its selection for the subsequent federated learning experiments.

4.2. Federated Learning Training

With the Dual-Stream model, across both training and testing loss for four and eight workers, decentralized FL outperforms centralized FL (Figure 8). Decentralized learning achieves faster convergence and a lower overall loss, but centralized learning exhibits less variance and sees a smoother and more stable convergence rate.

The RMSE for both frameworks is shown below in Table 2. Since there is no direct way to assess model error for the decentralized framework, two approaches were tested to approximate the average error of the decentralized framework. One approach used the RMSE calculated for each worker model and then averaged, while the other approach took a final federated average (FedAvg.) of all the individual workers, resulting in a new global "decentralized" model of which the RMSE was tested.

As shown in Table 2, both decentralized testing approaches had a lower average RMSE than the centralized model when trained with four workers. In comparison, when trained with eight workers, the models demonstrate

TABLE 2. RMSE COMPARISON ACROSS FL ARCHITECTURES WITH DIFFERENT WORKER CONFIGURATIONS

FL Architecture	4 workers	8 workers
Centralized	5.3058	5.4740
Decentralized (Avg.)	4.9286	5.5273
Decentralized (FedAvg.)	4.9286	5.1013

a noticeably smaller gap in RMSE. This highlights how, regardless of the federated learning framework used, the models converged to result in a performance in this given environment.

An interesting point to mention here is how in the decentralized FL architecture, there is a noticeable difference in RMSE between both testing approaches in the eight worker scenario. This shows that for the decentralized case, there are definitive personalized models that perform worse in their testing environment as compared to the other models. When the FedAvg. algorithm is applied in this scenario, they all aggregate together and the model that didn't perform better gets washed out, which decreases the average loss.

Comparing the model results qualitatively, as shown in Figure 9, there are minimal differences between the separate architectures. The two figures on the top showcase the ground truth angle vs. the predicted angles for centralized four workers, decentralized four workers with a final FedAvg., centralized eight workers, and decentralized eight workers with a final FedAvg. The separate figures on the bottom show the individual predictions for decentralized models without having to aggregate them all. These results confirm that regardless of the federated learning framework, or testing approach for the decentralized models, the models converged to a near-optimal solution for the given training environment.

Overall, decentralized workers outperformed centralized workers, likely due to the limited number of workers in the experiment. With fewer workers (e.g., four), decentralized

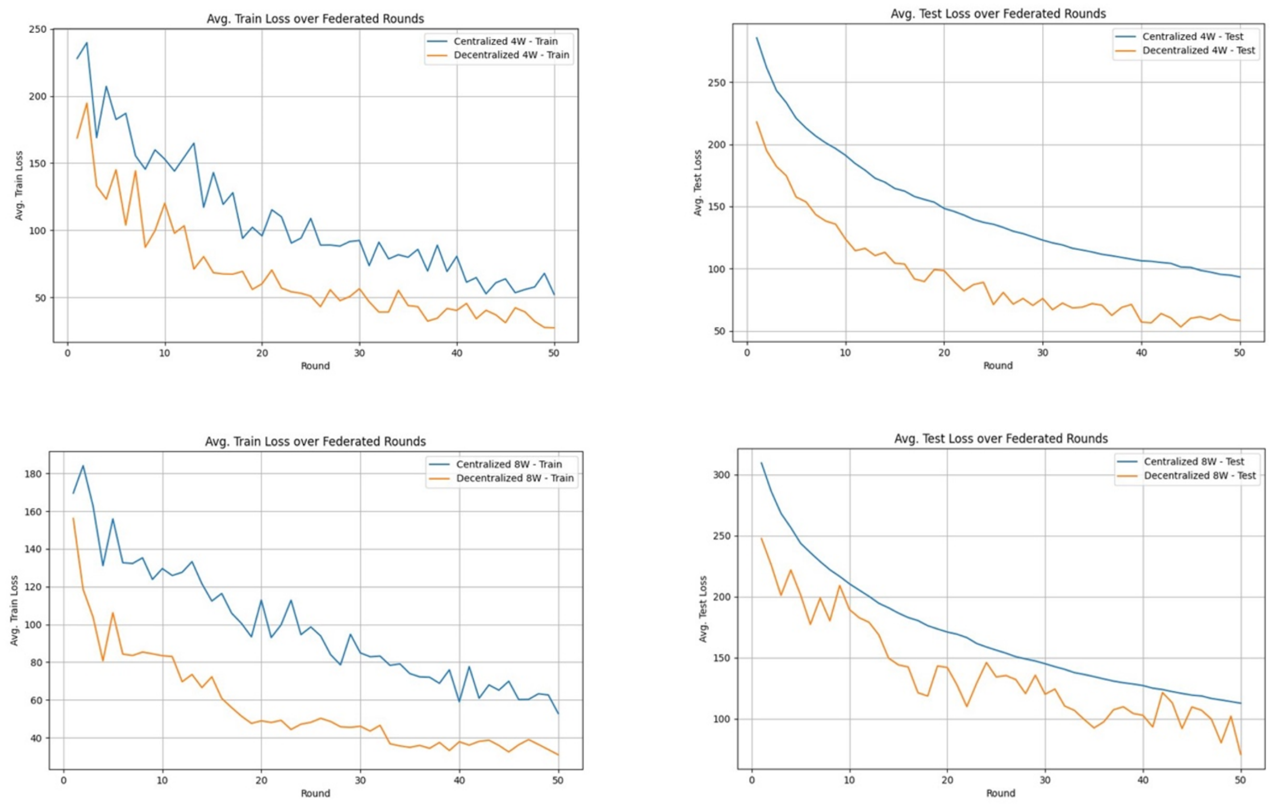

Figure 8. Training and testing loss for both decentralized and centralized FL across both four and eight workers

learning benefits from more localized updates, leading to a faster convergence. The results show that decentralized learning achieved a lower RMSE of 4.9286 compared to centralized learning, which achieved an RMSE of 5.3058, in the four worker setup. However, as the number of workers increased to eight, the performance gap narrowed, suggesting that centralized learning could scale better with more workers, while decentralized learning excels in smaller-scale scenarios.

4.3. Scalability

Federated learning significantly improves traditional machine learning practices by enabling collaborative model training across distributed devices without sharing raw data. This paradigm allows for continuous learning in dynamic environments, which is particularly critical in autonomous driving scenarios. Unlike traditional centralized machine learning, where data from edge devices must be aggregated to a central server, federated learning keeps data localized, reducing privacy concerns and improving scalability.

Centralized federated learning relies on a central parameter server to aggregate model updates and distribute the global model to workers. While this approach simplifies monitoring and global synchronization, it introduces a bottleneck as the number of workers increases. In contrast, decentralized federated learning distributes the aggregation workload across workers, allowing model updates to occur peer-to-peer. This alleviates pressure on the main parameter server but shifts the burden to individual workers, which must download and aggregate their neighbors' weights.

Experiments show that federated learning's efficiency improves with an increasing number of workers. Larger worker participation brings diverse datasets into the training process, enhancing the global model's ability to generalize across a wide range of conditions.

4.4. Real-World Feasibility

Deploying federated learning in real-world autonomous driving scenarios involves addressing critical infrastructure challenges and operational complexities.

Centralized federated learning infrastructure poses the following benefits and barriers:

- Monitoring and Debugging: Centralized federated learning offers easier monitoring and diagnostics due to the single point of aggregation. Any issues in model convergence or training can be promptly identified at the server level.
- Single Point of Failure: Reliance on a central server introduces a potential single point of failure. Any

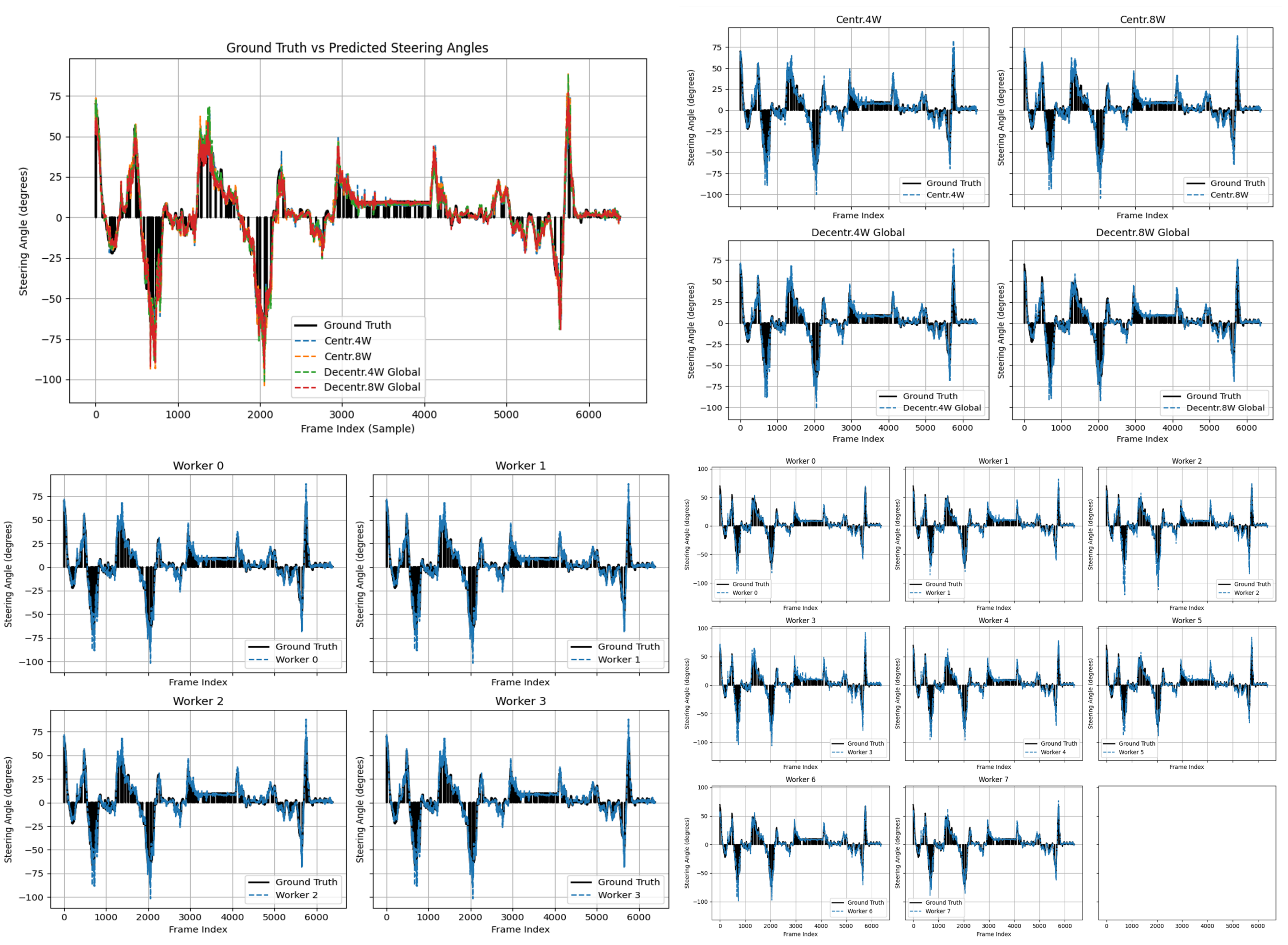

Figure 9. Qualitative metrics comparing the ground truth angle vs. the predicted angle for the model architectures

server downtime could halt the entire training process.

- Scalability Bottlenecks: As the number of workers grows, the server must handle increasingly large volumes of model weights. This can lead to significant communication delays and bandwidth congestion, particularly as the number of workers scales to thousands.

Decentralized federated learning poses the following benefits and barriers:

- Monitoring Complexity: Decentralized federated learning, while more robust, is harder to monitor and debug due to the lack of a central aggregation point. Effective implementation requires careful design of peer-to-peer communication protocols to ensure transparency and traceability.
- Peer-to-Peer (P2P) Communication: Decentralized federated learning eliminates the need for a central server by enabling direct communication between vehicles. This makes it more resilient to failures and adaptable to dynamic environments. For instance, vehicles can use vehicle-to-vehicle (V2V) communication to share model updates with nearby peers.
- Personalization: Decentralized federated learning allows for more personalized models by tailoring training to local data distributions. This is particularly beneficial in heterogeneous driving environments, where vehicles encounter vastly different road conditions and scenarios.

Key infrastructure considerations when choosing between the federated learning frameworks:

- Dynamic Peer Selection: In decentralized federated learning, vehicles must intelligently decide which peers to communicate with based on their proximity, relevance, and data reliability.
- Network Robustness: Ensuring reliable peer-to-peer communication in varying environmental conditions, such as urban areas versus highways, is critical.
- Scalability Challenges: For centralized federated learning, scaling beyond a certain number of workers requires robust server infrastructure capable of handling massive data exchanges. For decentralized fed-

erated learning, managing dynamic topologies and ensuring timely synchronization are key challenges.

By addressing these infrastructure challenges, federated learning can transition from experimental setups to real-world deployment, unlocking its potential for scalable, privacy-preserving autonomous driving solutions.

5. Conclusion

This paper explored the application of federated learning frameworks in the context of autonomous driving, with a focus on centralized and decentralized approaches. The results highlight the advantages of decentralized federated learning in enhancing scalability and robustness by leveraging peer-to-peer communication for model aggregation. By distributing computational loads across workers, decentralized frameworks reduce dependency on a single server, which is particularly beneficial in dynamic and diverse driving environments.

At the same time, centralized federated learning retains value in settings where centralized monitoring and debugging are prioritized, though its scalability limitations and susceptibility to single points of failure remain challenges. The comparative analysis underscores the adaptability of federated learning in accommodating the needs of autonomous systems, enabling continuous learning and maintaining privacy. However, implementing federated learning in real-world scenarios necessitates overcoming infrastructure challenges, including network robustness, dynamic peer selection, and the management of computational heterogeneity.

Future work should focus on integrating advanced model architectures and improving aggregation techniques (weighted FedAvg) to further enhance federated learning's scalability and efficiency. Additionally, increasing the number of workers should be a central focus to analyze whether decentralized still outperforms centralized learning in a larger scale system. To accomplish this, simulated test environments should be used (e.g., CARLA, Unreal) and trained in to minimize potential environment overfitting, simulate additional real-world driving conditions, and gather additional training data besides potential dataset augmentation methods.

References

[1] Zhang, H., Bosch, J., & Olsson, H. H. (2021). End-to-End Federated Learning for Autonomous Driving Vehicles.

[2] Nguyen, A., et al. (2022). Deep Federated Learning for Autonomous Driving.

[3] SullyChen Driving Dataset. (n.d.). Available on Github

[4] Fu, Y., et al. (2024). A Secure Personalized Federated Learning Algorithm for Autonomous Driving.

[5] Kairouz, P., McMahan, H. B., Avent, B., et al. (2021). Advances and Open Problems in Federated Learning. Foundations and Trends in Machine Learning

[6] Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition

[7] Tran, D., Bourdev, L., Fergus, R., Torresani, L., & Paluri, M. (2015). Learning Spatiotemporal Features with 3D Convolutional Networks.