Federated Learning for Autonomous Driving

Homayoun Afshari[1], Marcelo Bastos Lopes Ferreira[2], Gustavo Nicoletti Rosa[3]
Politecnico di Torino
M.Sc. in Data Science and Engineering
[1]s308563@studenti.polito.it, [2]s308964@studenti.polito.it, [3]s317672@studenti.polito.it

Abstract

This project aims to explore and enhance the Federated Learning (FL) framework within the context of Semantic Segmentation (SS) in autonomous driving. We begin by establishing a baseline within a centralized framework and train a model on the provided dataset. Subsequently, we transition into the federated framework and evaluate the model's performance from different perspectives. Our experimentation includes a pre-training phase and an assessment of domain adaptation in federated semantic segmentation. We also conduct self-supervised experiments and explore the possibility of using pseudo-labels in this field. To further enhance federated learning, we propose a novel idea Conditioned Cycle/Epoch Allocation (CCEA), which is built upon the existing concept of Federated Learning via Sequential Superclients Training (FedSeq) framework [37]. Our proposed idea involves scoring clients based on their performance during communication rounds and dynamically allocating cycles/epochs accordingly. The experimental results showcase promising improvements over the baseline, indicating the efficacy and efficiency of this proposal.

1. Introduction

Deep learning has profoundly changed many industries, including healthcare, finance, and the one that will be analyzed in this paper - the vehicles industry.

Although the development and research of artificial intelligence for vehicles are sometimes controversial, research in this field has happened for decades [8], with one of the objectives being to develop fully autonomous vehicles that can improve safety and quality of life.

Despite the fact that by July 2023 vehicles operating at Level 3 of automation ("eyes off" as per SAE International's classification) and above have not yet made an impact on the market, Japan nationally enforced the revised "Road Traffic Act" in the 1st of April 2023, allowing Level 4 vehicles, i.e. fully automated driving under certain conditions, on public roads. Nonetheless, even when we already see advancements in legislation regarding autonomous vehicles [17], there still remain many challenges yet to be solved and solutions yet to be improved.

In this paper, we benchmark a centralized SS model in Sec. 4.2, Federated Average (FedAvg) in Sec. 4.3, a recently introduced setting for semantic segmentation for autonomous driving, called Federated source-Free domain adaptation (FFreeDA) [30] in Sec. 4.5, in this context, the server will pre-train the model on labeled source domain data, possibly synthetic. Finally, we test the performance of CCEA in Sec. 4.6.

2. Related works

2.1. Semantic segmentation

The SS task aims at classifying each pixel in an image into a variety of possible classes. Essentially, it aims at dividing each image into different areas, each one representing a different class region in the original image. This task can be considered crucial in autonomous driving scenarios, where cars need to be equipped with well-designed and high-performing models to have a deep understanding of their surroundings. Although important and widely studied, SS in this setting is still a formidable challenge, one of the reasons being that it usually necessitates extensively annotated datasets. One of the papers that popularized using encoder-decoder networks for SS is [2], and [32] made an essential step by focusing computation times, which are especially important to enable real-time applications, and presented a unified architecture to approach classification, detection, and SS in the autonomous driving setting.

2.2. Federated learning

With the increase in computational power and an immense amount of data generated every day by devices, e.g. phones, wearables, and even autonomous vehicles, it has become very attractive to push computation to the edge and make use of such data. This increasing interest has led to the development of FL introduced in [24]. In such a setting, each participating device (client) holds a certain amount of

data, which is used to train local models, then the server would only receive the updates to the global model.

Optimization in FL comes with its own unique challenges that have to be carefully dealt with, including:

- Statistical heterogeneity: each client uses his device in unique ways, i.e. in the specific context of this paper, each client can be in distinct cities and use such device at different times of the day. Therefore the data held by such a client might not be a good representation of the distribution of all clients as a whole, each client will have their own distribution (Non-IID), and the model might not generalize well to new devices in the network. To solve this, [30] proposes a federated clustered aggregation scheme based on the clients' low-level features, while [19] proposes to give a higher relative weight to devices where the loss is higher.

- Communication bottleneck: as the number of clients participating is expected to be huge, communication might become a bottleneck for FL. For this reason, researchers have proposed using lossy compression and dropout [4], and decentralized training with local updating methods [18].

- Privacy: even though federated learning is designed to improve privacy by not sending the clients' dataset to the server, there are still potential privacy issues. While raw data isn't shared, sensitive information can still be obtained by properly inspecting the update sent to the server, i.e. by using model inversion attack methods [13]. One possible solution is using differential privacy methods, e.g. randomly perturbing the outputs [34].

2.3. Synthetic data

The training of models for the SS task usually requires a large amount of labeled data, but manually annotating pixels in images is a very time-consuming task, reaching up to 90 minutes per image [7], and the reliability of the labeling itself could vary across the dataset. In order to address this challenge, researchers started experimenting with synthetic data [12]. Such datasets have been shown to increase the accuracy of models when used together with real-world images [27] [1].

Obtaining images and meta-data from open-source games is easy, but most of these do not have high-quality graphics and diverse scenarios [5], unlike games from the industry with the highest development budgets. However, it has been demonstrated by [26] that extracting annotated data from these games, even without access to the source code, where, remarkably, they manage to annotate each image in an average time of just 7 seconds.

2.4. Domain adaptation

Models pre-trained with synthetic data cannot easily generalize to real-world scenarios due to the difference between the domains (domain shift). To solve this, Domain Adaptation (DA) methods were introduced [15], in which a model trained on the source domain (SD) is adapted to perform effectively in another, different but related, target domain (TD).

Although it can be considered sufficient to fine-tune models that were pre-trained on synthetic data on the TD, it still would need a great amount of labeled data on the target side, which can be quite troublesome to get in real-world cases, as already discussed. To overcome this barrier, researchers have deeply investigated Unsupervised Domain Adaptation (UDA) methods, where the TD data would have no annotations [36].

At the start, UDA methods focused primarily on non-deep learning strategies [3, 10], envisaging to bring closer the TD and SD features. In contrast, recently this setting got dominated by deep learning approaches with a variety of methods, including Discrepancy-based [10, 23, 28, 38], Adversarial Discriminative [9, 15, 33], Adversarial generative [11, 21, 31], Self-supervision-based with target pseudo-labels [20], and FDA methods [35], in which one makes use of low-level features from the TD by substituting the low-2D-frequency components of the SD with those of the target.

3. Methodology

To conduct our experiments, we first trained the DeepLabv3 [6] with MobileNetV2 [16] as the backbone on the ItalDesign DAtaset (IDDA) [1] within a centralized framework to do hyperparameter tuning and established a performance baseline in Sec. 4.2. Then, while considering our experience from the previous experiment, we tested the model in a federated framework in Sec. 4.3. However, we then reverted back to the centralized framework to train models with the GTA5 dataset [26], in Sec. 4.4 with and without Fourier Domain Adaptation (FDA) [35] to examine the effects of domain adaptation. Following these tests, we switched to the federated framework again to experiment pseudo-labeling technique using GTA5 pre-trained models to train in IDDA in Sec. 4.5. Finally, we evaluate CCEA in Sec. 4.6.

3.1. Self-Training with Pseudo-Labels

Regarding the self-supervised learning tests, we employed the teacher-student technique [14]. To implement this approach, we first trained a teacher model using the GTA5 dataset. We then utilized this teacher model to generate pseudo-labels for the training data points within the IDDA dataset. Next, we trained a student model using

this augmented dataset. Iteratively, we updated the teacher model using the predictions of the student model, creating a self-supervised learning framework.

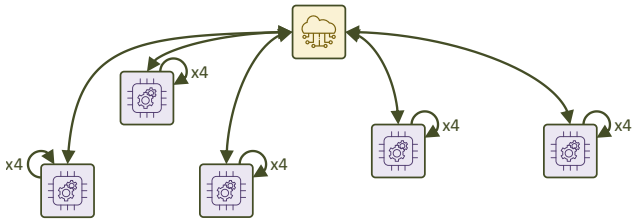
3.2. FedSeq

The cornerstone of our proposed idea is the FedSeq technique. It introduces two fundamental approaches to enhance federated learning: exclusive clustering of clients and sequential learning within each cluster [37]. By combining these two strategies, the learning process becomes a hybrid of sequential and federated frameworks. In this new framework, instead of training the model and uploading it to the server as illustrated in Fig. 1a, each client passes its updated model to another client within their assigned cluster as illustrated in Fig. 1b. After completing a learning cycle within each cluster, the final model is uploaded to the server. It then federates its received models into a single aggregated model that is used in the subsequent communication round. According to the findings presented in [37], the FedSeq technique has demonstrated effectiveness in improving the overall performance of federated learning.

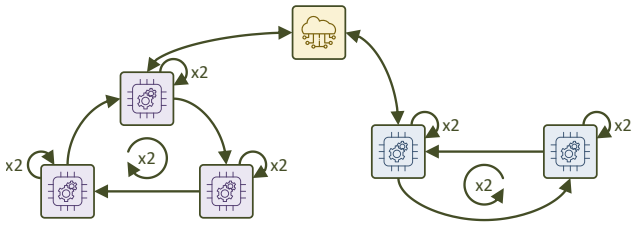
3.3. Proposed idea

Inspired by the hybrid framework discussed earlier, we are now able to propose a novel approach to further improve the federated SS task, the CCEA. In this approach, we score clients based on their performance during each communication round. We also assign a score to each cluster, that is the average score of its clients. Subsequently, as illustrated in Fig. 1c, in the next communication round, we leverage these scores to allocate a different number of cycles to each cluster and also a different number of epochs to each client. This allocation process can occur in two approaches: the *Lower-More* approach (CCEA_{lm}) and the *Higher-More* approach (CCEA_{hm}). In the CCEA_{lm} approach, clusters or clients with lower scores from the previous round are respectively assigned more cycles or epochs. Conversely, in the CCEA_{hm} approach, we implement the opposite, where the cycle/epoch allocation is inversely proportional to the scores.

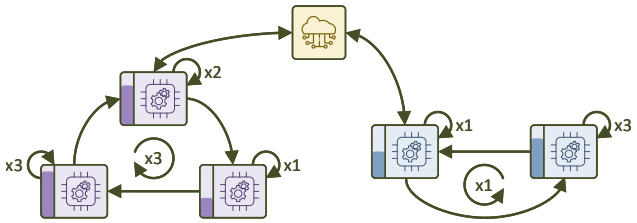
Please note that by scoring clients and treating them differently based on their performance, this approach incorporates an award/punishment mechanism, where well-behaving clients receive more attention. This can enhance the robustness of federated learning against potential misbehaving or malicious clients. Furthermore, CCEA operates as a self-adjusting mechanism since the likelihood of obtaining a high score in a round is determined solely by the previous round's performance, therefore, each client always has a chance to improve its performance and increase its effectiveness on the training.



(a) The FedAvg approach. Each self-pointing arrow depicts the number of epochs allocated to each client.



(b) The FedSeq approach. Each cluster is depicted with a special color. Also, each rotating arrow depicts the number of cycles allocated to each cluster.



(c) The CCEA approach. The schematic bar next to each client depicts its score from the previous communication round.

Figure 1. A conceptual illustration of the FedAvg, FedSeq, and CCEA approaches. In all three figures, the server is at located the top and clients are spread around it.

4. Experiments

4.1. Datasets and Training Details

As discussed before, we train the DeepLabV3 network [6] with MobileNet2 [16] as the backbone, which is done under different conditions and using two different datasets. A subset of the synthetic IDDA dataset [1] was used with a training set composed of 600 $(1920x1080)$ images, 120 on the test set of images belonging to the same domain, and 120 on the test set of images belonging to a different domain (i.e. heavy rain), respectively, we define the mean intersection over union (mIoU) evaluation metrics *Eval*, *Same*, *Diff*. The training set of IDDA was further divided into 24 clients, each with 25 images in a single coherent domain (town, weather, vehicle). A subset of GTA5 [26] synthetic dataset was also used for training with 500 $(1920x1080)$ images, and given the DA we do from the SD (GTA5) to TD (IDDA), we use the mIoU evaluated on these domains with prefix *SD* and *TD*. The experiments were held on the cloud with the support of Google Colaboratory [22] and Weights & Bias [29], on NVIDIA Tesla T4 GPU.

Table 1. Random search distributions for Centralized framework.

Object	Hyperparameter	Range	Distribution
SGD	wd	[1e-5, 1e-1]	Log uniform
	m	[0.5, 0.95]	Uniform
CA	lr	[1e-5, 1e-1]	Log uniform
	T_m	[5, 15]	Uniform

Table 2. Nine experiments of hyperparameter tuning for Centralized framework.

lr	wd	m	T_m	Eval	Same	Diff
1.4e-5	3.9e-2	6.5e-1	13	5.5%	7.2%	5.8%
1.5e-5	1.0e-5	6.9e-1	8	7.1%	8.9%	6.4%
2.6e-5	1.7e-4	8.2e-1	11	8.7%	10.9%	9.1%
1.0e-3	1.4e-2	5.9e-1	14	28.7%	29.6%	23.7%
1.6e-3	5.3e-4	5.6e-1	12	29.4%	30.4%	23.6%
1.0e-3	1.9e-5	7.8e-1	7	29.6%	30.6%	23.3%
8.2e-2	2.4e-3	6.6e-1	9	48.4%	49.4%	36.5%
5.6e-2	9.8e-4	6.9e-1	15	48.6%	50.3%	**37.6%**
7.3e-2	2.7e-3	7.2e-1	7	**49.6%**	**50.5%**	34.2%

4.2. Centralized framework

The creation of a baseline centralized framework was separated into three phases, firstly we did hyperparameter tuning on the Stochastic Gradient Descent (SGD) optimizer. Then we selected additional transformers for data augmentation finally we tuned for the Cosine Annealing (CA) scheduler.

4.2.1 First hyperparameter tuning

Given the large number of real values that many hyperparameters may take, we created a random search with the ranges and distributions described on Tab. 1. For this step, we trained each of the 42 executed runs over 5 epochs, used only the random horizontal flip as data augmentation transform, and normalization over each of the color channels.

On Tab. 2 we present three runs for each range of validation mIoU, lowest, intermediate, and highest, the best results were obtained with higher starting learning rates (lr), and smaller weight decays (wd), while both the momentum (m) and the CA's period (T_m) had a small effect on the metric. These results can be explained by the fact with a small number of epochs, the highest convergence is obtained by maintaining greater learning rates. From this phase, we chose to continue using *starting learning rate* 0.1, *weight decay* 0.0005, and *momentum* 0.65.

Table 3. Transformers experiments.

T_m	Transformer	Eval	Same	Diff
15	T1	45.9%	50.2%	33.1%
20	T1	47.4%	51.3%	32.7%
15	T2	53.3%	51.8%	34.8%
15	T3	53.4%	53.0%	39.8%
20	T2	54.3%	52.3%	40.0%
20	T3	**56.1%**	**54.4%**	**45.9%**

4.2.2 Data augmentation transformers

Three transformers were tested with 15 epochs, and varying T_m. The first (T1) is composed of a random image crop of size $(540, 960)$ pixels, without padding or resizing; color jittering with ranges of values $\pm 40\%$ of brightness, $\pm 40\%$ of contrast, $\pm 50\%$ of saturation, and $\pm 10\%$ of hue; random horizontal flip; channel standardization.

The second (T2) is composed of a random resized crop with expected output size $(1080, 1920)$ pixels, range of size of the original size cropped $(0.25, 1)$, aspect ratio of the original aspect ratio cropped equal to 1, done by bilinear interpolation; a color jittering, random horizontal flip, and standardization with the same parameters as the first transformer.

The third transformer (T3) is equal to the second transformer except for the aspect ratio of the original aspect ratio cropped of the randomly resized crop layer which will be within the range $(3/4, 4/3)$.
All transformers randomly select a crop of the image and add color jitter within ranges that would be able to generalize to different domains in terms of the time of day, and weather conditions. The model should also be robust to the color of the objects on the training set so that the network to learn other features that generalize better. To then be followed by standardization of the color channels. Additionally, T2 and T3 have the capability of resizing the crop, meaning that it learns with zoomed-in or zoomed-out objects, so they augment instances of objects perceived at different distances too. T3 outperforms the rest, as noted on Tab. 3, therefore such a transformer will be used in all of the following experiments.

4.2.3 CA hyperparameter tuning

Keeping the final value of CA to zero, we decide the ratio $T_m/epochs$ by doing a grid search with epochs fixed to 30 and obtained the results of Tab. 4, from which we decided to use the 1.5 ratio on the experiments that will follow. We further trained the best model reaching 40 epochs and obtaining *Eval* of 68.1%, *Same* of 62.7% and *Diff* of 52.5%.

Table 4. $T_m/epochs$ ratio experiments.

T_m	$T_m/epochs$	Eval	Same	Diff
30	1.00	61.8%	57.9%	47.0%
60	2.00	61.9%	58.0%	**48.4%**
35	1.16	62.5%	58.9%	47.9%
50	1.66	63.4%	59.1%	47.3%
40	1.33	**63.5%**	**59.3%**	47.4

Table 5. FedAvg experiments.

C	E	Rounds	5Eval	5Same	5Diff
2	1	1000	58.9%	60.5%	51.1%
24	1	467	59.3%	60.7%	49.2%
4	1	1000	62.9%	62.9%	**51.9%**
8	1	1000	64.1%	**63.4%**	51.3%
2	3	1000	64.1%	60.9%	50.1%
2	6	1000	**64.7%**	60.8%	47.1%

4.3. Supervised Federated Learning

Introducing the usage of FedAvg [25], the training set was divided into clients, each belonging to a specific domain based on the type of car, weather, time of day, and town. A server will manage communication rounds between itself and each of the clients available on the network, for each round a set of C clients will be randomly selected, and the server will send its model to each client which will train the model independently for E local epochs. Then the server will collect all the weights of the new models and compute the weighted average of them to create a new server model.

We trained on this setting using six configurations as seen in Fig. 2, in which for visualization purposes, we discretize the validation mIoU for every 5 rounds. All configurations were trained for 1000 rounds, except for the one with 24 clients per round, which was trained for 467 rounds. Given the instability of the metrics, on Tab. 5, we introduce the metric *5X*, which would be the average *X* mIoU over the latest 5 rounds.

From the results in Fig. 2 we observe that the greater the number of clients per round, the more stable the mIoU over the rounds, for the reason that the for each round, there is a higher probability of training with different domains, and in future rounds, the central model would have already made progress on learning the task on such domains. For an increasing number of local epochs, the convergence is faster (Tab. 5) but the mIoU is less stable.

4.4. Pre-Train Phase and Domain Adaptation

In order to simulate the case where we use a synthetic data pre-trained model to create the pseudo labels used for training locally in the client, we now train and perform hyperparameter tuning on DeepLabv3 [6] from scratch with the GTA5 dataset [26] after mapping the label numbers to match with the ones of the IDDA dataset [1]. And validated the results with the previously used training set of IDDA, and tested on the test set of IDDA on the same and different domains.

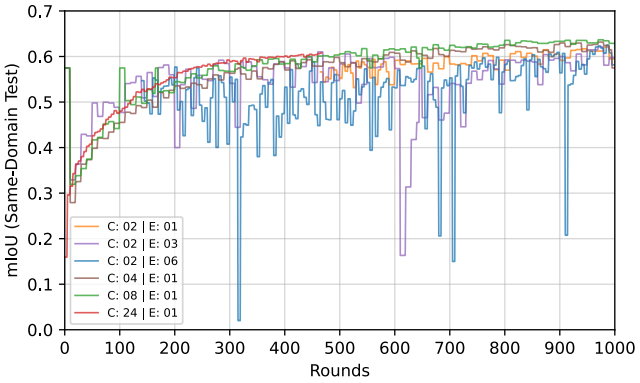


Figure 2. Discretized FedAvg mIoU on test set of same domain for different parameters

Table 6. Random search distributions for GTA5 pre-trained model.

Object	Hyperparameter	Range	Distribution
SGD	wd	[1e-5, 1e-2]	Log uniform
	m	[0.5, 0.95]	Uniform
CA	lr	[1e-3, 1]	Log uniform
	T_m	[5, 15]	Uniform

We used the T3 defined on Sec. 4.2.2 because it creates the most valuable generalizations, to which we will later add an FDA [35] layer to it. Notice we keep the same standardization statistics because we will to perform better on IDDA, not on GTA5. And when we add the FDA, the mean will be dependent on the IDDA style of each client.

In addition we did hyperparameter tuning similarly to Sec. 4.2.1, with a Random search following the distributions on Tab. 6, for 15 epochs for each experiment. We explored different ranges to adapt from what we observed on Sec. 4.2.1, so the starting learning rate cannot be lower than $1e^{-3}$, since the lowest values were not as promising before. And the weight decay, although not as important on Sec. 4.2.1, still the highest values were less performing, therefore we reduced the maximum value to $1e^{-2}$.

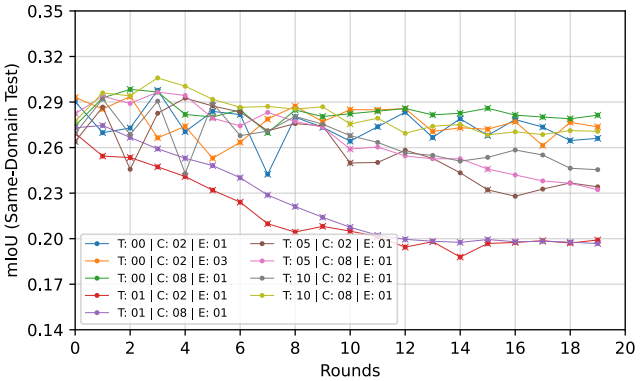
On Tab. 7 we present three runs for each range of target validation mIoU, lowest, intermediate, and highest. From the set of 33 experiments, we continue our paper using the best configuration on Tab. 7. For the best configuration we trained for 25 epochs and the best model saved was obtained on the 19th epochs with metrics *SD* of 46.7%, *TD* of 27.3%, *Same* of 27.3%, and *Diff* of 25.5%; this will be the GTA5

Table 7. Nine experiments of hyperparameter tuning for GTA5 pre-trained model.

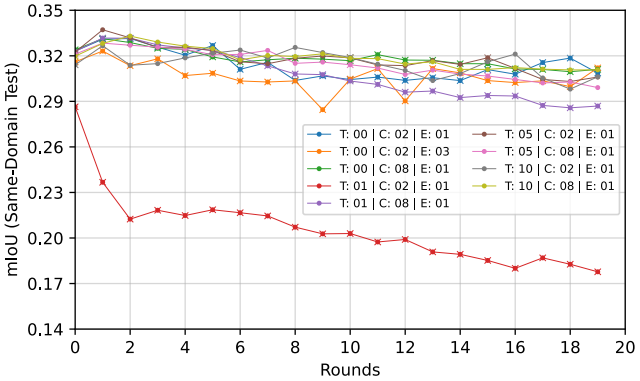
lr	wd	m	T_m	SD	TD	Same	Diff
5.5e-1	6.8e-3	.75	18	7.8%	6.7%	6.6%	6.7%
8.2e-1	3.3e-5	.70	23	15.3%	11.3%	11.2%	10.0%
3.6e-1	1.1e-5	.62	27	25.1%	13.8%	13.6%	5.7%
4.5e-2	3.6e-4	.87	29	38.9%	20.5%	20.0%	11.5%
3.6e-1	6.3e-5	.63	40	26.1%	20.8%	20.6%	13.6%
2.7e-3	1.6e-4	.94	22	41.0%	20.9%	20.9%	19.0%
2.3e-2	3.6e-5	.78	25	45.3%	25.6%	25.7%	**20.6%**
2.3e-2	7.9e-4	.84	24	45.3%	26.1%	**25.9%**	19.1%
1.4e-2	5.0e-4	.89	25	**45.4%**	**26.3%**	25.8%	19.6%

Table 8. FDA experiments.

Window Size	SD_{max}	TD_{max}	$Same_{max}$	$Diff_{max}$
18	37.9%	24.0%	23.7%	20.0%
12	38.9%	24.9%	24.6%	21.1%
4	38.6%	25.3%	25.0%	21.0%
16	38.1%	26.1%	25.6%	19.5%
2	38.0%	26.4%	26.0%	21.7%
20	38.8%	26.7%	26.5%	21.6%
8	37.1%	26.7%	26.6%	23.1%
6	38.4%	28.1%	28.0%	23.7%
10	**38.9%**	28.5%	28.3%	23.9%
14	38.1%	**29.1%**	**28.8%**	**24.3%**



(a) *Same* mIoU with GTA5 pre-trained teacher model



(b) *Same* mIoU with GTA5 FDA pre-trained teacher model

Figure 3. Comparison of *Same* mIoU among pseudo-labeling without FDA (*Top*) or with FDA *Bottom*

pre-trained model.

We now implement domain adaptation through the use of FDA [35] implemented on the GPU, using the same hyperparameters decided before, we need to tune the FDA window size based on the maximum value of the validation metrics over 10 epochs, the results are presented on Tab. 8, the window size equal to 14 outperformed the rest on generalizing over the IDDA dataset. So to create an FDA GTA5 pre-trained model, we trained with these hyperparameters for over 50 epochs, and the best model we saved was obtained on the 31st epoch with *SD* of 49.8%, *TD* of 29.8%, *Same* of 30.0%, and *Diff* of 18.8%.

4.5. Self-Training and Pseudo-Labeling

Having a GTA5 pre-trained model with mIoU on the IDDA test set of the same domain of the training set of 27.3% and the FDA GTA5 pre-trained model with 30.0%, we perform a set of experiments with a pseudo-labeling given a *teacher model* that creates the ground truths to train the *student model* in a FedAvg framework for 20 rounds. We start by setting the student model equal to the teacher one and test the update period T, that is the period of rounds for which we update the teacher model with the trained student model. Fig. 3a uses the first model and Fig. 3b the second each point marked by an x had the teacher model exchanged on such round. For both cases, we use the starting *learning rate* of the scheduler of 0.01, and *confidence threshold* of 90%.

In Fig. 3 we first observe that the implementation of FDA resulted in superior performance on the mIoU of the same domain tests for most configurations, so the technique demonstrated a partial success in adapting from the GTA5 to the IDDA domain. Another important observation is that in these experiments the pseud-labeling mechanism delivered a mostly decreasing mIoU over the communication rounds compared to Fig. 2, together with the observation that configurations that more frequently set the teacher model to be the same as the current round's student model had also a worse outcome, we can affirm that the model is not improving, rather it is intensifying its own mistakes, on Tab. 9 we observe that the training mIoU is very high, so it is indeed overfitting on low-quality pseudo-labels (initially producing around 30% mIoU). It is also worth noting that on the first two rounds, there is generally an increase in the metric, it is explained by the fact that the model is training for the first time with new data belonging to the same distributions of the test set.

Table 9. Results of self-training and pseudo-labeling experiments.

FDA	T	C	E	5Train	5Eval	5Same	5Diff
✗	1	2	1	69.7%	20.0%	19.8%	14.8%
✗	1	8	1	69.6%	20.0%	19.8%	15.1%
✗	5	2	1	75.3%	23.2%	23.3%	**23.2%**
✗	5	8	1	82.2%	24.0%	23.9%	19.0%
✗	10	2	1	81.0%	25.1%	25.2%	21.7%
✗	10	8	1	82.9%	27.1%	27.0%	21.6%
✗	0	2	1	83.3%	26.9%	27.0%	20.6%
✗	0	2	3	80.8%	27.0%	27.2%	21.3%
✗	0	8	1	82.8%	**28.0%**	**28.2%**	21.9%
✓	1	2	1	76.3%	17.9%	18.3%	15.2%
✓	1	8	1	88.4%	28.5%	28.9%	20.7%
✓	5	8	1	88.6%	29.8%	30.3%	21.8%
✓	0	2	3	90.1%	30.5%	30.4%	21.3%
✓	5	2	1	82.8%	30.6%	30.9%	21.4%
✓	10	2	1	81.9%	30.7%	30.9%	22.9%
✓	10	8	1	90.1%	31.0%	31.1%	22.4%
✓	0	8	1	88.8%	31.3%	31.2%	22.1%
✓	0	2	1	84.5%	**31.4%**	**31.2%**	**23.2%**

4.6. FedSeq and CCEA Techniques

In this section, we evaluate the abilities of the FedSeq and CCEA methods. For this purpose, we attempted to follow the same configuration we utilized in Sec. 4.3 during our supervised federated experiments. We trained the model only in 200 communication rounds with T_m 250, and performed similar tests by implementing FedAvg, FedSeq, FedSeq+CCEA$_{lm}$, FedSeq+CCEA$_{hm}$, and FedAvg+CCEA$_{hm}$ approaches. In addition, to preserve fairness in our comparisons, we used the hyperparameters presented in Tab. 10, where the average number of loops over the data of a single client is equal to 3. We employed random clustering in the FedSeq method. Also, each client was randomly allocated 1 epoch or 2 epochs due to the so-called fairness preservation. And used the training mIoU as the score of each client in the experiments involving the CCEA approach. Scores were normalized during cycle/epoch allocation.

Accordingly, as we can see in Fig. 4, all the approaches surpass FedAvg in terms of convergence speed and mIoU value. In addition, as shown in Fig. 4b, the experiment on the different-domain test dataset indicates that FedSeq+CCEA$_{hm}$ achieves even greater improvement compared to FedSeq. However, although this improvement is not explicitly evident in the same-domain experiment, Fig. 4a demonstrates that both FedSeq+CCEA$_{lm}$ and FedSeq+CCEA$_{hm}$ show less variance in the mIoU value during the training compared to the FedSeq. This stability is further enhanced by the FedAvg+CCEA$_{hm}$ approach. This implies that the CCEA approach aligns with the robustness we discussed in the previous section.

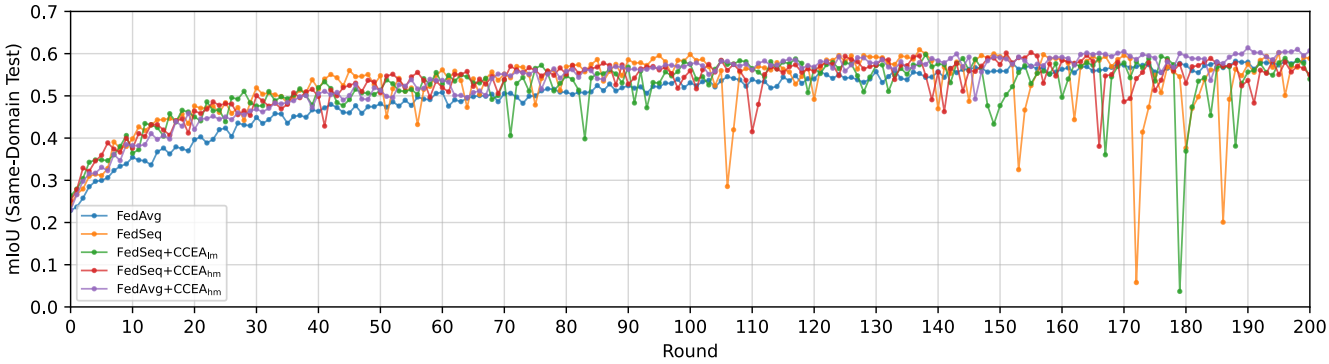
Table 10. List of parameters considered in testing, with N, L_{min}, L_{max}, E_{min}, and E_{max} respectively denoting the number of clusters, minimum number of cycles per cluster, maximum number of cycles per cluster, minimum number of epochs per client, and maximum number of cycles per client. All the tests are performed with 5 clients.

Method	N	L_{min}	L_{max}	E_{min}	E_{max}
FedAvg	1	1	1	3	3
FedSeq	3	2	2	1	2
FedSeq+CCEA$_{lm}$	3	1	2	1	3
FedSeq+CCEA$_{hm}$	3	1	2	1	3
FedAvg+CCEA$_{hm}$	1	1	1	1	5

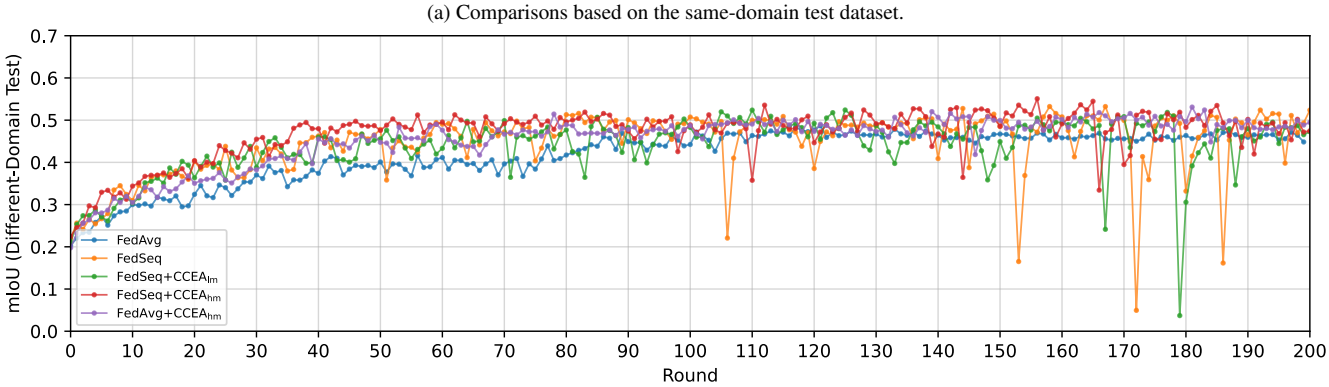
Table 11. Experiments of hyperparameter tinning for CCEA$_{hm}$ approach, with R and C respectively denoting the number of rounds and the number of clients. All the tests were performed with $L_{min} = E_{min} = 1$.

R	N	C	L_{max}	E_{max}	Eval	Same	Diff
100	2	5	2	5	61.2%	59.0%	50.5%
100	2	5	4	3	62.4%	59.6%	49.4%
100	2	5	4	5	61.3%	56.4%	48.3%
100	2	8	2	3	44.5%	44.0%	36.1%
100	2	8	2	5	63.3%	57.7%	50.1%
100	2	8	4	5	58.3%	57.3%	48.5%
100	3	5	2	3	62.7%	59.6%	51.8%
100	3	5	4	3	62.9%	**62.0%**	**52.6%**
100	3	5	4	5	**65.2%**	59.4%	51.1%
100	3	8	2	3	46.1%	48.2%	41.0%
100	3	8	2	5	56.3%	54.8%	48.3%
100	3	8	4	5	57.2%	53.7%	48.8%
250	2	5	2	3	67.6%	63.5%	53.8%
250	3	5	2	5	**69.2%**	**65.0%**	**54.0%**

Fig. 4 also includes a comparison between CCEA$_{lm}$ and CCEA$_{hm}$. The results indicate that CCEA$_{hm}$ is a better strategy than CCEA$_{lm}$. For this reason, we conducted a grid search only on the the CCEA$_{hm}$ approach to tune its hyperparameters and to further optimize its performance. The result is presented in Tab. 11. Consequently, we can decide that the combination of $N = 3$, $C = 5$, $L_{max} = 4$, $E_{max} = 3$ is the best configuration for the CCEA$_{hm}$ approach, when the model is trained within 100 rounds. However, as presented in Tab. 11, we also continued two experiments until $R = 250$ to further analyze this approach. We selected these two combinations because of their performance within 100 rounds. Accordingly, we can conclude that CCEA$_{hm}$ approach outperforms all other methods that we discussed in this section in terms of Eval, Same, and Diff metrics.



(a) Comparisons based on the same-domain test dataset.



(b) Comparisons based on the different-domain test dataset.

Figure 4. Comparing the FedAvg, FedSeq, and CCEA methods.

5. Conclusion

In this paper, we conducted various experiments with the use of the FedAvg, FedSeq, and CCEA techniques. We observed that, while moving from the centralized framework into a federated one can have negative effects on performance metrics, it provides the opportunity for parallelized learning with the computation on edge devices. We also tested this framework by implementing a self-supervision network and employing domain adaptation. Finally, we proposed a novel idea to improve FL based on the FedSeq approach. Our results demonstrated performance increase in terms of mIoU, convergence speed, and robustness.

References

[1] Emanuele Alberti, Antonio Tavera, Carlo Masone, and Barbara Caputo. IDDA: a large-scale multi-domain dataset for autonomous driving. *CoRR*, abs/2004.08298, 2020. 2, 3, 5

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017. 1

[3] et al. Ben-David, Shai. A theory of learning from different domains. In *Machine Learning*, volume 79, pages 151–175. Springer, 2010. 2

[4] Sebastian Caldas, Jakub Konečný, H. Brendan McMahan, and Ameet Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. *CoRR*, abs/1812.07210, 2018. 2

[5] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2722–2730, 2015. 2

[6] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017. 2, 3, 5

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. *CoRR*, abs/1604.01685, 2016. 2

[8] E.D. Dickmanns. The development of machine vision for road vehicles in the last decade. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 1, pages 268–281 vol.1, 2002. 1

[9] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks, 2016. 2

[10] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation.

In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2066–2073, 2012. 2

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. 2

[12] Ankur Handa, Viorica Patraucean, Vijay Badrinarayanan, Simon Stent, and Roberto Cipolla. Scenenet: Understanding real world indoor scenes with synthetic data. *CoRR*, abs/1511.07041, 2015. 2

[13] Ali Hatamizadeh, Hongxu Yin, Pavlo Molchanov, Andriy Myronenko, Wenqi Li, Prerna Dogra, Andrew Feng, Mona G. Flores, Jan Kautz, Daguang Xu, and Holger R. Roth. Do gradient inversion attacks make federated learning unsafe? *IEEE Transactions on Medical Imaging*, pages 1–1, 2023. 2

[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. 2

[15] Judy Hoffman, Dequan Wang, Fisher Yu, and Trevor Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *CoRR*, abs/1612.02649, 2016. 2

[16] Andrew Howard, Andrey Zhmoginov, Liang-Chieh Chen, Mark Sandler, and Menglong Zhu. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. In *CVPR*, 2018. 2, 3

[17] et al. Johannes Deichmann, Eike Ebel. Autonomous driving's future: Convenient and connected. *McKinsey Center for Future Mobility*, 2023. 1

[18] Anusha Lalitha, Xinghan Wang, Osman Kilinc, Yongxi Lu, Tara Javidi, and Farinaz Koushanfar. Decentralized bayesian learning over graphs, 2019. 2

[19] Tian Li, Maziar Sanjabi, and Virginia Smith. Fair resource allocation in federated learning. *CoRR*, abs/1905.10497, 2019. 2

[20] Yunsheng Li, Lu Yuan, and Nuno Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. *CoRR*, abs/1904.10620, 2019. 2

[21] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. *CoRR*, abs/1606.07536, 2016. 2

[22] Google LLC, 2018. 3

[23] Mingsheng Long, Yue Cao, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Transferable representation learning with deep adaptation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(12):3071–3085, 2019. 2

[24] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017. 1

[25] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016. 5

[26] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. *CoRR*, abs/1608.02192, 2016. 2, 3, 5

[27] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3234–3243, 2016. 2

[28] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Beyond sharing weights for deep domain adaptation. *CoRR*, abs/1603.06432, 2016. 2

[29] Lewis Shawn, Lukas Biewald, and Chris Van Pelt. Weights & biases – developer tools for ml, 2017. 3

[30] Donald Shenaj, Eros Fanì, Marco Toldo, Debora Caldarola, Antonio Tavera, Umberto Michieli, Marco Ciccone, Pietro Zanuttigh, and Barbara Caputo. Learning across domains and devices: Style-driven source-free domain adaptation in clustered federated learning, 2022. 1, 2

[31] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. *CoRR*, abs/1612.07828, 2016. 2

[32] Marvin Teichmann, Michael Weber, Marius Zöllner, Roberto Cipolla, and Raquel Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1013–1020, 2018. 1

[33] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation, 2020. 2

[34] Xi Wu, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey F. Naughton. Differentially private stochastic gradient descent for in-rdbms analytics. *CoRR*, abs/1606.04722, 2016. 2

[35] Yanchao Yang and Stefano Soatto. FDA: fourier domain adaptation for semantic segmentation. *CoRR*, abs/2004.05498, 2020. 2, 5, 6

[36] Kaichao You, Ximei Wang, Mingsheng Long, and Michael Jordan. Towards accurate model selection in deep unsupervised domain adaptation. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7124–7133. PMLR, 09–15 Jun 2019. 2

[37] Riccardo Zaccone, Andrea Rizzardi, Debora Caldarola, Marco Ciccone, and Barbara Caputo. Speeding up heterogeneous federated learning with sequentially trained superclients. *CoRR*, abs/2201.10899, 2022. 1, 3

[38] Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning, 2019. 2