Semantic Segmentation for Autonomous Driving

Quaranta Eleonora
Politecnico di Torino
Turin, Italy
s316198@studenti.polito.it

Rosato Sara
Politecnico di Torino
Turin, Italy
s317547@studenti.polito.it

Vanini Samuele
Politecnico di Torino
Turin, Italy
s318684@studenti.polito.it

Abstract

Semantic segmentation is a key task for autonomous driving, but it is also one of the most challenging, especially in a federated environment. In this scenario, the main problems are privacy concerns and heterogeneity, which affect both the data and the devices collecting them. In this paper, we propose a benchmark to test several existing techniques, starting from the FedAvg algorithm combined with Fourier domain adaptation and self-learning based on pseudo-labels. Finally, to improve the current baseline set by the previous benchmarks, we introduce a new batch normalization scheme based on SiloBN, which also leverages client clustering. We test our implementation on two datasets, IDDA and GTAV, and discuss the results. GitHub repository: https://github.com/SamueleVanini/MLDL23-FL-project.git

1. Introduction

Vehicles for autonomous driving need to "see" and understand what they are encountering to provide a high level of safety for passengers in and out of the car. Semantic segmentation [1], [2], [3] plays a crucial role in helping vehicles understand their environment. It associates individual pixels to known categories. While this method is effective, it's not the most efficient. In order to train robust semantic segmentation models that accurately represent real-world conditions, it's necessary to access large datasets, which may also raise privacy concerns if collected from users in a certain way.

Federated Learning (FL) can be a possible solution to overcome this problem [4], [5], [6]. In FL clients communicate periodically with the server to learn a global model, but they don't share data because the training is done directly on remote devices.

Despite these benefits, there are several open challenges for federated learning, including expensive communication, system heterogeneity related to different device characteristics, statistical heterogeneity of often non-IID data, and still some privacy concerns.

Another issue is domain shift, as a model must perform well in unseen domains. One solution could be the incorporation of style transfer methods, which aim at an effective generalization over unseen domains.

We conducted experiments evaluating the performance of some of the most used federated algorithms, such as FedAvg [4], various server optimizers, and some batch normalization techniques, like SiloBN [7], to assess the ability of current methods to address real-world challenges.

To improve the baseline obtained with those algorithms, we introduce Clustered Averaged Batch Normalization (ClAvBN), a new domain adaptation technique based on clusters of clients sorted through their styles extracted with Fourier Domain Adaptation (FDA) and used to aggregate batch normalizations statistics.

2. Related work

2.1. Semantic Segmentation

Semantic Segmentation is one of the crucial tasks of computer vision, consisting of a pixel-level labeling operation performed on a given image. During the last few years, promising results have been achieved in centralized settings thanks to the development of various deep learning techniques.

One of the crucial aspects to take into account when approaching this task is the availability of labeled training data: labeling an image is an expensive operation [8], and therefore the availability of a training set which is both sufficiently large and fully labeled is not always guaranteed. For this reason, it is possible to discern three main categories of segmentation methods based on the training data provided, as suggested in [10]. Supervised methods assume the availability of a sufficiently large, fully labeled training set. This is the case of DeepLabV3 [9], based on a Convolutional Neural Network leveraging context information about the image thanks to an Atrous Spatial Pyramid Pooling (ASPP) module, shown in Fig. 1.

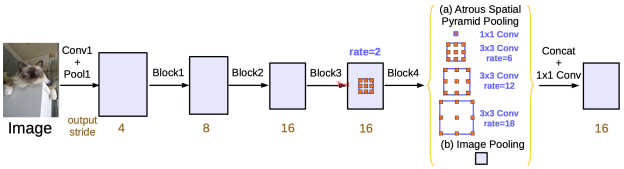


Figure 1. Parallel modules with atrous convolution (ASPP) [9] Atrous convolution, also called dilated convolution, is based on the idea of introducing "holes" in the feature extraction filter rather than increasing its size, allowing to increase the coverage of the image. ASPP extends this idea by combining multiple convolution layers with increasing dilation rates.

Weakly supervised methods leverage the availability of a large training set characterized by less accurate labels, consisting of tags, scribbles, and boxes [10] to reduce the labeling process complexity.

Differently from the latter, semi-supervised methods are employed when the training set contains a limited number of fully labeled images. These techniques leverage domain adaptation or few-shot learning to maximize the amount of information extracted from the data.

2.2. Federated Learning

Federated Learning is a machine learning technique leveraging a decentralized approach to train a global model on data that is not directly accessible to a centralized server. Its deployment allowed to tackle privacy concerns in applications requiring data to remain stored on client devices, such as those involving health or financial data, mobile and IoT devices, or autonomous driving.

Among the core challenges in Federated Learning, expensive communications are the bottleneck in the network [5]: the decentralized paradigm requires the server to continuously communicate with the involved clients; therefore, an improvement of the overall efficiency of the process can benefit the whole model. The first way to achieve this is by implementing a variable number of local updates on each machine during each communication round.

Aside from this, heterogeneity represents another criticality in Federated Learning. We refer to the differences between the devices in the network in terms of hardware, connectivity, and power as system heterogeneity, whereas statistical heterogeneity is the common non-i.i.d. property of data obtained from the different clients.

The most popular method used in federated learning is FederatedAveraging (FedAvg) [4], which is based on an iterative model averaging and works by having a server combine local stochastic gradient descent (SGD) of each client and perform model averaging. FedAvg allows to reach satisfying results both in the overall performance of the model and in the specific communication-efficiency problem, but it lacks convergence guarantees and tends to suffer from *client-drift*, leading to a slow and unstable convergence in settings characterized by data heterogeneity.

[11] proposes a Stochastic Controlled Averaging for Federated Learning (SCAFFOLD) which corrects the client-drift in local updates of FedAvg caused by data heterogeneity by leveraging variance reduction, also allowing a further reduction of the number of communication rounds.

2.3. Federated Domain Adaptation

As already mentioned, fully annotating sets of images to train semantic segmentation models can be an expensive operation; this has led to much research on the possibility to leverage synthetic data to train the model, in order to limit the labeling cost. The main limitation of this approach consists of the challenge of effectively transferring the knowledge learned from synthetic data to the real-world domain (domain adaptation). This issue has been widely analyzed and many techniques have been developed to address it.

Unsupervised Domain Adaptation (UDA) techniques [12] assume the lack of labels on the target domain and full availability of the training data in order to be able to reduce the domain gap between the source and the target domain (domain shift). In federated settings, however, source data is private and the model cannot rely on its availability to perform domain transfer tasks. To tackle this problem, new source-free domain adaptation paradigms have been proposed. They mostly rely on the concept of self-supervised learning (SSL), which is used to capture the features of the input data without the need for labels. Its basic idea consists in defining some auxiliary tasks which can be performed over the source data alone and forcing the network to extract meaningful information based on those tasks.

In particular, [13] introduces a Source-Free Domain Adaptation (SFDA) technique consisting of two different stages: knowledge transfer and model adaptation; this method involves a generator that synthetizes fake samples, a new dual attention distillation mechanism that allows to transfer and maintain the contextual information, and an intra-domain patch-level self-supervision module (IPSM), which uses the target domain pseudo-labels to implement the self-supervised learning mechanism.

[14] introduces Learning Across Domains and Devices (LADD) to solve Federated source-Free Domain Adaptation (FFreeDA), which assumes the possibility to pre-train the model on a labeled source dataset but forbids additional access to it, similarly to SFDA, thus introducing a more realistic setting for the semantic segmentation task. LADD leverages the presence of different data distributions over the clients to cluster them based on the style of their images. In this way, it is possible to consider two sets of parameters for the model: the shared parameters, aggregated over the entirety of the clients, and the cluster-specific parameters, aggregated exclusively over clients belonging to the same

cluster. Therefore, this technique also addresses other already mentioned criticalities of the federated setting, such as heterogeneity and communication efficiency.

Another way to address the challenge of domain adaptation is to leverage local-statistic batch normalization layers, as proposed by the developers of SiloBN [7].

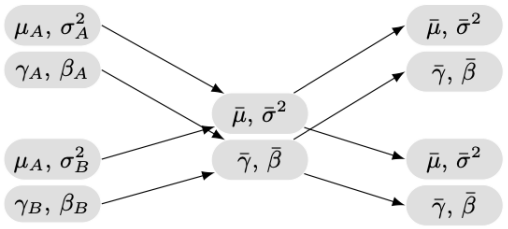
In general, a batch normalization layer performs the following operation on a tensor x:

$$BN(x) = \gamma \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta$$

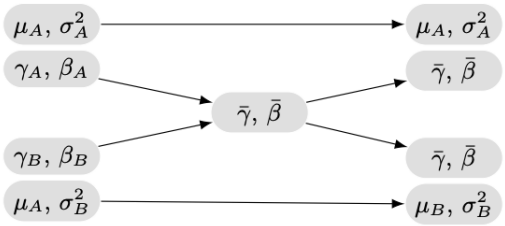
where μ and σ^2 are the running mean and variance computed for each channel of the tensor over both batch and spatial dimensions, thus encoding information about the local domain; β and γ, on the other hand, are learned normalization parameters which can be shared among different domains.

Usually, in a standard FedAvg iteration, both the local statistics (μ, σ^2) and the learned parameters (β, γ) are aggregated during each round, as shown in Fig. 2a; SiloBN, instead, distinguishes the roles of local statistics and learned normalization parameters by keeping the former local and performing aggregation exclusively on the latter, as shown in Fig. 2b .

The locality of μ and σ^2 improves the performance of the model on heterogeneous data and can therefore bring substantial improvements in federated settings.



(a) FederatedAveraging



(b) SiloBN

Figure 2. Comparison between plain FedAvg aggregation and SiloBN [7]

3. Proposed method

In this section, we present our proposed approach for federated learning for autonomous driving, starting with the introduction of the baseline for centralized settings, up to the generalization to federated settings and corresponding improvements.

3.1. Centralized setting

As a starting point for evaluating and comparing the performance of different methods and models, we created a central baseline for the IDDA dataset [8]. We used the DeepLabV3 network [9], a specific type of neural network architecture commonly used for image segmentation tasks. As a backbone, we used MobilenetV2 [15], a lightweight neural network architecture that is known for its efficiency in terms of computational resources and memory requirements.

We tested our model both on images belonging to the same domain as those we used for training and on images belonging to different domains.

Correctly addressing the problem of misalignment between pixels in the image and their corresponding labels during semantic segmentation was particularly important in this step. We applied some transformations to the image, mainly random resized crop and jitter. The first one scales an image by a random factor in the range (min_scale, max_scale) and, after resizing, extracts a smaller region or patch for an image typically at a random location, while jittering modifies image brightness, contrast, and saturation.

Combining all these techniques, the resulting image undergoes size and spatial variation, resulting in a more diverse dataset. This augmentation technique helps the model to be more robust and able to deal with variations in the scale and position of the objects during the inference process.

The best results obtained with the parameters reported in Sec. 4.2 are shown in Tab. 1.

3.2. Supervised federated setting

After having successfully defined the centralized baseline, we extended our model to the federated setting. Our implementation is based on the FedAvg algorithm [4], which can be summarized in the following steps: first, a subset of clients is randomly sampled to perform training on their local data by using the global model sent by the server; once each client has trained its local model, the updates computed with respect to the global model are sent to the server, which is in charge of aggregating them through an averaging operation in which each client's model is weighted by the size of its local dataset. This process is repeated for a customizable number of rounds, which is a parameter for the model. From a mathematical perspective, given the number of clients to be trained on for each round K, the global model configuration w, and the local empirical

risk L_k, the optimization problem can be written as:

$$\arg\min_{w} \sum_{k=1}^{K} p_k L_k(w)$$

where $p_k = \frac{n_k}{N}$ is the relative number of samples in the k^{th} client with respect to the total number of samples among all the clients. In this case, the local empirical risk results from the cross-entropy loss. The computation of each global model update at a given step *i+1* is:

$$w^{i+1} = \frac{1}{N} \sum_{k=1}^{K} n_k w_k^i$$

The parameters for the experiments have been chosen on the base of those performed in the centralized setting Sec. 4.2, therefore we only experimented with different numbers of local epochs, rounds, and clients involved in each round. The best experimental results are shown in Tab. 2 and will be discussed in Sec. 4.3.

3.3. Domain adaptation

As previously mentioned, in practical scenarios, self-driving vehicles do not have access to ground-truth labels, making semantic segmentation essential. Similarly, clients are unable to obtain the labels associated with the collected images, and even if they could, it would be an expensive process.

To address this challenge, we leveraged domain adaptation techniques [12], [13], [14] to transfer the knowledge learned from synthetic data to the real-world domain, as explained in Sec. 2.3.

Consequently, we assumed that our model was initially trained on an openly available supervised dataset, GTAV [16]. Instead of considering all of its semantic classes, we focused on the 16 classes shared with IDDA [8]. This allowed us to align the datasets and evaluate our model using the IDDA training set while actually training on the GTAV dataset.

We implemented checkpoints to save the most promising models throughout the evaluation process, ensuring the best model performance.

To improve semantic segmentation performance without requiring additional training, we implemented the Fourier Domain Adaptation (FDA) technique [17].
In this approach, the Fast Fourier Transform (FFT) was performed on each input image, and the low-level frequencies of the target images were replaced by those of the source images using the inverse Fast Fourier transform (iFFT). The goal was to minimize the domain discrepancy between the source and target datasets. For this purpose, we calculated the average style of each client in the IDDA training set and created a bank of target styles.

Next, we associated each source image style from the GTAV dataset with a random target style from the bank, keeping the semantic content intact.

Finally, we evaluated the performance of our FDA approach on the IDDA training set by conducting experiments with various window sizes.
The results of these evaluations are presented in Tab. 4.

3.4. Federated self-training

In this step, we describe the implementation of federated self-training leveraging pseudo-labels. As mentioned in the previous section, autonomous driving vehicles do not usually have access to a labeled dataset to perform the training. Aside from the already mentioned domain adaptation techniques, this issue can also be addressed through self-supervised learning techniques, like the one based on pseudo-labels presented by the authors of [18].

The implementation is based on the co-existence of two different models, the teacher model and the student model. The teacher model is pre-trained on GTAV, and the student model is initially set to be the same as the teacher. At first, the teacher model is used to predict the labels of the images contained in the IDDA dataset; the emitted predictions whose associated confidence is higher than a chosen threshold are fed into the student models of the involved clients as ground truth, and then used for their training. At each training round, the server performs FedAvg aggregation on all the clients selected as usual; the server model can be leveraged to further update the teacher model periodically, so as to hopefully improve its performance and prediction confidence.

Details about the performed experiments and corresponding results are discussed in Sec. 4.5.

3.5. Improvements: ClAvBN

Clustered Averaged Batch Normalization (ClAvBN) is a domain adaptation technique based on batch normalization layers to address the statistical heterogeneity challenge; it was inspired by a real-world scenario in which each client's dataset can incorporate different domains. Considering an actual autonomous driving vehicle, in fact, it can capture images in different weather conditions, lighting, and urban scenarios within the same day.

For these reasons, we decided to perform clustering on the clients based on their styles rather than different domains. To achieve this, we leveraged the implementation of FDA described in Sec. 2.3 and combined it with a variant of SiloBN [7] in a self-learning environment.

In particular, during training, we don't average the mean and standard deviation of each client, but we average γ and β parameters over clients belonging to the same cluster, as depicted in Fig. 3.

During the test phase, we try to assign each client to the

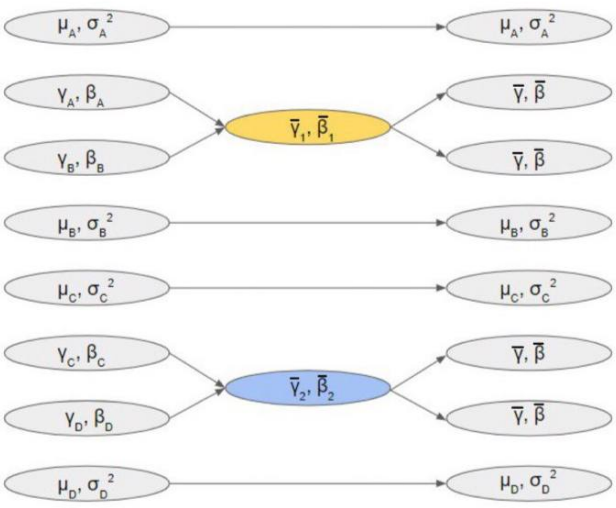


Figure 3. ClAvBN

closest cluster based on its style. At this point, the cluster parameters (γ, β) are fed into the client model and used for the semantic segmentation task. Our implementation involves only one teacher model, which is periodically updated by the server model, responsible for the generation of the pseudo-labels.
This process uses the client and corresponding cluster's statistics (μ, σ^2, γ, β), which allows to have a cluster-specific model. In addition, we incorporate the concept of distillation loss, where each client's loss is calculated by:

$$\mathcal{L}(x;W) = \alpha * \mathcal{H}\left(y, \sigma\left(z_s; T=1\right)\right) +$$

$$+\beta * \mathcal{H}\left(\sigma\left(z_t; T=\tau\right), \sigma\left(z_s, T=\tau\right)\right)$$

Here, α and β are hyperparameters that determine the weight of the fixed teacher model and the weight of the student model. τ is the parameter used in the context of softmax temperature explained in [19]. The results obtained with our ClAvBN model are reported in Sec. 4.6.

4. Experimental results and hyperparameters

This section summarizes all the experimental results obtained with the methods described above. In particular, the semantic segmentation performance is evaluated for each category using Intersection-over-Union (IoU) [10]. IoU is the area of overlap between the predicted segmentation and the ground truth divided by the area of union between the predicted segmentation and the ground truth. We compute the Mean Intersection-over-Union (mIoU) for the entire test set.

4.1. Datasets

Large-scale synthetic datasets are being generated from simulation engines with labels automatically obtained, which induces a large domain shift from the real-world data. We evaluated our model considering mainly two different datasets: IDDA and GTAV.
IDDA [8] is a synthetic dataset for semantic segmentation having 16 semantic classes and different weather conditions characterized by three axes: 7 towns (ranging from Urban to Rural environments), 5 viewpoints (simulating different vehicles), and 3 weather conditions (Noon, Sunset and Rainy scenarios) for a total of 105 domains. A domain is a triad: town $\in$ {T01, T02, T03}, wheater/illumination $\in$ {Clear noon, clear sunset, hard rain noon} and car $\in$ {AudiTT, Ford Mustang, Jeep Wrangler Rubicon}. We did some testing using partly images belonging to the same domains as the training images and partly images belonging to different domains.
GTAV [16] is a synthetic dataset collected in the high-fidelity rendered computer game GTA-V with pixel-wise semantic labels. It contains 24,966 images (video frames).

4.2. Centralized setting results

We tried different configurations for our experiments and saw that we got the best results when we used stochastic gradient descent as the optimizer instead of Adam. We tried different learning rates and set the value of momentum to 0.9.

The same considerations were made for the scheduler since we saw that the polynomial gave improvements with respect to the step. We set the power to 0.9 and decayed the learning rate of each parameter group using a polynomial function.

Almost all of our experiments are done with mean reduction since we didn't get any improvements with hard negative mining.

The other important hyperparameters that we have given to our model and that gave us significantly different results are as follows:

- Height and width: we found that a larger image gave better scores in terms of loss and mIoU on training and testing, in all the different domain conditions.
- Batch size: all our best results were achieved using batch size 4, but the last two rows of Tab. 1 also show some results obtained using batch sizes 8 and 16, respectively. Sometimes it was not possible to increase the batch size for certain configurations due to limited power resources.
- Number of training epochs: mostly between 50 and 100, since we saw that the results were good enough even without the necessity of increasing it.

We can state that all our results, as shown in Tab. 1, are satisfactory, even compared to the state of the art.

Table 1. Centralized baseline: results

parameters	epochs	loss	Train mIoU	SameDom mIoU	DiffDom mIoU
h=972 w=1728 lr=0.05	**50**	**0.06**	**69.42**	**61.39**	**47.62**
h=910 w=1620 lr=0.02	50	0.07	67.38	59.72	43.09
h=756 w=1344 lr=0.05	50	0.07	66.59	61.61	45.7
h=512 w=928 lr=0.05	50	0.09	61.63	53.57	33.38
h=512 w=928 lr=0.05	100	0.09	64.04	56.66	35.77
h=756 w=1344 lr=0.1	80	0.09	62.68	59.4	33.14

As mentioned above, our best results are obtained with large values for height and width, with only a small part of the image removed. In this way, our model has more pixels to evaluate, resulting in as better performance.

4.3. Federated setting results

The experiments have been performed using the optimal set of transformations found in the first step: random flips, random resized crop, and jittering of the images. Despite the optimal configuration of the hyperparameters would require us to set the height and width for the resize to be 972×1728, we encountered some computational limitations and therefore had to compromise to a final dimension of 756×1344, and a learning rate of 0.05; this allowed to find a reasonable trade-off between training times and performance of the model. The same computational limitations influenced the choice of the number of rounds as well, hence the lower number for the most costly experiments.

Table 2. Supervised federated setting: results

Number of clients	Number of epochs	Number of rounds	Train mIoU	SameDom mIoU	DiffDom mIoU
2	1	500	53.08	55.28	38.72
2	3	300	55.48	57.20	38.48
2	**6**	**500**	**60.69**	**59.61**	**43.32**
2	6	200	55.81	56.68	40.55
4	1	500	51.56	54.13	42.03
4	2	300	53.54	56.33	39.16
4	8	100	54.33	56.08	39.19
8	1	500	53.03	55.63	38.96

From the results shown in Tab. 2, it is possible to notice how the model benefits from a higher number of local epochs and total rounds; this can be justified by the fact that increasing the number of local epochs reflects into a better leverage of the local data of each client, and at the same time increasing the number of rounds allows for more iterations of the decentralized training, leading to a better model refinement. Even though our model achieves satisfying results, we firmly believe it could further benefit from an additional increase in both the number of rounds and local epochs.

4.4. Domain adaptation results

The configurations used for the optimizer, data augmentation, and batch size are those described in Sec. 4.2 since we found, after several tests, that we get better results with them than with others. We tried to remove the jitter but noticed that applying it gave an improvement of almost 10% on the whole test set.
We evaluated our model several times, and thanks to the use of checkpoints, we arrived at a total number of epochs of 100 for each run.

First, we trained our model from scratch, as explained in Sec. 2.3, without FDA, obtaining the results reported in Tab. 3.

Table 3. Best model without FDA

parameters	jitter	Source mIoU	SameDom mIoU	DiffDom mIoU	Target mIoU
h=756 w=1344 lr=0.01	**true**	**70.43**	**29.57**	**29.15**	**23.32**
h=756 w=1344 lr=0.01	false	64.12	19.40	19.39	10.93
h=756 w=1344 lr=0.01	true	60.02	24.09	24.35	21.18

From our results in Tab. 4, we noticed that the application of Fourier Domain Adaptation doesn't give any improvements with respect to the application of jitter in data augmentation. When no jitter is applied, instead, FDA gives better results, especially if the window size used is small.

4.5. Federated self-training results

Our experiments leverage the pre-trained models from Sec. 4.4, in which we performed training on the GTAV dataset and testing on the IDDA dataset, both with and without FDA.

A crucial hyperparameter to tune is the confidence threshold that the teacher model's predictions need to meet in order to be used as ground truth by the student models.

Table 4. Domain adaptation with FDA: results

window_size	Source mIoU	Target mIoU	SameDom mIoU	DiffDom mIoU
0.0018	**66.43**	**26.47**	**26.50**	**20.85**
0.018	64.07	25.89	25.98	13.82
0.09	62.84	21.70	21.46	10.29
0.18	62.45	19.73	19.82	10.11
0.3	61.44	25.22	24.92	13.78
0.5	60.88	24.43	24.59	14.32

On the one hand, we want the threshold to be sufficiently high to avoid feeding wrong labels to the student models, but on the other hand we need it to be low enough in order to have sufficient labels for the students' training. Based on the test we performed and on the results presented in [18], we opted for a confidence threshold of 0.9.

We experimented with different teacher model update intervals in order to find the best trade-off between performance and efficiency, keeping the number of local epochs fixed to 1 and the number of rounds fixed to 100.

From our experiments, whose results are shown in Tab. 5 and Tab. 6, we can see how the different teacher model update intervals have the greatest impact on the model performance. In particular, the best results are obtained with no updates, and they get progressively worse as the update frequency increases, leading to the worst results when the teacher model is updated at each round. On the other hand, the model pre-trained using FDA does not guarantee consistent improvements when compared to the non-FDA one, which reinforces the results obtained in the Sec. 4.4.

Table 5. Federated self training setting (no FDA): results

Number of clients	Teacher update	Train mIoU	SameDom mIoU	DiffDom mIoU
2	∞	**19.52**	**19.13**	**15.25**
8	∞	18.17	17.78	14.66
2	1	10.05	9.76	8.86
8	1	8.47	8.25	8.12
2	10	16.73	16.58	13.78
8	10	13.17	12.82	12.33
2	24	18.21	17.91	14.27
8	24	14.77	14.43	13.68

4.6. ClAvBN results

Similarly to Sec. 4.5, we performed our evaluation of ClAvBN by varying numbers of clients per round and teacher model update intervals, while keeping the number of local epochs fixed to 1, the number of rounds fixed to 100, and a confidence threshold of 0.9 for the self-learning mechanism. The values of α and β are set to 0.3 and 0.7 respectively, whereas τ is set to 2. We additionally compared our algorithm with the plain version of SiloBN, also incorporating distillation loss; the results of this analysis are presented in Tab. 8 and show analogous performance of ClAvBN and SiloBN.

Table 6. Federated self training setting (FDA): results

Number of clients	Teacher update	Train mIoU	SameDom mIoU	DiffDom mIoU
2	∞	17.97	17.77	14.22
8	∞	**18.97**	**18.49**	**16.17**
2	1	9.93	9.77	8.56
8	1	8.59	8.24	8.12
2	10	18.30	18.03	14.03
8	10	13.15	12.83	12.27
2	24	18.29	17.82	14.23
8	24	14.91	14.65	13.96

Table 7. Clustered Averaged BN (ClAvBN): results

Number of clients	Pre-trained with FDA	Teacher update	Train mIoU	SameDom mIoU	DiffDom mIoU
2	false	∞	29.67	29.56	23.27
4	false	∞	29.32	29.24	22.88
2	**false**	**10**	**29.86**	**29.57**	**23.17**
4	false	10	29.48	29.39	22.75
2	true	∞	25.24	25.45	19.46
4	true	∞	25.27	25.43	19.29
2	true	10	25.83	26.11	20.12
4	true	10	25.23	25.41	18.73

Table 8. SiloBN: results

Number of clients	Pre-trained with FDA	Teacher update	Train mIoU	SameDom mIoU	DiffDom mIoU
2	false	∞	29.19	29.07	21.98
4	false	∞	29.21	29.06	22.09
2	false	10	29.42	29.10	22.11
4	**false**	**10**	**29.67**	**29.36**	**22.56**
2	true	∞	25.00	25.25	18.24
4	true	∞	24.87	25.06	18.13
2	true	10	25.56	25.81	18.22
4	true	10	25.15	25.29	18.00

Even with limited resources and reduced datasets, the results obtained with ClAvBN are satisfactory and show promising potential in a real-world scenario.

5. Conclusions

In this paper, we presented a semantic segmentation framework for autonomous driving. We set a centralized baseline and used it for our experiments in different settings.

In particular, we started from a plain FedAvg implementation and subsequently enriched the model with domain adaptation and self-learning techniques.

We also proposed our ClAvBN method to further address statistical heterogeneity in a federated environment. We presented our results and the comparison with SiloBN.

We believe that our algorithm could be better exploited with the right computational resources and provide further improvements in a real-world scenario, where clients' datasets are more heterogeneous, containing various domains at the same time.

References

[1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017. 1

[2] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers, 2021. 1

[3] Gabriel L. Oliveira, Wolfram Burgard, and Thomas Brox. Efficient deep models for monocular road segmentation. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4885–4891, 2016. 1

[4] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data, 2023. 1, 2, 3

[5] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, may 2020. 1, 2

[6] Lidia Fantauzzo, Eros Fani', Debora Caldarola, Antonio Tavera, Fabio Cermelli, Marco Ciccone, and Barbara Caputo. Feddrive: Generalizing federated learning to semantic segmentation in autonomous driving, 2022. 1

[7] Mathieu Andreux, Jean Ogier du Terrail, Constance Beguier, and Eric W. Tramel. Siloed federated learning for multi-centric histopathology datasets, 2020. 1, 3, 4

[8] Emanuele Alberti, Antonio Tavera, Carlo Masone, and Barbara Caputo. IDDA: a large-scale multi-domain dataset for autonomous driving. *CoRR*, abs/2004.08298, 2020. 1, 3, 4, 5

[9] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017. 1, 2, 3

[10] Shijie Hao, Yuan Zhou, and Yanrong Guo. A brief survey on semantic segmentation with deep learning. *Neurocomputing*, 406:302–321, 2020. 1, 2, 5

[11] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning, 2021. 2

[12] Sicheng Zhao, Xiangyu Yue, Shanghang Zhang, Bo Li, Han Zhao, Bichen Wu, Ravi Krishna, Joseph E. Gonzalez, Alberto L. Sangiovanni-Vincentelli, Sanjit A. Seshia, and Kurt Keutzer. A review of single-source deep unsupervised visual domain adaptation, 2020. 2, 4

[13] Yuang Liu, Wei Zhang, and Jun Wang. Source-free domain adaptation for semantic segmentation, 2021. 2, 4

[14] 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. 2, 4

[15] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. 3

[16] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games, 2016. 4, 5

[17] Yanchao Yang and Stefano Soatto. Fda: Fourier domain adaptation for semantic segmentation, 2020. 4

[18] Yunsheng Li, Lu Yuan, and Nuno Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation, 2019. 4, 7

[19] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. 5