FedCV: A Federated Learning Framework for Diverse Computer Vision Tasks

Chaoyang He[1], Alay Dilipbhai Shah[1], Zhenheng Tang[2], Di Fan[1]
Adarshan Naiynar Sivashunmugam[1], Keerti Bhogaraju[1], Mita Shimpi[1], Li Shen[3], Xiaowen Chu[2],
Mahdi Soltanolkotabi[1], Salman Avestimehr[1]
University of Southern California[1], Hong Kong Baptist University[2], Tencent AI Lab[3]
{chaoyang.he, avestime}@usc.edu

Abstract

Federated Learning (FL) is a distributed learning paradigm that can learn a global or personalized model from decentralized datasets on edge devices. However, in the computer vision domain, model performance in FL is far behind centralized training due to the lack of exploration in diverse tasks with a unified FL framework. FL has rarely been demonstrated effectively in advanced computer vision tasks such as object detection and image segmentation. To bridge the gap and facilitate the development of FL for computer vision tasks, in this work, we propose a federated learning library and benchmarking framework, named `FedCV`*, to evaluate FL on the three most representative computer vision tasks: image classification, image segmentation, and object detection. We provide non-I.I.D. benchmarking datasets, models, and various reference FL algorithms. Our benchmark study suggests that there are multiple challenges that deserve future exploration: centralized training tricks may not be directly applied to FL; the non-I.I.D. dataset actually downgrades the model accuracy to some degree in different tasks; improving the system efficiency of federated training is challenging given the huge number of parameters and the per-client memory cost. We believe that such a library and benchmark, along with comparable evaluation settings, is necessary to make meaningful progress in FL on computer vision tasks.* `FedCV` *is publicly available:* `https://github.com/FedML-AI/FedCV`.

1. Introduction

FL has the potential to rescue many interesting computer vision (CV) applications which centralized training cannot handle due to various issues such as privacy concerns (e.g. in medical settings), data transfer and maintenance costs (most notably in video analytic), or sensitivity of proprietary data (e.g. facial recognition) [30]. In essence, FL is an art of trade-offs among many optimization objectives, including improving model accuracy and personalization [75, 14, 42, 55, 28, 20, 10], system efficiency (communication and computation) [23], robustness to attacks [67, 2, 15, 1, 68, 7, 58, 13, 7, 56, 49], and privacy [4, 16, 48, 53, 45, 62, 60, 57, 71, 61]. There has been steady progress in FL algorithmic research to achieve these goals.

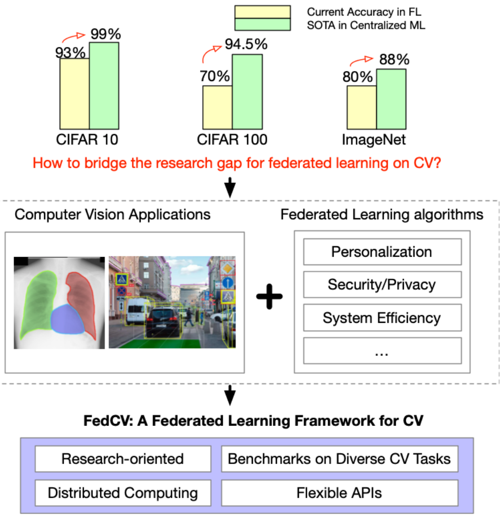

Current Accuracy in FL
SOTA in Centralized ML
93%
99%
70%
94.5%
80%
88%
CIFAR 10
CIFAR 100
ImageNet
How to bridge the research gap for federated learning on CV?
Computer Vision Applications
Federated Learning algorithms
Personalization
Security/Privacy
System Efficiency
...
FedCV: A Federated Learning Framework for CV
Research-oriented
Benchmarks on Diverse CV Tasks
Distributed Computing
Flexible APIs

Figure 1. Our philosophy of federated learning on computer vision: connecting the algorithmic FL research and CV application-drive research with an unified research framework.

However, the research gap between computer vision (CV) and federated learning (FL) is large. First, research in the FL community focuses almost exclusively on distributed optimization methods with small-scale datasets and models in image classification (see Table 16 in the Appendix), while the research trends in CV focus more on large-scale supervised/self-supervised pre-training [8] with efficient CNN [59] or Transformer models [11], which largely improves the performance of classification tasks on ImageNet and various downstream tasks such as object detection and image segmentation. Due to the lack of exploration in diverse tasks, model performance in FL is far behind that of centralized training.

Second, CV model training normally requires large-scale computing research in a distributed computing environment, but current FL algorithms are mostly published as standalone simulations, which further enlarges the research gap (e.g., the recently released FedVision library [40] only contains object detection and single GPU training).

Third, the efficacy of proposed FL algorithms on diverse CV tasks is still vague. When combined with multiple optimization objectives such as privacy, security, and fairness, the problem becomes even more challenging. Currently, only image classification in small-scale datasets and models has been evaluated in these algorithms (see Table 16 in the Appendix). Researchers may attempt to solve a specific problem in realistic CV tasks by designing new algorithms, but the current research community lacks such a library to connect diverse CV tasks with algorithmic exploration.

Due to these obstacles, there is an urgent need to bridge the gap between pure algorithmic research and CV application-driven research. Our philosophy to do so can be illustrated in Figure 1. Specifically, we design a unified federated learning library, named `FedCV`, to connect various FL algorithms with multiple important CV tasks, including image segmentation and object detection. Under this framework, the benchmark suite is provided to assist further research exploration and fair comparison. To approach a realistic federated dataset, we provide methods to partition the dataset into non-identical and independent distribution. Model-wise, we believe the best solution for CV is to improve pre-training for SOTA models with efficient federated learning methods, which requires us to design efficient and effective task-specific models with pre-trained models as the backbone in the FL setting. To reduce the learning curve and engineering burden for CV researchers, we provide various representative FL algorithms as one line, easy-to-use APIs. Most importantly, these APIs provide distributed computing paradigm, which is essential to accelerating the federated training of most CV models. Moreover, we also make the framework flexible in exploring algorithms with new protocols of distributed computing, such as customizing the exchange information among clients and defining specialized training procedures.

To demonstrate the ability of our framework and provide benchmarking experimental results, we run experiments in three computer visions: image classification, image segmentation, and object detection. Our benchmark study suggests that there are multiple challenges that deserve future exploration: many deep learning training tricks may not be directly applied to FL; the non-IID dataset actually downgrades the model accuracy to some degree in different tasks; improving the system efficiency of federated training is challenging given the huge number of parameters and the per-client memory cost. We hope `FedCV` will serve as an easy-to-use platform for researchers to explore diverse research topics at the intersection of computer vision and federated learning, such as improving models, systems, or federated optimization methods.

2. Related Works

[27] is the first work that applies federated learning to a real-world image dataset, Google Landmark [69], which has now become the standard image dataset for federated learning research. [6, 33, 35] apply federated learning on medical image segmentation tasks, which aims at solving the issue in which the training data may not be available at a single medical institution due to data privacy regulations. In the object detection task, [74] proposes a KL divergence method to mitigate model accuracy loss due to non-I.I.D. Our work is closely related to FedVision [40], a federated learning framework for computer vision. It supports object detection in the smart city scenario using models including FastRCNN and YOLOv3. However, FedVision only supports the FedAvg algorithm and single-GPU training. Our FedCV platform provides diverse computer tasks and various FL algorithms. For federated learning in other application domains, we refer to the comprehensive vision paper [30].

3. Preliminary and Challenges

Federated learning (FL) is a distributed learning paradigm that can leverage a scattered and isolated dataset to train a global or personalized model for each client (participant) while achieving privacy preservation, compliance with regulatory requirements, and savings on communication and storage costs for such large edge data. The most straightforward formulation is to assume all clients need to collaboratively train a global model. Formally, the objective function is as follows:

$$\begin{aligned} \min_{\boldsymbol{W}} F(\boldsymbol{W}) &\overset{\text{def}}{=} \min_{\boldsymbol{W}} \sum_{k=1}^{K} \frac{N^{(k)}}{N} \cdot f^{(k)}(\boldsymbol{W}) \\ f^{(k)}(\boldsymbol{W}) &= \frac{1}{N^{(k)}} \sum_{i=1}^{N^{(k)}} \ell(\boldsymbol{W}; \boldsymbol{X}_i, y_i) \end{aligned} \tag{1}$$

In computer vision, $\boldsymbol{W}$ can be any CNN or Transformer model (e.g., ViT). $f^{(k)}(\boldsymbol{W})$ is the kth client's local objective function that measures the local empirical risk over the heterogeneous dataset $\mathcal{D}^k$. ℓ is the loss function of the global CNN model. For the image classification task, ℓ is the cross-entropy loss.

To solve this federated optimization problem, FedAvg is the first federated optimization algorithm to propose the concept of FL. To better understand the challenges of FL on CV, we rewrite its optimization process in Algorithm 1 with annotations. As we can see, there are several clear characteristics that distinguish FL from conventional distributed training in a sealed data center:

1. Data heterogeneity and label deficiency at the edge. In conventional distributed training, centralized datasets are

Algorithm 1 `FedAvg` Algorithm: A Challenge Perspective

```
Initialization: there is a number of clients in a network; the client $k$ has local dataset $\mathcal{D}^k$ ; each client's local model is initialized as $\boldsymbol{W}_0$;

Server_Executes:
for each round $t = 0, 1, 2, \dots$ do
    $S_t \leftarrow$ (sample a random set of clients)
    for each client $k \in S_t$ in parallel do
        $\boldsymbol{W}_{t+1}^k \leftarrow \text{ClientUpdate}(k, \boldsymbol{W}_t)$
    end for
    $\boldsymbol{W}_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} \boldsymbol{W}_{t+1}^k$
end for

Client_Training($k, \boldsymbol{W}$): // Run on client $k$
$\mathcal{B} \leftarrow$ (split $\mathcal{D}^k$ into batches)
for each local epoch $i$ with $i = 1, 2, \cdots$ do
    for batch $b \in \mathcal{B}$ do
        $\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta \nabla_{\boldsymbol{W}} F(\boldsymbol{W}; b)$
    end for
end for
return $\boldsymbol{W}$ to server
```

evenly distributed into multiple compute nodes for parallel computing, while in FL, the data is generated at the edge in a property of non-identical and independent distribution (non-I.I.D.). For example, in the CV scenario, smartphone users generate images or videos with distinct resolutions, qualities, and contents due to differences in their hardware and user behaviors. In addition, incentivizing users to label their private image and video data is challenging due to privacy concerns.

2. System constraints and heterogeneity. Training large DNN models at the edge is extremely challenging even when using the most powerful edge devices. In terms of memory, edge training requires significantly more memory than the edge inference requires. The bandwidth for edge devices is smaller than that of distributed training in the data center environment (InfiniBand can be used); the edge devices normally do not have GPU accelerators. What is even worse is that these system abilities are heterogeneous due to diverse hardware configurations.

3. Robustness and Privacy. Since federated training is not in a sealed data center environment, as is the traditional distributed training, it is easier to manipulate the data and model poisoning attacks. Therefore, making the training algorithm robust against attacks is also an important research direction in FL. In addition, although privacy preservation is one of the main goals, researchers also demonstrate that the exchanged gradient between the client and the server may, to some degree, lead to privacy leaks. More privacy-preserving techniques must be evaluated on various computer vision applications.

4. `FedCV` Design

To solve these challenges in diverse CV tasks, we need a flexible and efficient distributed training framework with easy-to-use APIs, benchmark datasets and models, and reference implementations for various FL algorithms.

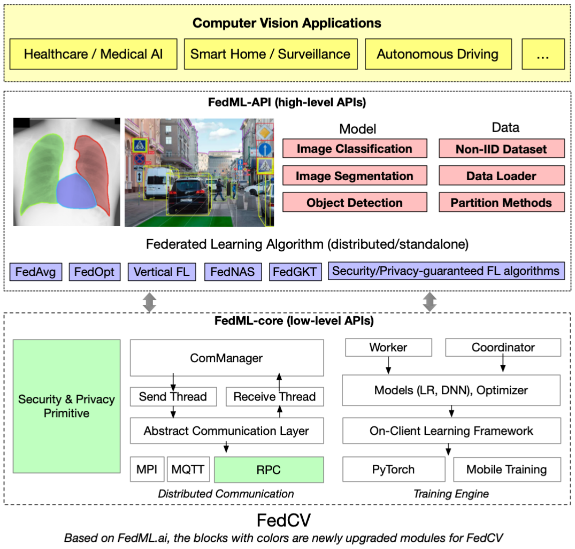

Computer Vision Applications
Healthcare / Medical AI
Smart Home / Surveillance
Autonomous Driving
...
FedML-API (high-level APIs)
Model
Data
Image Classification
Non-IID Dataset
Image Segmentation
Data Loader
Object Detection
Partition Methods
Federated Learning Algorithm (distributed/standalone)
FedAvg
FedOpt
Vertical FL
FedNAS
FedGKT
Security/Privacy-guaranteed FL algorithms
FedML-core (low-level APIs)
Security & Privacy Primitive
ComManager
Send Thread
Receive Thread
Abstract Communication Layer
MPI
MQTT
RPC
Distributed Communication
Worker
Coordinator
Models (LR, DNN), Optimizer
On-Client Learning Framework
PyTorch
Mobile Training
Training Engine
FedCV
Based on FedML.ai, the blocks with colors are newly upgraded modules for FedCV

Figure 2. Overview of `FedCV` System Architecture Design

To bridge the gap between CV and FL research, we have designed an open-source federated learning system for computer vision, named `FedCV`. `FedCV` is built based on the `FedML` research library [24], which is a widely used FL library that only support image classification, ResNet and simple CNN models. The system architecture of `FedCV` is illustrated in Figure 2. To distinguish FedCV from FedML, we color-code the modules specific to `FedCV`. `FedCV` makes the following contributions:

Benchmark Suite for Diverse CV Tasks: `FedCV` supports three computer vision tasks: image classification, image segmentation, and object detection. Related datasets and data loaders are provided. Users can either reuse our data distribution or manipulate the non-I.I.D. by setting hyperparameters. Models are curated for benchmark evaluation. More details of the benchmark suite are given in Section 5.

Reference Implementation for Representative FL Algorithms: Currently, `FedCV` includes the standard implementations of multiple state of the art FL algorithms: Federated Averaging (FedAvg) [44], FedOpt (server Adam) [50], FedNova (client optimizer) [66], FedProx [54], FedMA [65], as well as some novel algorithms that have diverse training paradigms and network typologies, including FedGKT (efficient edge training) [23], Decentralized FL [25], Vertical Federated Learning (VFL) [72], Split Learning [19, 64], Federated Neural Architecture Search (FedNAS) [22], and

Turbo-Aggregate [57]. These algorithms support multi-GPU distributed training, which enables training to be completed in a reasonable amount of time. Note that most published FL optimization algorithms are based on standalone simulations, which lead to a extremely long training time. In this paper, we bridge this gap and make the CV-based FL research computationally affordable.

Easy-to-use APIs for Algorithm Customization: With the help of the `FedML` API design, `FedCV` enables diverse networks, flexible information exchange among workers/clients, and various training procedures. We can easily implement new FL algorithms in a distributed computing environment. We defer API design details to the Appendix.

Other Functionality: We support several development tools to simplify the research exploration. Specifically, researchers can load multiple clients into a single GPU, which scales up the client number with fewer GPUs, although there may be GPU contention among processes; in the lowest layer, `FedCV` reuses `FedML-core` APIs but further supports tensor-aware RPC (remote procedure call), which enables the communication between servers located at different data centers (e.g., different medical institutes); enhanced security and privacy primitive modules are added to support techniques such as secure aggregation in upper layers.

5. `FedCV` Benchmark Suite: Datasets, Models, and Algorithms

Table 1. Summary of benchmark suite.

Task	Dataset	Model
Image Classification	CIFAR-100 GLD-23k[69]	EfficientNet[59] MobileNet[26] ViT[12]
Image Segmentation	PASCAL VOC[21]	DeeplabV3+[38] UNet[47]
Object Detection	COCO[39]	YOLOv5[29]
FL Algorithms	FedAvg, FedOpt ...	

We summarize the benchmark suite in `FedCV` in Table 1, and introduce such a curated list task-by-task as follows:

Image Classification. The curated datasets are Google Landmarks Dataset 23k (GLD-23K) [69] and CIFAR-100 dataset [32] with non-I.I.D partition. GLD-23K dataset is suggested by Tensorflow Federated [18], a natural federated dataset from smartphone users. For the model, we suggest EfficientNet [59] and MobileNet-V3 [26], which are two lightweight CNNs. Since the attention-based Transformer model has become a trending model in CV, we suggest Vision Transformer (ViT) [12] (ViT-B/16) to conduct experiments. As the research progresses, we may be able to support more efficient Transformers.

Image Segmentation. We use the augmented PASCAL VOC dataset with annotations from 11355 images [21]. These images are taken from the original PASCAL VOC 2011 dataset which contains 20 foreground object classes and one background class. For models, DeepLabV3+ [38] and U-Net [47] are supported since they are representative image segmentation models in centralized training. In our experiments, we utilize ResNet-101 and MobileNet-V2 as two backbones of DeepLabV3+ and U-Net.

Object Detection. We use the COCO [39] dataset since it contains realistic images that include detecting and segmenting objects found in everyday life through extensive use of Amazon Mechanical Turk. We then use YOLOv5 [29], an optimized version of YOLOv4 [3] as the baseline model. It outperforms all the previous versions and approaches EfficientDet Average Precision(AP) with higher frames per second (FPS). In YOLOv5, four network models (YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x) with different network depths and widths are provided to cater to various applications. We use these four models as the pretrained models.

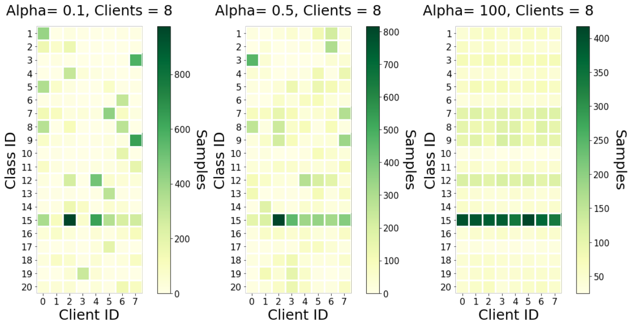


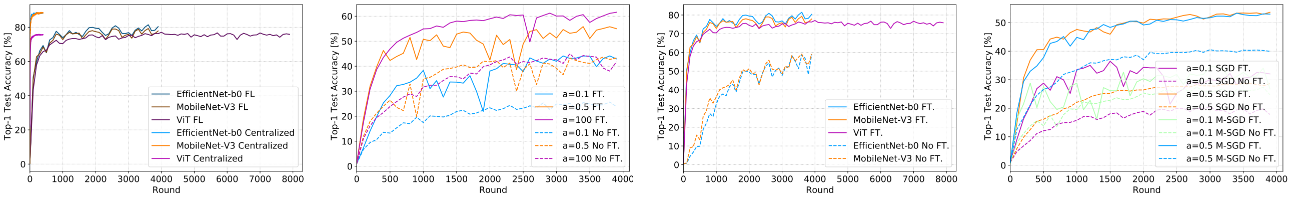
Figure 3. Non I.I.D. data distribution on augmented PASCAL VOC dataset with different α values. Each square represents the number of data samples of a specific class for a client.

Non-I.I.D. Preprocessing. Our non-I.I.D. partition method is Latent Dirichlet Allocation (LDA) [63], which is a common practice in FL research to obtain synthetic federated datasets. As an example, we visualize the non-I.I.D. in Figure 3. Further details of the dataset partition can be found in the Appendix. For all datasets, we provide data downloading scripts and data loaders to simplify the data preprocessing. Note that we will update regularly to support new datasets and models.

The supported FL algorithms include FedAvg, FedOpt and many other representative algorithms. We provide a full list and description in the appendix.

6. Experiments

In this section, we present the experimental results on image classification, image segmentation, and objective detection tasks on varying deep learning models and datasets.



(a) Three models on GLD-23k (b) EfficientNet on CIFAR-100 (c) Three models on GLD-23k (d) EfficientNet on CIFAR-100

Figure 4. Experiments on classification task. Figure (a): Test accuracy on GLD-23k with FedAvg and centralized training. The maximum number of epochs of centralized training is 400. The learning rate is 0.3 for EfficientNet and MobileNet of centralzed training, 0.03 for ViT of centralized traning, and 0.1 for all three models of FedAvg. Here the learning scheduler is not used for FedAvg. Figure (b): Test accuracy of FedAvg with EfficientNet on CIFAR-100 with different Non-IID degree. Hyper-parameters of this figure are set as Table 11 in the appendix. Here, FT. means fine-tuning, i.e. loading a pretrained model and doing FedAvg on this model. Figure (c): Test accuracy of FedAvg with EfficientNet, MobileNet and ViT on GLD-23K, with/without fine-tuning. Hyper-parameters of this figure can be found in Tables 13, 14 and 15 in appendix. Figure (d): Test accuracy of FedAvg with EfficientNet on CIFAR-100, with/without fine-tuning, SGD or momentum SGD. Hyper-parameters of this figure are set as Table 11 and Table 12 in appendix. Here, M-SGD means using local SGD with momentum.

6.1. Image Classification

6.1.1 Implementation Details

For image classification, the client number per round is 10. All experiments are conducted on a computing cluster with GTX 2080Ti. Each client has one GPU and the communication bandwidth is 10 Gbps. We conduct extensive experiments with EfficientNet [59], MobileNet V3 [26] and ViT [12] on CIFAR-100 [32], and CLD-23K [69][18] datasets. The hyper-parameter settings are listed in the Appendix.

6.1.2 Experimental Results

The main experimental results are presented in table 2. Below, we provide detailed comparisons of the implemented classification models on the proposed FedCV platform.

Dataset	Model	Partition	LR	Acc
CIFAR-100	EfficientNet	Cent.	0.01	0.6058
		a=0.1	0.003	0.4295
		a=0.5	0.01	0.5502
		a=100.0	0.003	0.6158
	MobileNet V3	Cent.	0.01	0.5785
		a=0.1	0.003	0.4276
		a=0.5	0.01	0.4691
		a=100.0	0.003	0.5203
GLD-23k	EfficientNet	Cent.	0.3	0.8826
		Non-IID	0.1	0.8035
	MobileNet V3	Cent.	0.3	0.8851
		Non-IID	0.03	0.7841
	ViT-B/16	Cent.	0.03	0.7565
		Non-IID	0.03	0.7611

Table 2. Summary of experimental results on image classification. In this table, Cent. refers to centralized training. For all experiments, we use a batch size of 256 for centralized training and 32 for FedAvg. We use a linear learning rate scheduler with a step size of 0.97 for centralized training, but no scheduler for FedAvg. We use momentum SGD with momentum coefficient of 0.9 for all experiments. More experimental results on other settings can be found in Tables 11, 12, 13, 14 and 15 in the **Appendix**.

GLD-23k NonIID vs. IID. Figure 4(a) shows that the test accuracy of centralized training with EfficientNet and MobileNet outperforms FedAvg training by ten percent. And for the ViT, the accuracy of centralized training is similar with FedAvg.

Impacts of different degrees of Non-IID. Figure 4(b) and Figure 15 (in Appendix) show the influence of different degrees of Non-IID on the training performance of EfficientNet and MobileNetV3. Experimental results align with the results of LDA [63]. A higher α (i.e., lower degree of Non-IID) causes the test accuracy to increase.

Fine-tuning vs. training from scratch. Figure 4(b), Figure 15 in appendix, and Figure 4(c) show that the performance of fine-tuning is more effective than training from scratch. For the convergence speed, fine-tuning can achieve a test accuracy of 60%, nearly 20× faster than training from scratch. After training is completed, fine-tuning outperforms training from scratch by about 20 percent.

Momentum SGD vs SGD. Figure 4 (d), and Figures 14(a)-(b) (in appendix) show that SGD with momentum cannot guarantee better performance than vanilla SGD. When using EfficientNet On CIFAR-100 dataset of $\alpha = 0.5$, momentum SGD has similar performance to SGD with fine tuning, but with a much higher test accuracy than SGD training from scratch. With $\alpha = 0.1$, the performance of momentum SGD is not significantly influenced by fine-tuning, whereas vanilla SGD can see significant improvement.

Learning rate scheduler. Figure 5, and Figure 14(c)-(d) (in appendix) show an interesting result in which the linear learning rate decay may not improve the performance, and even leads to performance decrease. One reason may be that in the last training epochs, each client cannot converge with too small learning rate. However, learning rate decay is able to make the training process more stable. For cases where $\alpha = 0.1$ and $\alpha = 0.5$, four curves of linear learning rate decay are smoother than without learning rate decay.

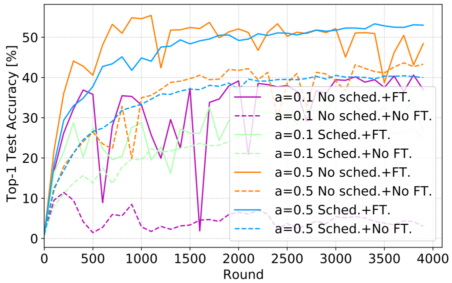


Figure 5. Test accuracy on CIFAR-100 with EfficientNet trained with momentum SGD, with/without fine-tuning and learning rate scheduler. Hyper-parameters are set as Table 11 in appendix. Here, Sched. means using learning rate scheduler with step size of 0.99.

Model	MobileNet-V3	EfficientNet	ViT-B/16
Params	4M	3.8M	81.8M
MMACs	2137	3796.4	16067.5
Comm rounds	4000	4000	8000
Total time	5.16h	5.05h	31.1h
Comm cost	0.278h	0.264h	5.68h

Table 3. Efficiency of training MobileNet V3, EfficientNet, Vit models with FedAvg. In this table, MMACs refer to the forward computation for one sample. Total time refers to the entire training time plus evaluating time; we evaluate the model per 100 communication rounds. For the MobileNet and EfficientNet, the number of total communication rounds is 4000, and for ViT it is 8000. The communication cost is theoretically calculated out. Note the actual communication time should be larger than the theoretical communication time due to the straggler problem and other overhead.

Efficiency analysis. We summarize the system performance of three models in Table 3, which demonstrate that if we train a big deep learning model such as ViT in the federated setting, there exists a huge communication overhead compared with small models. Furthermore, in the real federated environment, the communication bandwidth could be even worse.

6.2. Image Segmentation

6.2.1 Implementation Details

For the image segmentation, we train DeeplabV3+ and U-Net within the FedCV platform, in which the number of clients involved in each round of image segmentation are either 4 or 8. These studies are carried out on a computing cluster with a Quadro RTX 5000 graphics card. Each client has one GPU, with a 15.754 GB/s communication bandwidth.

Table 4 comprises a list of models and hyper-parameters we explored for evaluating performance of segmentation tasks in the federated setting. Note that we use following abbreviation throughout our analysis: **TT:** Training Type for Backbone. There are three strategies that we use for training the backbone. (i) Fine-Tuning **(FT)**: We start with a ImageNet-pretrained backbone and fine-tune it for our task. (ii) Freezed Backbone **(FZ)**: Similarly to FT, we start with ImageNet[52] pretrained backbone but do not train or fine-tune the backbone at all to save on computational complexity. (iii) Scratch Federated Learning **(SFL)**: Training the entire architecture end-to-end starting from scratch.

Dataset	Augemented PASCAL VOC
Model	DeeplabV3+, UNet
Backbone	ResNet-101, MobileNetV2
Backbone TT	FT, FZ, SFL
Batch Size Range	4 to 16
LR Range	0.0007 to 0.1

Table 4. Dataset, models and hyper-parametesr choices for federated image segmentation task.

Model	**Backbone (TT)**	**DD**	**C**	**mIOU**
DeeplabV3+	ResNet-101 (FT)	IID	4	77.9%
DeeplabV3+	ResNet-101 (FT)	N-IID	4	76.47%
DeeplabV3+	ResNet-101 (FT)	N-IID	8	75.69%
DeeplabV3+	ResNet-101 (SFL)	N-IID	4	75.44%
DeeplabV3+	ResNet-101 (FZ)	N-IID	4	68.24%
DeeplabV3+	MobileNetV2 (FT)	N-IID	4	69.31%
UNet	ResNet-101 (FT)	IID	4	75.14%
UNet	ResNet-101 (FT)	N-IID	4	74.34%
UNet	ResNet-101 (FT)	N-IID	8	73.65%
UNet	ResNet-101 (SFL)	N-IID	4	74.2%
UNet	ResNet-101 (FZ)	N-IID	4	51.19%
UNet	MobileNetV2 (FT)	N-IID	4	66.14%

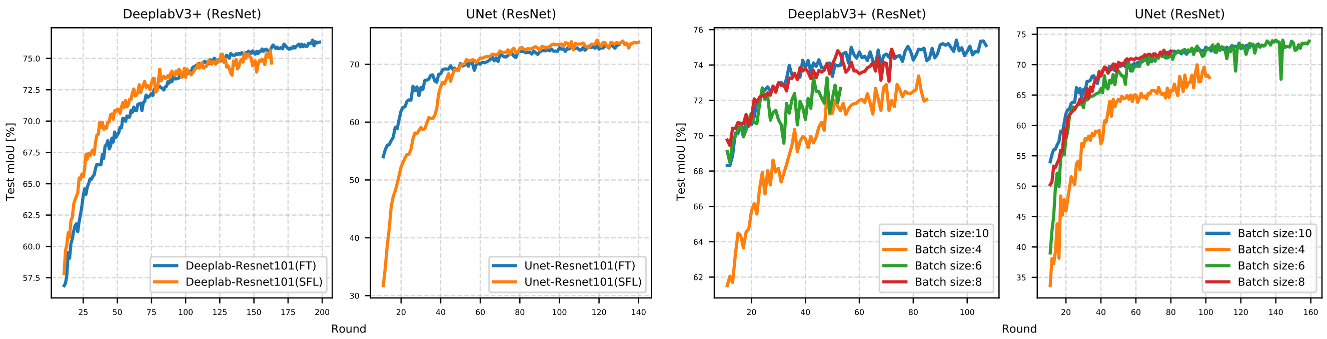
Table 5. Summary of test results on Pascal VOC dataset for federated image segmentation task. **DD:** Data Distribution Type. **N-IID**: Heterogeneous distribution with partition factor α=0.5 **IID:** Homogeneous distribution. **C:** Number of Clients

6.2.2 Experimental Results

In this section, we analyze and discuss our results for image segmentation tasks in the federated setting. We summarize our top results in Table 5 for a variety of training setups.

Backbone Training vs. Fine-Tuning. Figure 6(a) shows that pre-trained backbones coupled with fine-tuning results in only a slightly better performance (less than 2%) compared to training from scratch, which indicates that while pre-trained backbones aid in federated image segmentation accuracy, they are not necessary. This finding opens the door to advanced tasks such as medical imaging, where pre-trained backbones may not be useful and end-to-end training from scratch is the only viable alternative.

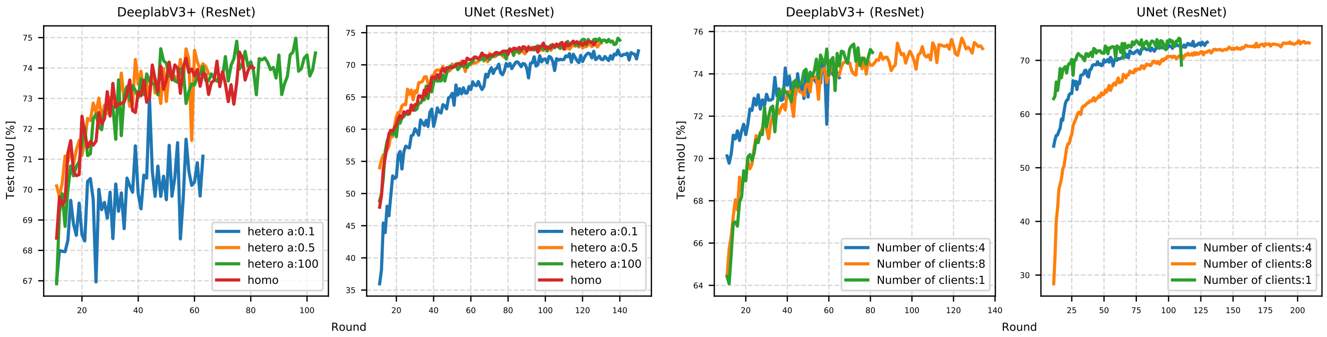
Batch Size vs. Memory Trade-Off. Figure 6(b) and Table 6 show that a smaller batch size, such as 4 instead of 10, reduces memory by roughly a factor of two while sacrificing nearly 2% accuracy. This is an important trade off to make because edge devices in a federated learning setup may have constrained memory.



(a) Experiments with/without fine-tuning

(b) Experiments with varying batch sizes

Figure 6. Performance evaluation of segmentation tasks on Pascal VOC dataset. Figure (a): Comparing performance of DeeplabV3+ and UNet models with fine-tuning (FT) Resnet101 backbones against training from scratch (SFL) . Figure (b): Evaluating performance of DeeplabV3+ and UNet models on various batch sizes.



(a) Experiments on various partition factors

(b) Experiments on varying number of clients

Figure 7. Performance evaluation of segmentataion task on Pascal VOC dataset. Figure (a): Evaluating performance of DeeplabV3+ and UNet models with Resnet101 as a backbone on various partition factors (a).Figure (b): Evaluation performance of DeeplabV3+ and UNet models with Resnet-101 as backbone on varying number of clients.

Model	**Backbone**	**BS**	**Memory**	**mIOU**
DeeplabV3+	ResNet-101	4	6119M	72.38%
DeeplabV3+	ResNet-101	6	8009M	73.28%
DeeplabV3+	ResNet-101	8	10545M	74.89%
DeeplabV3+	ResNet-101	10	13084M	75.5%
UNet	ResNet-101	4	6032M	71.54%
UNet	ResNet-101	6	8456M	71.89%
UNet	ResNet-101	8	10056M	72.4%
UNet	ResNet-101	10	12219M	73.55%

Table 6. Performance and memory analysis for various batch size of segmentation models on Pascal VOC Dataset. **BS:** Batch Size

Data distribution impact analysis. For various partition values α, Figure 3 depicts the distribution of classes among clients. Even when the partition factor changes from totally homogeneous to extremely heterogeneous, as shown in Figure 7 (a), the accuracy only degrades by about 2%. This further demonstrates that federated segmentation learning can instill enough generalization capability in local clients to allow them to perform well on unknown data, obviating the need for centralized or widely distributed data.

Resiliency in the face of increasing clients The number of rounds needed for the model to converge increases as the number of clients increases (see figure 7(b)). When compared to smaller client sizes, which are theoretically expected to perform better since each local client has more data points to train on, it has little effect on final accuracy after a sufficient number of rounds.

6.2.3 System Performance Analysis

ResNet is one of the most widely used backbones for encoder-decoder architecture in image segmentation tasks; however, it has a high computing cost that many edge devices might not be able to bear. There are two obvious ways to trim the cost down: (i) Freezing the pre-trained backbone; (ii) Plugging computationally efficient backbone (Eg. MobileNetV2). Figure 8 depicts the performance variance when one of the two described strategies is applied for backbones in DeeplabV3+ and UNet architectures for federated image segmentation. When compared to every other mix, the accuracy of ResNet-101 backbone is demonstrably higher. On the other hand, as shown in Table 7, the alternatives are extremely efficient at the cost of performance degradation.

Model	Backbone (TT)	Dataset	Params	FLOPS	Memory (BS)	Total Time
DeeplabV3+	ResNet-101 (FT)	PASCAL VOC	59.34M	88.85G	13084M (10)	14.16h
DeeplabV3+	ResNet-101 (FZ)	PASCAL VOC	16.84M	88.85G	7541M (16)	23.59h
DeeplabV3+	MobileNetV2 (FT)	PASCAL VOC	5.81M	26.56G	12104M (16)	20.5h
UNet	ResNet-101 (FT)	PASCAL VOC	51.51M	62.22G	12219M (10)	14.5h
UNet	ResNet-101 (FZ)	PASCAL VOC	9.01M	62.22G	7687M (16)	51.11h
UNet	MobileNetV2 (FT)	PASCAL VOC	7.91M	14.24G	11706M (16)	22.03h

Table 7. System performance chart of segmentation network architectures we considered. **TT:** Training Type. **BS:** Batch Size

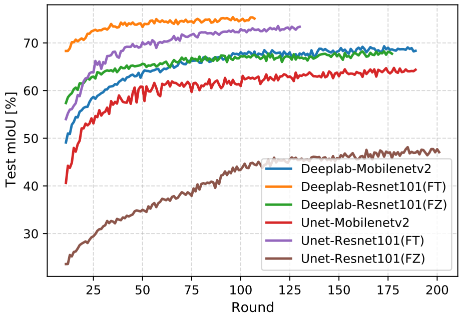
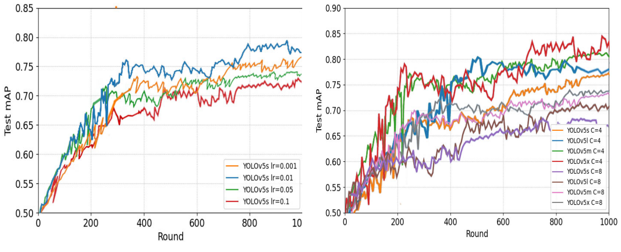


Figure 8. Performance comparison of DeeplabV3+ and UNet with ResNet101 and MobileNetV2 as backbones. DeeplabV3+ (Resnet101) reaches a better accuracy compared to other alternatives. **FT:** Fine-Tuning Backbone. **FZ:** Freezed Backbone

6.3. Object Detection

6.3.1 Implementation Details

For object detection, we use pre-trained YOLOV5 for federated learning experiments with the FedAvg algorithm. The client number we used include 4 and 8 for performance comparison. Each client was run at one GPU (NVIDIA V100). The metric in our experiments is mAP@0.5 (mean average precision with a threshold of 0.5 for IOU).



(a) Different learning rates (b) Different number of clients

Figure 9. Experiments on detection tasks on varying learning rates and number of clients. Figure (a): Non-IID data comparsion with different learning rate. Figure (b): Non-IID data comparsion with different number of clients.

6.3.2 Experimental Results

Learning rate. In the federated setting, different learning rates are evaluated. While keeping the other hyperparameters (e.g., client number is set to 4), we notice that lr=0.01 can have a better result compared to the other choices from Figure 9 (a).

Non-I.I.D. evaluation. For the client numbers 4 and 8, we use the partition method introduced in the appendix to obtain synthetic federated datasets. We found that when using YOLOv5, it is difficult for FedAvg to reach the same results as that of centralized for Non-IID dataset. Figure 9 shows there is a large gap between centralized training and FedAvg-based federated training. In centralized training of YOLOv5, test mAP of all four model variants is over 0.95 [29], whereas the best accuracy in the federated setting is smaller than 0.85. The main reason is that the optimizer and training tricks used in centralized training could not be directly transplanted to the FL framework, indicating that further research for object detection in the federated setting is required.

Evaluation on different number of clients. We also show the performance with different clients among 4 models in Figure 9 (b). Results showing that $C = 8$ has a lower performance compared to the $C = 4$.

System performance analysis. Table 8 summarizes the system performance of four different model variants. We can see that as the network structure depth and width increased among the four models, the model performed well with a better mAP.

Model	Layers	Parameters	FLOPS	Total Time
YOLOv5s	283	7.27M	17.1G	25.1h
YOLOv5m	391	21.4M	51.4G	49.3h
YOLOv5l	499	47.1M	115.6G	73.5h
YOLOv5x	607	87.8M	219.0G	92.4h

Table 8. System performance of YOLOv5

7. Conclusion

In this work, we propose an easy-to-use federated learning framework for diverse computer vision tasks, including image classification, image segmentation, and object detection, dubbed `FedCV`. We provide several non-IID benchmarking datasets, models, and various reference FL algorithms. We hope that `FedCV` can open doors for researchers to develop new federated algorithms for various computer vision tasks.

References

[1] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948, 2020.

[2] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pages 634–643, 2019.

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

[4] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for federated learning on user-held data. *arXiv preprint arXiv:1611.04482*, 2016.

[5] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. *arXiv e-prints*, page arXiv:1602.05629, Feb. 2016.

[6] Qi Chang, Hui Qu, Yikai Zhang, Mert Sabuncu, Chao Chen, Tong Zhang, and Dimitris Metaxas. Synthetic Learning: Learn From Distributed Asynchronized Discriminator GAN Without Sharing Medical Image Data. *arXiv e-prints*, page arXiv:2006.00080, May 2020.

[7] Chien-Lun Chen, Leana Golubchik, and Marco Paolieri. Backdoor attacks on federated meta-learning. *arXiv preprint arXiv:2006.07026*, 2020.

[8] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2020.

[9] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data. 2019.

[10] Canh T Dinh, Nguyen H Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. *arXiv preprint arXiv:2006.08848*, 2020.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[13] David Enthoven and Zaid Al-Ars. An overview of federated deep learning privacy attacks and defensive strategies. *arXiv preprint arXiv:2004.04676*, 2020.

[14] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

[15] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866*, 2018.

[16] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[17] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2672–2680, Cambridge, MA, USA, 2014. MIT Press.

[18] Google. Tensorflow federated datasets. https://www.tensorflow.org/federated/api_docs/python/tff/simulation/datasets.

[19] Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

[20] Filip Hanzely, Boxin Zhao, and Mladen Kolar. Personalized federated learning: A unified framework and universal optimization techniques. *arXiv preprint arXiv:2102.09743*, 2021.

[21] Bharath Hariharan, Pablo Arbelaez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *International Conference on Computer Vision (ICCV)*, 2011.

[22] Chaoyang He, Murali Annavaram, and Salman Avestimehr. Fednas: Federated deep learning via neural architecture search. *arXiv preprint arXiv:2004.08546*, 2020.

[23] Chaoyang He, Murali Annavaram, and Salman Avestimehr. Group knowledge transfer: Federated learning of large cnns at the edge. 2020.

[24] Chaoyang He, Songze Li, Jinhyun So, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, Li Shen, Peilin Zhao, Yan Kang, Yang Liu, Ramesh Raskar, Qiang Yang, Murali Annavaram, and Salman Avestimehr. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

[25] Chaoyang He, Conghui Tan, Hanlin Tang, Shuang Qiu, and Ji Liu. Central server free federated learning over single-sided trust social networks. *arXiv preprint arXiv:1910.04956*, 2019.

[26] A. Howard, M. Sandler, B. Chen, W. Wang, L. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le. Searching for mobilenetv3. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019.

[27] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Federated Visual Classification with Real-World Data Distribution. *arXiv e-prints*, page arXiv:2003.08082, Mar. 2020.

[28] Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. Personalized cross-silo federated learning on non-iid data. AAAI, 2021.

[29] Glenn Jocher. *YOLOv5*, 2020.

[30] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith

Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[31] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. *arXiv preprint arXiv:1910.06378*, 2019.

[32] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[33] Daiqing Li, Amlan Kar, Nishant Ravikumar, Alejandro F Frangi, and Sanja Fidler. Fed-Sim: Federated Simulation for Medical Imaging. *arXiv e-prints*, page arXiv:2009.00668, Sept. 2020.

[34] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. *arXiv preprint arXiv:1905.10497*, 2019.

[35] Wenqi Li, Fausto Milletarì, Daguang Xu, Nicola Rieke, Jonny Hancox, Wentao Zhu, Maximilian Baust, Yan Cheng, Sébastien Ourselin, M. Jorge Cardoso, and Andrew Feng. Privacy-preserving Federated Brain Tumour Segmentation. *arXiv e-prints*, page arXiv:1910.00962, Oct. 2019.

[36] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.

[37] Zhize Li, Dmitry Kovalev, Xun Qian, and Peter Richtárik. Acceleration for compressed gradient descent in distributed and federated optimization. *arXiv preprint arXiv:2002.11364*, 2020.

[38] George Papandreou Florian Schroff-Hartwig Adam Liang-Chieh Chen, Yukun Zhu. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018.

[39] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[40] Yang Liu, Anbu Huang, Yun Luo, He Huang, Youzhi Liu, Yuanyuan Chen, Lican Feng, Tianjian Chen, Han Yu, and Qiang Yang. Fedvision: An online visual object detection platform powered by federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13172–13179, 2020.

[41] Grigory Malinovsky, Dmitry Kovalev, Elnur Gasanov, Laurent Condat, and Peter Richtarik. From local sgd to local fixed point methods for federated learning. *arXiv preprint arXiv:2004.01442*, 2020.

[42] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

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

[44] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282, 2017.

[45] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706. IEEE, 2019.

[46] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. *arXiv preprint arXiv:1902.00146*, 2019.

[47] Thomas Brox Olaf Ronneberger, Philipp Fischer. U-net: Convolutional networks for biomedical image segmentation, 2015.

[48] Tribhuvanesh Orekondy, Seong Joon Oh, Yang Zhang, Bernt Schiele, and Mario Fritz. Gradient-leaks: Understanding and controlling deanonymization in federated learning. *arXiv preprint arXiv:1805.05838*, 2018.

[49] Saurav Prakash and A. Salman Avestimehr. Mitigating byzantine attacks in federated learning. *arXiv preprint arxiv: 2010.07541*, 2020.

[50] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

[51] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. *arXiv preprint arXiv:2007.07682*, page 12, 2020.

[52] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[53] Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. A generic framework for privacy preserving deep learning. *arXiv preprint arXiv:1811.04017*, 2018.

[54] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet Talwalkar, and Virginia Smith. On the convergence of federated optimization in heterogeneous networks. *ArXiv*, abs/1812.06127, 2018.

[55] Karan Singhal, Hakim Sidahmed, Zachary Garrett, Shanshan Wu, Keith Rush, and Sushant Prakash. Federated reconstruction: Partially local federated learning. *arXiv preprint arXiv:2102.03448*, 2021.

[56] Jinhyun So, Basak Guler, and A. Salman Avestimehr. Byzantine-resilient secure federated learning. *IEEE Journal on Selected Areas in Communication, Series on Machine Learning for Communications and Networks*, 2020.

[57] Jinhyun So, Basak Guler, and A Salman Avestimehr. Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning. *arXiv preprint arXiv:2002.04156*, 2020.

[58] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

[59] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. 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 6105–6114. PMLR, 09–15 Jun 2019.

[60] Aleksei Triastcyn and Boi Faltings. Federated learning with bayesian differential privacy. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2587–2596. IEEE, 2019.

[61] Aleksei Triastcyn and Boi Faltings. Federated generative privacy. *IEEE Intelligent Systems*, 2020.

[62] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, pages 1–11, 2019.

[63] Matthew Brown Tzu-Ming Harry Hsu, Hang Qi. Measuring the effects of non-identical data distribution for federated visual classification, 2019.

[64] Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

[65] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020.

[66] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *arXiv preprint arXiv:2007.07481*, 2020.

[67] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 2512–2520. IEEE, 2019.

[68] Wenqi Wei, Ling Liu, Margaret Loper, Ka-Ho Chow, Mehmet Emre Gursoy, Stacey Truex, and Yanzhao Wu. A framework for evaluating gradient leakage attacks in federated learning. *arXiv preprint arXiv:2004.10397*, 2020.

[69] T. Weyand, A. Araujo, B. Cao, and J. Sim. Google landmarks dataset v2 – a large-scale benchmark for instance-level recognition and retrieval. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2572–2581, 2020.

[70] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. Dba: Distributed backdoor attacks against federated learning. In *International Conference on Learning Representations*, 2019.

[71] Runhua Xu, Nathalie Baracaldo, Yi Zhou, Ali Anwar, and Heiko Ludwig. Hybridalpha: An efficient approach for privacy-preserving federated learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, pages 13–23, 2019.

[72] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.*, 10(2), Jan. 2019.

[73] Felix X Yu, Ankit Singh Rawat, Aditya Krishna Menon, and Sanjiv Kumar. Federated learning with only positive labels. *arXiv preprint arXiv:2004.10342*, 2020.

[74] Peihua Yu and Yunfeng Liu. Federated object detection: Optimizing object detection model with federated learning. In *Proceedings of the 3rd International Conference on Vision, Image and Signal Processing*, pages 1–6, 2019.

[75] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. Salvaging federated learning by local adaptation. *arXiv preprint arXiv:2002.04758*, 2020.

[76] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Trong Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. *arXiv preprint arXiv:1905.12022*, 2019.

Appendix

In the appendix, we provide more details of the benchmark suite and experiments.

7.1. Benchmark Suite

7.1.1 Dataset

CIFAR-100. The CIFAR-100 dataset [32] has 100 classes, of which each contains 600 images with size 32 × 32.

Google Landmarks Dataset 23k (GLD-23K) is a subset of Google Landmark Dataset 160k [69]. This GLD-23K dataset includes 203 classes, 233 clients, and 23080 images. We follow the setting of GLD-23K from Tensorflow federated [18]. We also provide the data loader for GLD-160K in our source code.

PASCAL VOC - Augmented. We use the augmented PASCAL VOC dataset with annotations from 11355 images [21]. These images are taken from the original PASCAL VOC 2011 dataset, which contains 20 foreground object classes and one background class.

COCO [39] is a dataset for detecting and segmenting objects found in everyday life through extensive use of Amazon Mechanical Turk.

7.1.2 Non-I.I.D. Partition and Distribution Visualization

For GLD-23K, we follow the setting of GLD-23K from Tensorflow federated [18], which means that the number of total clients is 233 for GLD-23K.

We make use of Latent Dirichlet Allocation (LDA) [63] method to partition CIFAR-100 and PASCAL VOC into non-I.I.D. dataset. The settings of our non-I.I.D. partition can be referred to table 9.

Comparing to the classification and segmentation task with one picture has one label using the LDA partition method, the object detection task always has several labels on one image. In this case, we take a different partition method. First, we calculate the frequency of each object in all images (one object is only counted one time even it occurs more than one time on one image). Second, we sort the objects' frequency and put the object with the highest frequency into one client. Finally, we sort the remaining pictures and repeat the first step until all the images have been assigned to the clients. Figure 10 shows the COCO non-IID data distribution on 8 clients. Different color represents different clients, and we could see every categories label data are non-IID distributed to different clients.

Figure 11(a)-(c) show the visualization of non-IID CIFAR-100 with different α. When the α increases, the similarity of data distribution becomes higher. Figure 11(d) visualize the data distribution of GLD-23K on 233 clients. For non-IID CIFAR-100 with a low α value and GLD-23K, we can see that the visualization matrix is sparse. Some clients have many samples of some labels, but few samples of some other labels. This makes training become much harder.

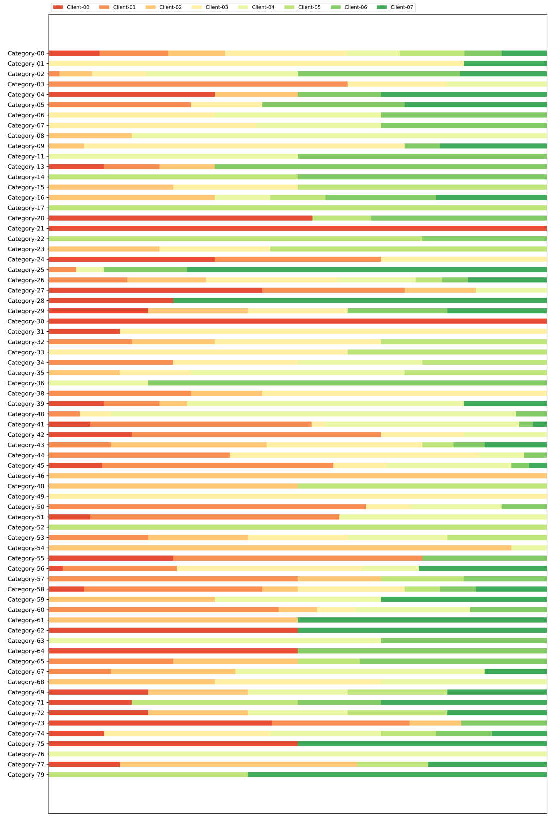
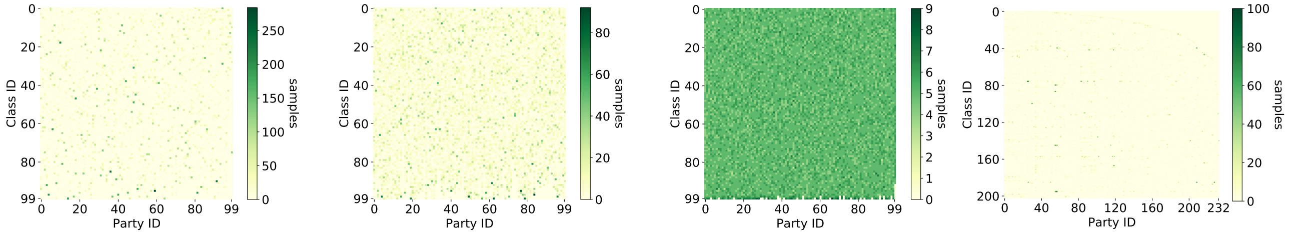


Figure 10. Non-IID data distribution on COCO dataset. Here we show the figure that COCO data has been distributed to 10 clients in a non-partition way. For the COCO dataset non-IID distribution, we have 80 categories and they are assigned to different clients showed by different color.

7.1.3 Models

EfficientNet [59] and **MobileNet-V3** [26] are two light weighted convolutional neural networks. They achieves the goal of improving accuracy and greatly reducing the amount of model parameters and calculations. In this paper, we use EfficientNet-b0 and MobileNet-V3 Large to conduct experiments.

Vision Transformer (ViT) [12] is a novel neural net-



(a) CIFAR-100 with $\alpha = 0.1$ (b) CIFAR-100 with $\alpha = 0.5$ (c) CIFAR-100 with $\alpha = 100.0$ (d) GLD-23K

Figure 11. Visualization on classification tasks. Figure (a): Visualization of CIFAR-100 distribution on 100 clients with $\alpha = 0.1$. Figure (b): Visualization of CIFAR-100 distribution on 100 clients with $\alpha = 0.5$. Figure (c): Visualization of CIFAR-100 distribution on 100 clients with $\alpha = 100.0$. Figure (d): Visualization of GLD-23K distribution on 233 clients.

work exploiting transformer into Computer Vision and attain excellent results compared to state-of-the-art convolutional networks. We use ViT-B/16 to conduct experiments.

MobileNetV2 [43] is a lightweight Convolutional Neural Network. It is primarily designed to support running neural networks in mobile and edge devices that have severe memory constraints. It implements Inverted Residuals concept where the residual connections are used between bottleneck layers. It also applies Linear Bottlenecks and Depthwise Separable Convolutions concept. In our implementation, we use a network that was pre-trained on the ImageNet dataset.

DeepLabV3+ [38] is a neural network that employs two main principles Atrous Convolutions, Depthwise Separable Convolutions, and Encoder-Decoder architecture (as shown in Figure 12 which attain great performance on Image Segmentation. In our experiments, we exploit MobileNet-V2 and ResNet-101 as two kinds of backbones of DeepLabV3+ to conduct our experiments.

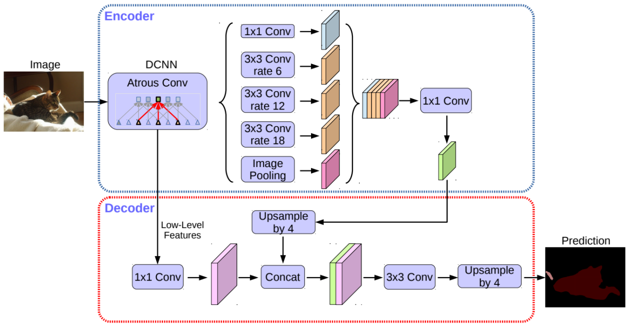


Figure 12. DeeplabV3+ Architecture taken from [38]

U-Net [47] is a Convolutional Neural Network that follows an Encoder-Decoder architecture pattern. As shown in Figure 13, U-Net does not need a backbone network to perform segmentation. However, we have experimented with Resnet-101 and MobileNetV2 pre-trained backbones during experimentation to improve the segmentation output further.

YOLOv5 [29] is an optimized version of YOLOv4 [3]. It outperforms all the previous versions and gets near to EfficientDet Average Precision(AP) with higher frames per second (FPS). Four network models (Yolov5s, Yolov5m, Yolov5l, Yolov5x) with different network depths and widths cater to various applications. Here we use these four models as the pre-trained models.

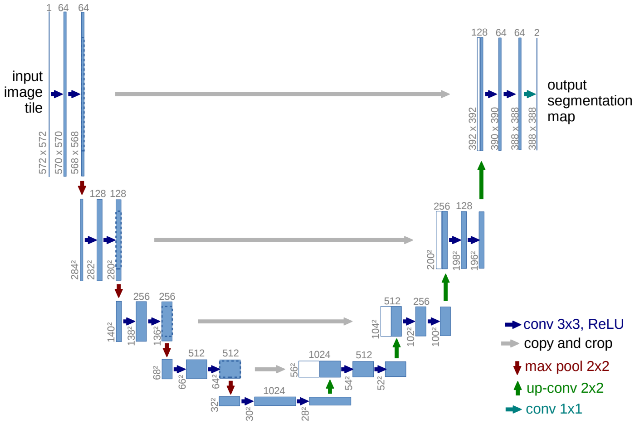


Figure 13. U-Net Architecture taken from [47]

7.1.4 FL Algorithms

Optimizer. Unlike vanilla SGD in FedAvg [5], here we use Momentum SGD to optimize models. For centralized training, our Momentum SGD is the same as the traditional Momentum SGD. However, for FedAvg, the accumulated gradient will be cleared when clients receive the global model from the server. The momentum coefficient is 0.9 for all experiments.

Learning Rate Scheduler. For centralized training on all tasks, we exploit linear learning rate decay each epoch. For FedAvg, we do not use a learning rate scheduler on all tasks. However, we try it on image classification in order to evaluate its effect. Note that the learning rate decay is based on the communication round, not the local epochs as the traditional training process.

FedAvg Training. We use the FedAvg [5] algorithm to conduct federated learning. We list our FedAvg training settings in table 10, in which C means the number of clients

Dataset	clients	Partition	Num of labels
CIFAR-100	100	LDA with α in $\{0.1, 0.5, 100\}$	100
GLD-23K	233	Default	203
augmented PASCAL VOC	4, 8	LDA with αin $\{0.1, 0.5, 100\}$	21
COCO	8	Our partition	80

Table 9. Summary of Dataset partition.

participating in calculation per round, and E means the number of local epochs per round.

Task	C	E
Image Classification	10	1
Image Segmentation	4, 8	1, 2
Object Detection	4, 8	1

Table 10. Summary of FedAvg settings.

Image Transform. For federated learning, it can not be assumed that clients know data distribution of other clients. So we use the average RGB value as $[0, 5, 0.5, 0.5]$ and standard deviation as $[0, 5, 0.5, 0.5]$, instead of them of all images.

Layer-wise Learning Both the models for image segmentation task have encoder-decoder architecture. For encoder layers, we employ ImageNet-1K pre-trained backbones, and hence they are to be just fine-tuned. The decoder layers are trained from scratch. To enforce this set-up, we modify our learning rates layer-wise so that the decoder layers get a learning rate 10 times more than the encoder layers.

7.2. More Experimental Results and Hyper-parameters

For image classification, we list all experiment results and the corresponding hyper-parameters in table 11, 12, 13.

As shown in figure 15, on CIFAR-100 dataset, the higher α makes training more difficult. The fine-tuning can increase the performance a lot.

Figure 4(d) (In main paper) and 14(a)(b) compare the effect of SGD and Momentum SGD. On CIFAR-100 dataset, when $\alpha = 0.5$ and with fine-tuning, Momentum SGD has similar performance with SGD. However, on the GLD-23K dataset, the performance of Momentum SGD is worse than SGD on both fine-tuning and without fine-tuning. One potential reason is that the accumulated gradients make local clients go too far in its local direction, which means that the diversity of the model parameters between different clients is increased by Momentum SGD. Here the local gradient accumulation is much different from the centralized training, in which the accumulated gradient has information of many data samples. But in federated learning, each client only has accumulated gradients with information of its own data samples. Maybe this could be addressed if we can develop some protocols to preserve the gradient information of other clients.

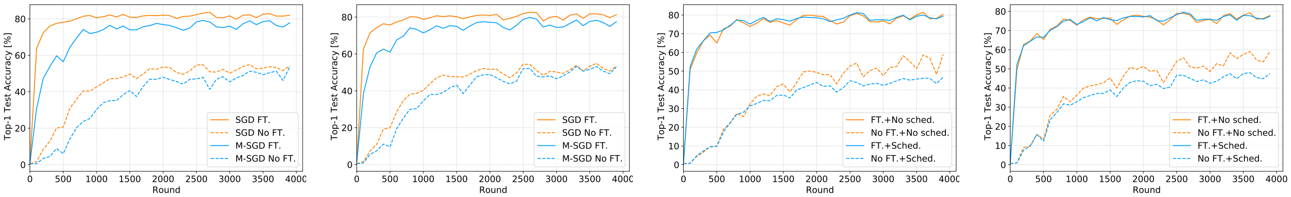
Data	Opt.	Sched.	LR	SFL. Acc	FT. Acc
Cent.	M SGD	Linear	0.003		0.6011
			0.01		0.6058
			0.03		0.5931
			0.1		0.6055
			0.3		0.215
a=0.1	M SGD	No	0.003	0.2239	0.4295
			0.01	0.2306	0.2651
			0.03	0.03025	0.37
		Linear	0.01	0.2384	0.3104
			0.03	0.2469	0.2676
	SGD	Linear	0.01	0.1358	0.1451
			0.03	0.1779	0.3203
a=0.5	M SGD	No	0.003	0.3315	0.5112
			0.01	0.4092	0.5502
			0.03	0.433	0.4841
		Linear	0.01	0.3045	0.5329
			0.03	0.3997	0.53
	SGD	Linear	0.01	0.1847	0.4479
			0.03	0.2861	0.5366
a=100	M SGD	No	0.003	0.3732	0.6158
			0.01	0.4176	0.5749
			0.03	0.4006	0.5527

Table 11. Summary of test accuracy on CIFAR-100 with EfficientNet-b0. In this table, Opt. represents optimizer, Sched. means learning rate scheduler, SFL. means **scratch federated learning**, i.e. not loading a pretrained model, and FT. means **fine-tining**, i.e. loading a pretrained model. In other tables, we also use these abbreviations.

Data	Opt.	Sched.	LR	SFL. Acc	FT. Acc
Cent.	M SGD	Linear	0.003		0.5619
			0.01		0.5785
			0.03		0.5478
			0.1		0.4375
			0.3		0.2678
a=0.1	M SGD	No	0.003	0.2629	0.4276
			0.01	0.2847	0.2959
			0.03	0.2754	0.04848
a=0.5	M SGD	No	0.003	0.3411	0.3714
			0.01	0.426	0.4691
			0.03	0.4418	0.4412
a=100	M SGD	No	0.003	0.3594	0.5203
			0.01	0.4507	0.4046
			0.03	0.4764	0.5193

Table 12. Summary of test accuracy on CIFAR-100 with MobileNet-V3.

Figure 5 (In main paper) and 14(c)(d) compare the effect of linear learning rate decay scheduler. On the GLD-23K dataset, when using fine-tuning, the performance between using the scheduler and not using is similar. However, when not using fine-tuning, using a scheduler cannot improve performance. But on CIFAR-100 dataset, when $\alpha = 0.1$



(a) EfficientNet on GLD-23K (b) MobileNet on GLD-23K (c) EfficientNet on GLD-23K (d) MobileNet on GLD-23K

Figure 14. Experiments on classification with different deep learning models and datasets. Figure (a): Test accuracy of FedAvg with EfficientNet on GLD-23K, using or not using fine tuning, SGD or momentum SGD. Hyper-parameters of this figure can be found in table 13. Figure (b): Test accuracy of FedAvg with MobileNet on GLD-23K, using or not using fine tuning, SGD or momentum SGD. Hyper-parameters of this figure can be found in table 14. Figure (c): Test accuracy of FedAvg with EfficientNet on GLD-23K, using and not using fine tuning, learning rate sceduler or not. Hyper-parameters of this figure can be found in table 13. Figure (d): Test accuracy of FedAvg with MobileNet on GLD-23K, using or not using fine tuning, learning rate sceduler or not. Hyper-parameters of this figure can be found in table 13.

Data	Opt.	Sched.	LR	SFL. Acc	FT. Acc
Cent.	M SGD	Linear	0.03		0.879
			0.1		0.8821
			0.3		0.8826
			0.6		0.88
NonIID	M SGD	No	0.03	0.5319	0.7938
			0.1	0.5901	0.8035
			0.3	0.5615	0.756
		Linear	0.03	0.3706	0.7693
			0.1	0.4681	0.7938
			0.3	0.5355	0.7779
	SGD	Linear	0.01	0.01582	0.5707
			0.3	0.5436	0.8193

Table 13. Summary of test accuracy on GLD-23K with EfficientNet-b0.

Data	Opt.	Sched.	LR	SFL. Acc	FT. Acc
Cent.	M SGD	Linear	0.03		0.8698
			0.1		0.88
			0.3		0.8851
			0.6		0.8729
NonIID	M SGD	No	0.03	0.4992	0.7841
			0.1	0.5911	0.7785
			0.3	0.4788	NaN
		Linear	0.03	0.3267	0.7601
			0.1	0.4758	0.7744
			0.3	0.5294	0.7749
	SGD	Linear	0.01	0.01327	0.6085
			0.3	0.5339	0.8132

Table 14. Summary of test accuracy on GLD-23K with MobileNet-V3.

and not using fine-tuning, we can see the benefit from the scheduler. It would be a non-trivial problem of how to use the learning rate scheduler in FedAvg. And another interesting result of FedAvg is that the test accuracy may drop after some training iterations. The reason for this may be the diverse optimal model parameter of different clients. Maybe some new learning rate scheduler could solve this problem well.

Data	Opt.	Sched.	LR	FT. Acc
Cent.	M SGD	Linear	0.003	0.5967
			0.01	0.7259
			0.03	0.7565
NonIID	M SGD	No	0.003	0.6554
			0.01	0.7386
			0.03	0.7611
			0.1	0.7586
			0.3	0.7489
		Linear	0.03	0.4957
			0.1	0.7458

Table 15. Summary of test accuracy on GLD-23K with ViT. Because the scratch training without pretrained model cannot get converge, we just ignore the results of scratch training here.

7.2.1 Key Takeaways

Fine Tuning. From experiment results, it is obvious that fine-tuning can improve the performance of FedAvg a lot.

Optimizer. The experiment results show that in FedAvg, Momentum SGD cannot guarantee better performance than SGD. Because each client has non-I.I.D. dataset, the direction of gradients have high diversity. The accumulated gradients may be in the wrong direction of the current global model, making the model parameters deviate from the right way. So in federated learning, those optimizers that need an accumulation of gradients, like momentum SGD and RMSprop, may not be suitable in federated learning. Maybe we can develop more new optimizers for FedAvg.

Learning Rate Scheduler. Experiment results show that the performance of learning rate decay is various. FedAvg has much more global epochs than traditional training, which makes the learning rate decay a lot in late training stages. Thus, it is necessary to consider a suitable learning rate scheduler for federated learning.

Image Transform. For federated learning, it can not be assumed that clients know the data distribution of other clients. In such a case, how to decide on a suitable Image Transform is not a trivial problem. Moreover, this is not limited to computer vision but also happens in other areas under federated learning settings. Because it is hard to get the global information of the data, we cannot directly do normalization of them.

Data Augmentation. In recent years, there is a lot of advanced data augmentation techniques in computer vision proposed like AutoAugment [9], and GAN [17], and are exploited in many advanced models like EfficientNet [59]. However, in federated learning, the whole data distribution cannot be attained by the server or any client. In this case, AutoAugment and GAN cannot be directly used in federated learning. This may lead to a performance drop.

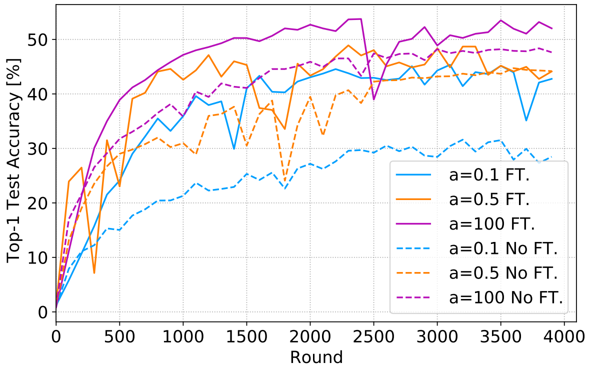


Figure 15. Test accuracy of FedAvg with MobileNet on CIFAR-100 with different Non-IID degree. Hyper-parameters of this figure can be found in table 12

Table 16. various datasets and models used in latest publications from the machine learning community

Conference	**Paper Title**	**dataset**	**partition method**	**model**	**worker/device number**
ICML 2019	Analyzing Federated Learning through an Adversarial Lens [2]	Fashion-MNIST	natural non-IID	3 layer CNNs	10
		UCI Adult Census datase	-	fully connected neural network	10
ICML 2019	Agnostic Federated Learning [46]	UCI Adult Census datase	-	logistic regression	10
		Fashion-MNIST	-	logistic regression	10
		Cornell movie dataset	-	two-layer LSTM mode	10
		Penn TreeBank (PTB) dataset	-	two-layer LSTM mode	10
ICML 2019	Bayesian Nonparametric Federated Learning of Neural Networks [76]	MNIST	Dir(0.5)	1 hidden layer neural networks	10
		CIFAR10	Dir(0.5)	1 hidden layer neural networks	10
ICML 2020	Adaptive Federated Optimization [50]	CIFAR-100	Pachinko Allocation Method	ResNet-18	10
		FEMNIST	natural non-IID	CNN (2xconv)	10
		FEMNIST	natural non-IID	Auto Encoder	10
		Shakespeare	natural non-IID	RNN	10
		StackOverflow	natural non-IID	logistic regression	10
		StackOverflow	natural non-IID	1 RNN LSTM	10
ICML 2020	FetchSGD: Communication-Efficient Federated Learning with Sketching [51]	CIFAR-10/100	1 class / 1 client	ResNet-9	-
		FEMNIST	natural non-IID	ResNet-101	-
		PersonaChat	natural non-IID	GPT2-small	-
ICML 2020	Federated Learning with Only Positive Labels [73]	CIFAR-10	1 class / client	ResNet-8/32	-
		CIFAR-100	1 class / client	ResNet-56	-
		AmazonCAT	1 class / client	Fully Connected Nets	-
		WikiLSHTC	1 class / client	-	-
		Amazon670K	1 class / client	-	-
ICML 2020	SCAFFOLD: Stochastic Controlled Averaging for Federated Learning[31]	EMNIST	1 class / 1 client	Fully connected network	-
ICML 2020	From Local SGD to Local Fixed-Point Methods for Federated Learning [41]	a9a(LIBSVM)	-	Logistic Regression	-
		a9a(LIBSVM)	-	Logistic Regression	-
ICML 2020	Acceleration for Compressed Gradient Descent in Distributed and Federated Optimization [37]	a5a	-	logistic regression	-
		mushrooms	-	logistic regression	-
		a9a	-	logistic regression	-
		w6a LIBSVM	-	logistic regression	-
ICLR 2020	Federated Learning with Matched Averaging [65]	CIFAR-10	-	VGG-9	16
		Shakespheare	sampling 66 clients	1-layer LSTM	66
ICLR 2020	Fair Resource Allocation in Federated Learning [34]	Synthetic dataset use LR	natural non-IID	multinomial logistic regression	10
		Vehicle	natural non-IID	SVM for binary classification	10
		Shakespeare	natural non-IID	RNN	10
		Sent140	natural non-IID	RNN	10
ICLR 2020	On the Convergence of FedAvg on Non-IID Data [36]	MNIST	natural non-IID	logistic regression	10
		Synthetic dataset use LR	natural non-IID	logistic regression	10
ICLR 2020	DBA: Distributed Backdoor Attacks against Federated Learning [70]	Lending Club Loan Data	-	3 FC	10
		MNIST	-	2 conv and 2 fc	10
		CIFAR-10	-	lightweight Resnet-18	10
		Tiny-imagenet	-	Resnet-18	10
MLSys2020	Federated Optimization in Heterogeneous Networks [54]	MNIST	natural non-IID	multinomial logistic regression	10
		FEMNIST	natural non-IID	multinomial logistic regression	10
		Shakespeare	natural non-IID	RNN	10
		Sent140	natural non-IID	RNN	10

*Note: we will update this list once new publications are released.