Improving Generalization in Federated Learning by Seeking Flat Minima

Improving Generalization in Federated Learning by Seeking Flat Minima

Debora Caldarola*[1], Barbara Caputo[1,2], and Marco Ciccone*[1]

[1]Politecnico di Torino, [2]CINI
name.surname@polito.it

Abstract. Models trained in federated settings often suffer from degraded performances and fail at generalizing, especially when facing heterogeneous scenarios. In this work, we investigate such behavior through the lens of geometry of the loss and Hessian eigenspectrum, linking the model's lack of generalization capacity to the sharpness of the solution. Motivated by prior studies connecting the sharpness of the loss surface and the generalization gap, we show that i) training clients locally with Sharpness-Aware Minimization (SAM) or its adaptive version (ASAM) and ii) averaging stochastic weights (SWA) on the server-side can substantially improve generalization in Federated Learning and help bridging the gap with centralized models. By seeking parameters in neighborhoods having uniform low loss, the model converges towards flatter minima and its generalization significantly improves in both homogeneous and heterogeneous scenarios. Empirical results demonstrate the effectiveness of those optimizers across a variety of benchmark vision datasets (e.g. Cifar10/100, Landmarks-User-160k, Idda) and tasks (large scale classification, semantic segmentation, domain generalization).

1 Introduction

Federated Learning (FL) [57] is a machine learning framework enabling the training of a prediction model across distributed clients while maintaining their privacy, never disclosing local data. In recent years it has had a notable resonance in the world of computer vision, with applications ranging from large-scale classification [31] to medical imaging [24] to domain generalization [55] and many others [48,88,23,81]. The learning paradigm is based on communication rounds where a sub-sample of clients trains the global model independently on their local datasets, and the produced updates are later aggregated on the server-side. The heterogeneous distribution of clients' data, which is usually non-i.i.d. and unbalanced, poses a major challenge in realistic federated scenarios, leading to degraded convergence performances [89,30,50]. Locally, the model has only access to a small portion of the data failing to generalize to the rest of the underlying distribution. That contrasts with the standard centralized training, where the

* Equal contribution
Official code: https://github.com/debcaldarola/fedsam

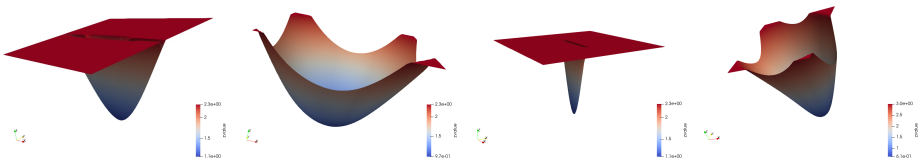


Fig. 1: Cross-entropy loss landscapes of the global model in heterogeneous ($\alpha = 0$) and homogeneous ($\alpha = 1k$) federated scenarios on CIFAR100. When trained with `FedAvg`, the global model converges towards sharp minima. The sharpness-aware optimizer `ASAM` significantly smooths the surfaces.

learner can uniformly sample from the whole distribution. While many promising works in the literature focus on regularizing the local objective to align the global and local solutions, thus reducing the client drift [50,38,1], less attention has been given to the explicit optimization of the loss function for finding better minima. Several works studied the connection between the sharpness of the loss surface and model's generalization [28,39,46,42,71,34,13], and proposed effective solutions based on the minimization of the derived generalization bound [85,20,44] or on averaging the network's parameters along the trajectory of SGD [33].

In this work, we first analyze the heterogeneous federated scenario to highlight the causes behind the poor generalization of the federated algorithms. We hypothesize during local training the model overfits the current distribution, and the resulting average of the updates is strayed apart from local minima. Thus, the global model is not able to generalize to the overall underlying distribution and has a much slower convergence rate, *i.e.* it needs a much larger number of rounds to reach the performance of the homogeneous setting. To speed up training and reduce the performance gap in the case of non-i.i.d. data, we look at improving the generalization ability of the model. Motivated by recent findings relating the geometry of the loss and the generalization gap [39,18,46,36] and by the achievements in the field of Vision Transformers [13], we analyze the loss landscape in the federated scenario and find out that models converge to sharp minima (Fig.1), hence the poor generalization. As a solution, we introduce methods of the current literature that explicitly look for flat minima: i) Sharpness-Aware Minimization (`SAM`) [20] and its adaptive version (`ASAM`) [44] on the client-side and ii) Stochastic Weight Averaging (`SWA`) [33] on the server-side. These modifications, albeit simple, surprisingly lead to significant improvements. Their use is already effective if taken individually, but the best performance is obtained when combined. The resultant models exhibit smoother loss surfaces and improved final performance consistently across several vision tasks. To summarize, our main contributions are:

– We analyze the behavior of models trained in heterogeneous and homogeneous federated scenarios by looking at their convergence points, loss surfaces and Hessian eigenvalues, linking the lack in generalization to sharp minima.

- To encourage convergence towards flatter minima, we introduce `SAM` and `ASAM` in the local client-side training and `SWA` in the aggregation of the updates on the server-side. The resultant models show smoother loss landscapes and lower Hessian eigenvalues, with improved generalization capacities.
- We test our approach on multiple vision tasks, *i.e.* small and large scale classification [31], domain generalization [7] and semantic segmentation [56,12].
- We compare our method with strong data augmentations techniques and state-of-the-art FL algorithms, further validating its effectiveness.

2 Related Works

We describe here the existing approaches closely related to our work. For a comprehensive analysis of the state of the art in FL, we refer to [37,49,86].

2.1 Statistical Heterogeneity in Federated Learning

Federated Learning is a topic in continuous growth and evolution. Aiming at a real-world scenario, the non-i.i.d. and unbalanced distribution of users' data poses a significant challenge. The *statistical heterogeneity* of local datasets leads to unstable and slow convergence, suboptimal performance and poor generalization of the global model [89,30,31]. `FedAvg` [57] defines the standard optimization method and is based on multiple local SGD [66] steps per round. The server-side aggregation is a weighted average of the clients' updates. This simple approach is effective in homogeneous scenarios. Still, it fails to achieve comparable performance against non-i.i.d. data due to local models straying from each other and leading the central model away from the global optimum [38]. To mitigate the effect of the *client drift*, many works enforce regularization in local optimization so that the local model is not led too far apart from the global one [50,38,31,1,48]. Indeed, averaging models/gradients collected from clients having access to a limited subset of tasks may translate into oscillations of the global model and suboptimal performance on the global distribution [54]. Therefore, other lines of research look at improving the aggregation stage using server-side momentum [30] and adaptive optimizers [64], or aggregating task-specific parameters [71,9,10].

In this work, we attempt to explain the behavior of the model in federated scenarios by looking at the loss surface and convergence minima, which is, in our opinion, a fundamental perspective to fully understand the reasons behind the degradation of heterogeneous performance relative to centralized and homogeneous settings. To this end, *we focus on explicitly seeking parameters in uniformly low-loss neighborhoods, without any additional communication cost.* By encouraging local convergence towards flatter minima, we show that the generalization capacity of the global model is consequently improved. Moreover, thanks to the cyclical average of stochastic weights - accumulated along the trajectory of SGD during rounds on the server-side - broader regions of the weight space are explored, and wider optima are reached. Referring to the terminology introduced by [84], we aim at bridging the *participation gap* introduced by unseen clients

distributions. Concurrently, [63] provide a theoretical analysis of SAM in FL, matching the convergence rates of the existing methods. Unlike our work, they do not explicitly focus on the issue of statistical heterogeneity in vision tasks.

2.2 Real-world Vision Scenarios in Federated Learning

Research on FL has mainly focused on algorithmic aspects, often overlooking its application to real scenarios and vision tasks. Here, we perform an analysis of the following real-world settings.

Large-scale Classification. Synthetic federated datasets for classification tasks are usually limited in size and do not offer a faithful representation of reality in the data distribution across clients [31]. [31] addresses such issue by adapting the large-scale Google Landmarks v2 [77] to the federated context, using authorship information. We employ the resulting *Landmarks-User-160k* in our experiments.

Semantic Segmentation. A crucial task for real-world applications [21,61], *e.g.* autonomous driving [70,74], is Semantic Segmentation (SS), which assigns each image pixel to a known category. Most studies of SS in FL focus on medical imaging applications and propose ad hoc techniques to safeguard the patients' privacy [68,52,82,6]. Differently, [58] focuses on object segmentation using prototypical representations. A recently studied application is FL in autonomous driving, motivated by the large amount of privacy-protected data collected by self-driving cars: the authors of [19] propose a new benchmark for analyzing such a scenario, FedDrive. None of those works study the relation between loss landscape and convergence minima of the proposed solution. We apply our approach to the FedDrive benchmark and prove its efficacy in addressing the federated SS task.

Domain Generalization. When it comes to image data collected from devices around the world, it is realistic to assume there may be different *domains* resulting from the several acquisition devices, light, weather conditions, noise, or viewpoints. With the rising development of FL and the privacy concerns, the problem of Domain Generalization (DG) [7] in a federated setting becomes crucial. DG aims to learn a domain-agnostic model capable of satisfying performances on unseen domains, and its application to federated scenarios is still poorly studied. For instance, [55,75] focus on domain shifts deriving from equipment in the medical field, while [19] analyzes the effects of changing landscapes and weather conditions in the setting of autonomous driving. We show that our approach improves generalization to unseen domains both in classification and SS tasks.

2.3 Flat Minima and Generalization

To understand neural networks' generalization, several theoretical and empirical studies analyze its relationship with the geometry of the loss surface [28,39,18,46,36], connecting sharp minima with poor generalization. "*Flatness*" [28] is defined as the dimension of the region connected around the minimum in which the training loss remains low. Interestingly, it has been shown [36] that sharpness-based measures highly correlate with generalization performance. The above studies lead to the introduction of Sharpness-Aware Minimization (`SAM`) [20]

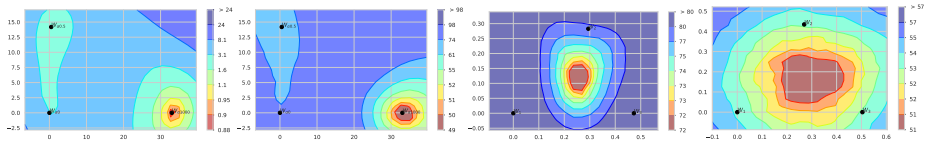
which explicitly seeks flatter minima and smoother loss surfaces through a simultaneous minimization of loss sharpness and value during training. As highlighted by [44], `SAM` is sensitive to parameter re-scaling, weakening the connection between loss sharpness and generalization gap. `ASAM` [44] solves such issue introducing the concept of adaptive sharpness. Encouraged by their effectiveness across a variety of architectures and tasks[13,4], we ask whether `SAM` and `ASAM` can improve generalization in FL as well and find it effective even in the most difficult scenarios. In addition, [22,17] show that local optima found by SGD are connected through a path of near constant loss and that ensambling those points in the weight space leads to high performing networks. Building upon these insights, [33] proposes to average the points traversed by SGD to improve generalization and indeed show the model converges towards wider optima. We modify this approach for FL and use it to cyclically ensemble the models obtained with `FedAvg` on the server side.

3 Behind the Curtain of Heterogeneous FL

3.1 Federated Learning: Overview

The standard federated framework is based on a central server exchanging messages with K distributed clients. Each device k has access to a privacy-protected dataset $\mathcal{D}_k$ made of N_k images belonging to the input space $\mathcal{X}$. The goal is to learn a global model f_θ parametrized by $\theta \in \mathcal{W} \subseteq \mathbb{R}^d$, where $f_\theta : \mathcal{X} \to \mathcal{Y}$ when solving the classification task and $f_\theta : \mathcal{X} \to \mathcal{Y}^{N_p}$ in semantic segmentation, with $\mathcal{Y}$ being the output space and N_p the total number of pixels of each image. We assume the structure of θ to be identical across all devices. The learning procedure spans over T communications rounds, during which a subset of clients $\mathcal{C}$ receives the current model parameters θ^t with $t \in [T]$ and trains it on $\mathcal{D}_k \forall k \in \mathcal{C}$, minimizing a local loss function $\mathcal{L}_k(\theta^t) : \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$. In `FedAvg` [57], the global model is updated as a weighted average of the clients' updates θ_k^t, aiming at solving the global objective $\arg\min_{\theta \in \mathbb{R}^d} \frac{1}{N} \sum_{k \in \mathcal{C}} N_k \mathcal{L}_k(\theta)$, with $N = \sum_{k \in \mathcal{C}} N_k$ being the total training images. In particular, from the generalization perspective - defined $\mathcal{D} \triangleq \bigcup_{k \in [K]} \mathcal{D}_k$ the overall clients' data, $\mathfrak{D}$ its distribution and $\mathcal{L}_\mathcal{D} = {}^1/_{\sum_k N_k} \sum_{k \in [K]} N_k \mathcal{L}_k(\theta)$ the training loss - we aim at learning a model having low population loss $\mathcal{L}_\mathfrak{D}(\theta) \triangleq \mathbb{E}_{(x,y) \sim \mathfrak{D}} \left[\mathbb{E}_\mathcal{D} [\mathcal{L}_k(y, f(x, \theta))] \right]$ [84]. The difference between the population and training losses defines the *generalization gap*, *i.e.* the ability of the model to generalize to unseen data [20].

In realistic scenarios, given two clients i and j, $\mathcal{D}_i$ likely follows a different distribution than $\mathcal{D}_j$, *i.e.* $\mathfrak{D}_i \neq \mathfrak{D}_j$, and the loss $\mathcal{L}_i(\theta)\ \forall i \in [K]$ is typically non-convex in θ. The loss landscape comprehends a multiplicity of local minima leading to models with different generalization performance, *i.e.* significantly different values of $\mathcal{L}_\mathfrak{D}(\theta)$ [20]. Moreover, at each round, the model is likely not to see the entire distribution, further widening the generalization gap [27,26].



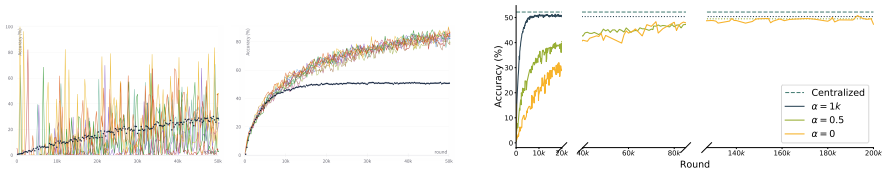
(a) (b) (c) (d)

Fig. 2: **Left**: CNN convergence points in distinct federated scenarios with $\alpha \in [0, 0.5, 1k]$ on CIFAR100. Please refer to Appendix C for implementation details. **(a)** Train loss surface showing the weights obtained at convergence. **(b)** Test error surface of the same models. **Right**: Test error surfaces computed on CIFAR100 using three distinct local models after training. **(c)** When $\alpha = 0$, the local models are not able to generalize to the overall data distribution, being too specialized on the local data. **(d)** When $\alpha = 1k$, the resulting models are connected through a low-loss region.

3.2 Where Heterogeneous FL Fails at Generalizing

In order to fully understand the behavior of a model trained in a heterogeneous federated scenario, we perform a thorough empirical analysis from different perspectives. Our experimental setup replicates that proposed by [31] both as regards the dataset and the network. The CIFAR100 dataset [43], widely used as benchmark in FL, is split between 100 clients, following a Dirichlet distribution with concentration parameter α. To replicate a heterogeneous scenario, we choose $\alpha \in \{0, 0.5\}$, while α is set to 1000 for the homogeneous one. The model is trained over $20k$ rounds. Fore more details, please refer to Appendix C.

Model Behavior in Heterogeneous and Homogeneous Scenarios. In Fig. 3, we compare the training trends in centralized, homogeneous and heterogeneous federated settings: in the latter, not only is the trend much noisier and more unstable, but the performance gap is considerable. Consequently, we question the causes of such behavior. First of all, we wonder if the heterogeneous distribution of the data totally inhibits the model from achieving comparable performances: we find it is only a matter of rounds, *i.e.* with a much larger round budget - 10 times larger in our case - the model reaches convergence (Fig. 3). So it becomes obvious the training is somehow slowed down and there is room for improvement. This hypothesis is further validated by the convergence points of the models trained in different settings (Fig. 2): when $\alpha = 1k$ a low-loss region is reached at the end of training, while the same does not happen with lower values of α, meaning that local minima are still to be found. Moreover, the shift between the train and test surfaces suggests us the model trained in the heterogeneous setting ($\alpha = 0$) is unable to generalize well to unseen data, finding itself in a high-loss region [33]. By analyzing the model behavior, we discover that shifts in client data distribution lead to numerous fluctuations in learning, *i.e.* at each round the model focuses on a subset of the just seen tasks and is unable to generalize to the previously learned ones. This phenomenon is also known as *catastrophic interference* of neural networks [41] and is typical of the world of multitask learning [11,72]. Fig. 3 highlights this by comparing the accuracy of



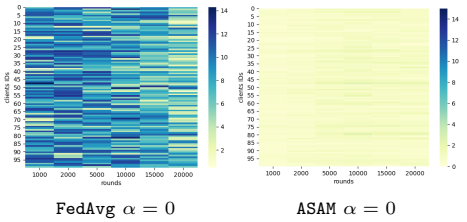
(a) (b) (c)

Fig. 3: CIFAR100 Accuracy trends. **Left**: Global model on local distributions with **(a)** $\alpha = 0$ and **(b)** $1k$ @ $20k$ rounds. Each color represents a local distribution (*i.e.* one class for $\alpha = 0$). **(c)**: $\alpha \in \{0, 0.5, 1k\}$ with necessary rounds to reach convergence.

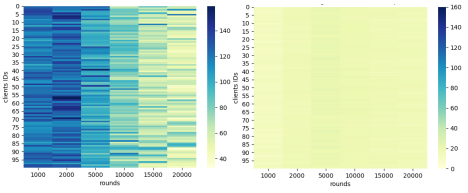
the global model on the clients' data and the test set when $\alpha = 0$ and $\alpha = 1k$. In the first case, at each round the model achieves very high performances on one class but forgets about the others and this behavior is only slightly attenuated as the training continues. In the homogeneous scenario, on the other hand, the model behaves very similarly on each client and convergence is easily reached, giving way to overfitting as the number of rounds increases.

We analyze the clients' local training for further insights from the characteristics of the updated models. By plotting the position of the weights in the loss landscape after training, we find the models easily overfit the local data distribution (Fig. 2): when tested on the test set, the clients' updates are positioned in very high-error regions and as a result the global model moves away from the minimum, meaning the clients specialize too much on their own data and are not able to generalize to the overall underlying distribution. Moreover, Fig. 2 highlights another relevant issue: models trained on homogeneous distributions are connected through a path of low error and can therefore be ensambled to obtain a more meaningful representation [22], but the same does not hold when $\alpha = 0$, where the models are situated in different loss-value regions. Therefore, `FedAvg` averages models that are too far apart to lead to a meaningful result.

Federated Training Converges to Sharp Minima. Many works tried to account for this difficulty arising in federated scenarios by enforcing regularization in local optimization not to lead the local model too far apart from the global one [50,38,31,1,48], or by using momentum on the server-side [30], or learning task-specific parameters keeping distinct models on the server-side [71,9,10]. To the best of our knowledge, this is the first work addressing such behavior by looking at the loss landscape. Inspired by a recent trend in Deep Learning connecting the geometry of the loss and the generalization gap [39,18,46,36,44,33], we investigate the geometry of the loss surface of models trained in non-i.i.d. scenarios with the intention of understanding whether sharp minima may cause the lack of generalization in FL. Following [46], we plot the loss surfaces obtained with models trained in a heterogeneous and in a homogeneous scenario (Fig. 1) showing that both converge to sharp regions, providing a plausible explanation for the highlighted lack of generalization. Additionally, [39] characterizes flatness



FedAvg $\alpha = 0$ ASAM $\alpha = 0$



FedAvg $\alpha = 1k$ ASAM $\alpha = 1k$

Fig. 4: λ_{max}^k for each client k as rounds pass

Table 1: CIFAR100 Hessian eigenvalues.

Algorithm	λ_{max}		λ_{max}/λ_5	
	$\alpha = 0$	$\alpha = 1k$	$\alpha = 0$	$\alpha = 1k$
FedAvg E=1	93.46	106.14	2.00	1.31
FedAvg E=2	110.62	118.35	2.32	1.30
FedSAM	70.29	51.28	**1.79**	1.48
FedASAM	**30.11**	**20.19**	1.80	**1.27**
FedAvg + SWA	97.24	120.02	1.49	1.39
FedSAM + SWA	73.16	54.20	1.56	1.61
FedASAM + SWA	**24.57**	**20.49**	**1.51**	**1.30**

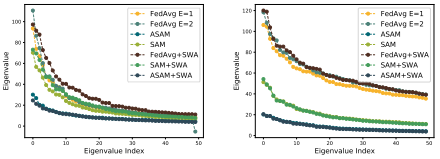


Fig. 5: Hessian eigenspectra of the global model with $\alpha \in \{0, 1k\}$

through the eigenvalues of the Hessian: the dominant eigenvalue λ_{max} evaluates the worst-case landscape curvature, *i.e.* the larger λ_{max} the greater the change in loss in that direction and the steeper the minimum. Hence, we compute the Hessian eigenspectrum (first 50 eigenvalues) using the power iteration mode and analyze it both from the global and local perspectives (Fig. 4,5). Table 1 reports the values of λ_{max} and the ratio λ_{max}/λ_5, commonly used as a proxy for sharpness [35], as the heterogeneity varies. As expected, λ_{max} is large in all settings when using `FedAvg`, implying that such method leads the model towards sharp minima regardless of the data distribution, confirming what was noted in the loss landscapes. As for the client-side analysis, we compute the value of λ_{max}^k using the locally updated parameters θ_k^t on the k-th device's data $\mathcal{D}_k$ $\forall t \in [T]$. Comparing the i.i.d. and non-i.i.d. settings, we note i) the local values of λ_{max} are much lower if $\alpha = 0$, *i.e.* the clients locally reach wide minima (low Hessian maximum eigenvalue, $\lambda_{max}^k \leq 14$) due to the simplicity of the learned task, *i.e.* a narrow subset of the classes, but the average of the distinct updates drives the model towards sharper minima (high Hessian eigenvalues of the global model, $\lambda_{max} \simeq 94$). ii) When $\alpha \in \{0.5, 1k\}$, λ_{max} decreases as the rounds pass, *i.e.* the global model is moving towards regions with lower curvature, while this is not as evident in the heterogeneous setting. Motivated by these results, we believe that introducing an explicit search for flatter minima can help the model generalize.

4 Seeking Flat Minima in Federated Learning

Common first-order optimizers (*e.g.* SGD [66], Adam [40]) are usually non-robust to unseen data distributions [13], since they only aim at minimizing the

Algorithm 1 SAM/ASAM and SWA applied to FedAvg

Require: Initial random model f_θ^0, K clients, T rounds, learning rates γ_1, γ_2, neighborhood size $\rho > 0$, $\eta > 0$, batch size $|\mathcal{B}|$, local epochs E, cycle length c

for each round $t = 0$ **to** $T - 1$ **do**
  if $t = 0.75 * T$ **then** ▷ Apply SWA from 75% of training onwards
    $\theta_{\texttt{SWA}} \leftarrow \theta^t$ ▷ Initialize SWA model
  end if
  if $t \geq 0.75 * T$ **then**
    $\gamma = \gamma(t)$ ▷ Compute LR for the round (Eq. 7 in Appendix)
  end if
  Subsample a set $\mathcal{C}$ of clients
  for each client k in $\mathcal{C}$ in parallel **do** ▷ Iterate over subset $\mathcal{C}$ of clients
    $\theta_{k,0}^{t+1} \leftarrow \theta^t$
    for $e = 0$ **to** $E - 1$ **do**
      for $i = 0$ **to** $N_k/|\mathcal{B}|$ **do**
        Compute gradient $\nabla_\theta \mathcal{L}_\mathcal{B}(\theta_{k,i}^{t+1})$ on batch $\mathcal{B}$ from $\mathcal{D}_k$
        Compute $\hat{\epsilon}(\theta_{k,i}^{t+1}) = \rho \nabla_\theta \mathcal{L}_\mathcal{B}(\theta_{k,i}^{t+1}) / ||\nabla_\theta \mathcal{L}_\mathcal{B}(\theta_{k,i}^{t+1})||_2 =: \hat{\epsilon}(\theta)$ ▷ Solve local maximization (Eq. 3)
        $\theta_{k,i+1}^{t+1} \leftarrow \theta_{k,i}^{t+1} - \gamma\big(\nabla_\theta \mathcal{L}_\mathcal{B}(\theta_{k,i}^{t+1})\big|_{\theta+\hat{\epsilon}(\theta)} \big)$ ▷ Local update with sharpness-aware gradient (Eq. 4)
      end for
    end for
    Send θ_k^{t+1} to the server
  end for
  $\theta^{t+1} \leftarrow \frac{1}{\sum_{k \in \mathcal{C}} N_k} \sum_{k \in \mathcal{C}} N_k \theta_k^{t+1}$ ▷ FedAvg
  if $t \geq 0.75 * T$ **and** $\text{mod}(t, c) = 0$ **then** ▷ End of cycle
    $n_{\text{models}} \leftarrow t/c$
    $\theta_{\texttt{SWA}} \leftarrow \frac{\theta_{\texttt{SWA}} \cdot n_{\text{models}} + \theta^{t+1}}{n_{\text{models}} + 1}$ ▷ Update SWA average (Eq. 8)
  end if
end for

training loss $\mathcal{L}_\mathcal{D}$, without looking at higher-order information correlating with generalization (*e.g.* curvature). The federated scenario exacerbates such behavior due to its inherent statistical heterogeneity, resulting in sharp minima and poor generalization. We hypothesize that encouraging the local model to converge towards flatter neighborhoods may help bridging the generalization gap. To this end, we introduce sharpness-aware minimizers, namely SAM [20] and ASAM [44], on the client-side during local training, and Stochastic Weight Averaging [33] on the server-side after the aggregation, adapting the scenario of [33] to FL. By minimizing the sharpness of the loss surface and the generalization gap, the local models are more robust towards unseen data distributions and, when averaged, build a more solid central model. Defined the *sharpness* of a training loss $\mathcal{L}_\mathcal{D}$ as $\max_{||\epsilon||_p \leq \rho} \mathcal{L}_\mathcal{D}(\theta + \epsilon) - \mathcal{L}_\mathcal{D}(\theta)$, with ρ being the neighborhood size and $p \in [1, \infty)$, SAM aims at minimizing it by solving $\min_{\theta \in \mathbb{R}^d} \max_{||\epsilon||_p \leq \rho} \mathcal{L}_\mathcal{D}(\theta + \epsilon) + \lambda ||\theta||_2^2$. SWA averages weights proposed by SGD, while using a learning rate schedule to explore regions of the weight space corresponding to high performing networks. For a detailed explanation of SAM, ASAM and SWA we refer the reader to Appendix A. Algorithm 1 sums up the details of our approach.

5 Experiments

In this Section, we show the effectiveness of SAM, ASAM and SWA in federated scenarios when addressing tasks of image classification (Sec. 5.1), large-scale

Table 2: FedSAM, FedASAM and SWA on CIFAR100 and CIFAR10

	Algorithm	$\alpha = 0$			$\alpha = 0.5/0.05$			$\alpha = 1000/100$		
		$5cl$	$10cl$	$20cl$	$5cl$	$10cl$	$20cl$	$5cl$	$10cl$	$20cl$
CIFAR100	FedAvg E=1	30.25	36.74	38.59	40.43	41.27	42.17	49.92	50.25	50.66
	FedAvg E=2	24.94	31.81	35.18	38.21	39.59	40.94	48.72	48.64	48.45
	FedSAM	31.04	36.93	38.56	44.73	44.84	46.05	54.01	53.39	53.97
	FedASAM	**36.04**	**39.76**	**40.81**	**45.61**	**46.58**	**47.78**	**54.81**	**54.97**	**54.50**
	FedAvg + SWA	39.34	39.74	39.85	43.90	44.02	42.09	50.98	50.87	50.92
	FedSAM + SWA	39.30	39.51	39.24	47.96	46.76	46.47	**53.90**	53.67	**54.36**
	FedASAM + SWA	**42.01**	**42.64**	**41.62**	**49.17**	**48.72**	**48.27**	53.86	**54.79**	54.10
CIFAR10	FedAvg E=1	65.00	65.54	68.52	69.24	72.50	73.07	84.46	84.50	84.59
	FedAvg E=2	61.49	62.22	66.36	69.23	69.77	73.48	83.93	84.10	84.21
	FedSAM	70.16	71.09	72.90	73.52	74.81	76.04	84.58	84.67	**84.82**
	FedASAM	**73.66**	**74.10**	**76.09**	**75.61**	**76.22**	**76.98**	**84.77**	**84.72**	84.75
	FedAvg + SWA	69.71	69.54	70.19	73.48	72.80	73.81	84.35	84.32	84.47
	FedSAM + SWA	74.97	73.73	73.06	**76.61**	75.84	76.22	84.23	84.37	84.63
	FedASAM + SWA	**76.44**	**75.51**	**76.36**	76.12	**76.16**	**76.86**	**84.88**	**84.80**	**84.79**

classification, SS and DG (Sec. 5.2). Their strength indeed lies in finding flatter minima (Sec. 5.1), which consequently help the model to generalize especially in the heterogeneous scenario. We compare our method with algorithms proper of the FL literature and strong data augmentations (Sec. 5.1), commonly used to improve generalization in DL, further validating the efficacy of our proposal. We refer to App. C for implementation details and App. E for the ablation studies.

5.1 The Effectiveness of the Search for Flat Minima in FL

In Sec. 3.2, we have shown that, given a fixed number of rounds, FL models trained in heterogeneous settings present a considerable performance gap compared to their homogeneous counterparts. Indeed, the gap between the two scenarios can be significant with a difference of up to 20% points (Table 2). We identify the clients' overspecialization on local data as one of the causes of the poor generalization of the global model to the underlying training distribution. We confirm this by showing the model converges to sharp minima, correlated to a poor generalization capacity. In Table 2, we show that explicitly optimizing for flat minima in both the local training and the server-side aggregation does help improving performances, with evident benefits especially in heterogeneous scenarios. We test SAM, ASAM and their combination with SWA on the federated CIFAR10 and CIFAR100 [43,30,31] with several levels of heterogeneity ($\alpha \in \{0, 0.05, 100\}$ for CIFAR10 and $\alpha \in \{0, 0.5, 1k\}$ for CIFAR100) and clients participation ($K \in \{5, 10, 20\}$, *i.e.* 5%, 10%, 20%). As for CIFAR100, we additionally test our approach on the setting proposed by [64], later referred to as CIFAR100-PAM, where the splits reflect the "coarse" and "fine" label structure proper of the dataset. Since both SAM and ASAM perform a step of gradient ascent and one of gradient descent for each iteration, they should be compared with FedAvg with 2 local epochs. However, the results show FedAvg with $E = 2$ suffers even more from statistical heterogeneity, so we will compare our baseline with the better-performing FedAvg with $E = 1$. Our experiments reveal that applying ASAM to FedAvg leads to the best accuracies with a gain of +6% and +8% points respectively on CIFAR100 and CIFAR10 in the most challenging scenario, *i.e.* $\alpha = 0$ and 5 clients per round. This gain

Table 3: Accuracy results on CIFAR100-PAM with ResNet18

Algorithm	Aug	$E=1$						$E=2$					
		10 clients			20 clients			10 clients			20 clients		
		@5k	@10k	w/ SWA	@5k	@10k	w/ SWA	@5k	@10k	w/ SWA	@5k	@10k	w/ SWA
FedAvg		46.60	47.03	52.70	46.51	45.83	50.28	44.58	43.90	51.10	43.31	42.88	47.95
FedSAM		50.71	**53.10**	**55.44**	52.96	53.41	**54.67**	**52.36**	52.04	55.23	51.41	51.35	53.41
FedASAM		49.31	51.10	54.25	47.21	**53.50**	54.29	49.03	49.33	53.01	**53.88**	52.94	54.18
FedAvg	Mixup	43.47	49.25	56.71	50.33	49.89	55.74	**44.76**	46.44	57.15	47.10	47.59	54.40
FedSAM	Mixup	42.83	**51.92**	53.96	49.66	**55.77**	57.70	42.17	51.04	56.54	**53.50**	54.75	**58.88**
FedASAM	Mixup	43.13	51.09	56.31	50.51	52.62	56.89	**44.74**	50.14	**58.31**	49.87	50.87	55.86
FedAvg	Cutout	48.64	48.59	55.40	47.00	46.96	51.70	45.19	45.46	55.40	44.68	44.25	49.39
FedSAM	Cutout	48.28	53.53	57.25	52.06	**54.37**	**56.70**	**49.39**	51.88	**57.32**	**52.16**	52.37	55.45
FedASAM	Cutout	47.52	**52.13**	57.01	50.01	50.66	53.54	48.99	50.09	55.77	48.48	48.77	52.00

Table 4: FedAvg, SAM, ASAM and SWA w/ strong data augmentations (Mixup, Cutout)

	Algorithm	SWA	Aug	$\alpha=0$			$\alpha=0.5/0.05$			$\alpha=1000/100$		
				5cl	10cl	20cl	5cl	10cl	20cl	5cl	10cl	20cl
CIFAR100	FedAvg	✗		29.91	33.67	35.67	35.10	37.80	39.34	55.34	**55.81**	**55.98**
	FedSAM	✗		30.46	34.10	35.89	38.76	40.31	42.03	54.21	54.94	55.24
	FedASAM	✗	Mixup	34.04	36.82	36.97	40.71	42.24	**44.45**	49.75	49.87	49.68
	FedAvg	✓	Mixup	35.56	36.07	36.08	39.21	39.22	38.31	**55.43**	55.37	55.39
	FedSAM	✓	Mixup	35.62	36.25	35.66	42.13	41.95	42.03	52.9	53.14	53.48
	FedASAM	✓	Mixup	**40.08**	**38.74**	**37.47**	**44.53**	**43.97**	44.22	46.97	47.24	46.93
	FedAvg	✗		24.24	31.55	32.44	37.72	38.45	39.48	53.48	53.83	52.90
	FedSAM	✗		23.51	30.92	33.12	40.33	40.31	42.58	**54.27**	**54.75**	**54.76**
	FedASAM	✗	Cutout	30.05	33.62	34.51	41.86	41.84	43.33	51.88	51.78	53.03
	FedAvg	✓	Cutout	33.65	34.40	35.03	40.43	40.12	39.32	53.87	54.09	52.75
	FedSAM	✓	Cutout	34.00	34.08	34.26	43.09	42.81	42.85	53.78	54.28	53.93
	FedASAM	✓	Cutout	**39.30**	**37.46**	**36.27**	**44.76**	**43.48**	**43.95**	50.00	49.65	50.81

is further improved by FedASAM + SWA with a corresponding increase of +12% and +11.5%. The stability introduced by SWA especially helps with lower clients participation, where the trend is noisier. Our ablation studies (Appendix E.3) prove the boost given by SWA is mainly related to the average of the stochastic weights, rather than the cycling learning rate. Table 3 shows the results on CIFAR100-PAM with ResNet18: here SAM and SAM + SWA help more than ASAM.

ASAM and SWA Lead to Flatter Minima in FL. We extend the analysis on the loss landscape and the Hessian eigenspectrum to the models trained with FedSAM, FedASAM and SWA. As expected, both the loss surfaces (Fig. 1) and the Hessian spectra (Fig. 5) indicate us those methods indeed help converging towards flatter minima. The value of λ_{max} goes from 93.5 with FedAvg to 70.3 with FedSAM to 30.1 with FedASAM in the most heterogeneous setting (Table 1). The result is further improved by FedASAM + SWA, obtaining $\lambda_{max} = 24.6$. We notice there is a strict correspondence between the best λ_{max} and the best ratio λ_{max}/λ_5. Even if the maximum eigenvalue resulting with FedAvg + SWA and FedSAM + SWA is higher than the respective one without SWA, the corresponding lower ratio λ_{max}/λ_5 actually tells us the bulk of the spectrum lies in a lower curvature region [20], proving the effectiveness of SWA. Looking at ASAM's behavior from each client's perspective (Fig. 4), flat minima are achieved from the very beginning of the training and that reflects positively on the model's performance.

ASAM and SWA Enable Strong Data Augmentations in FL. Data augmentations usually play a key role in the performance of a neural network and its ability to generalize [87,79,5], but their design often requires domain expertise

Table 5: Comparison of improvements (%) in centralized and heterogeneous federated scenarios ($\alpha = 0$, 5 clients) on CIFAR100, computed w.r.t. the reference at the bottom

Algorithm	Accuracy		Absolute Improvement		Relative Improvement	
	Centr.	$\alpha = 0$	Centr.	$\alpha = 0$	Centr.	$\alpha = 0$
SAM	55.22	31.04	+3.02	+0.79	+5.79	+2.61
ASAM	55.66	36.04	+3.46	+5.79	+6.63	+19.14
SWA	52.72	39.34	+0.52	+9.09	+1.00	+30.05
SAM + SWA	55.75	39.30	+0.55	+9.05	+1.06	+29.92
ASAM + SWA	55.96	42.01	+3.76	**+11.76**	+7.20	**+38.88**
Mixup	58.01	29.91	+5.81	-0.34	+11.13	-1.12
Cutout	55.30	24.24	+3.10	-6.01	+5.94	-19.87
Centralized: **52.20** - FedAvg: **30.25**						

and greater computational capabilities, two elements not necessarily present in a federated context. In Table 3 and 4, we distinctly apply `Mixup` [87] and `Cutout` [16] on CIFAR100-PAM and CIFAR100 (CIFAR10 in Appendix F.2). Surprisingly, both lead to worse performances across all algorithms, so instead of helping the model to generalize, they further slow down training. When combined with our methods, the performance improves in the heterogeneous scenarios w.r.t. the corresponding baseline (`FedAvg` + data augmentation) and `SWA` brings a significant boost, enabling the use of data augmentation techniques in FL.

Heterogeneous FL Benefits Even More from Flat Minima. Given the marked improvement brought by `SAM`, `ASAM` and their combination with `SWA`, one might wonder if this simply reflects the gains achieved in the centralized scenario. In Table 5, we prove the positive gap obtained in the heterogeneous federated scenario is larger than the centralized one, showing those approaches are actually helping the training. We also note that while `Cutout` and `Mixup` improve the performances in the centralized setting, they do not help in FL, where they achieve a final accuracy worse than `FedAvg` (Appendix F.1 for $\alpha \in \{0.5, 1k\}$).

Comparison with FL SOTA. We compare our method with `FedProx` [50], `SCAFFOLD` [38], `FedAvgM` [30], `FedDyn` [1] and `AdaBest` [76], both on their own and combined with `SAM`, `ASAM` and `SWA` (Table 6). `FedProx` adds a proximal term to the local objective and, as expected [47,76], does not bring any notable improvement. `SCAFFOLD` uses control variates to reduce the client drift, exchanging twice the parameters at each round. While performing on par with `FedAvg` in the homogeneous scenario (84.5% on CIFAR10 and 51.9% on CIFAR100), its performance is heavily affected by the data statistical heterogeneity. The same happens for `FedAvgM`. `FedDyn` dynamically aligns global and local stationary points and, as highlighted by [76], is prone to parameters explosion: while it achieves good results on the simpler CIFAR10, it requires heavy gradient clipping and is unable to reach the end of training on CIFAR100. As a solution, `AdaBest` is proposed, exceeding `FedAvg` by a few points. Our results demonstrate the consistent effectiveness of `FedASAM` w.r.t. the SOTA baselines, improving the accuracy by $\approx 6\%$ points on the best SOTA on both datasets. Moreover, by adding `ASAM`, all FL algorithms notably increase their performance. In particular i) we enable `FedAvgM` and `SCAFFOLD` to train in most of the settings with highest heterogeneity, ii) even if limited by the necessary gradient clipping, the results

Table 6: SOTA comparison on CIFAR10 and CIFAR100 (centralized performance)

	Algorithm	w/o SWA				w/ SWA			
		$\alpha = 0$		$\alpha = 0.05/0.5$		$\alpha = 0$		$\alpha = 0.05/0.5$	
		$5cl$	$20cl$	$5cl$	$20cl$	$5cl$	$20cl$	$5cl$	$20cl$
CIFAR10	FedAvg	65.00	68.52	69.24	73.07	69.71	70.19	73.48	73.81
	FedSAM	70.16	72.90	73.52	76.04	74.97	73.06	76.61	76.22
	FedASAM	**73.66**	**76.09**	**75.61**	76.98	76.44	**76.36**	76.12	76.86
	FedAvgM	10.00	10.00	10.00	**78.51**	10.00	10.00	10.00	**84.00**
	FedProx	62.72	68.44	68.38	73.02	70.56	70.08	74.27	73.67
	SCAFFOLD	32.25	15.56	54.46	44.76	11.98	10.00	33.25	24.11
	FedDyn	67.69	73.81	71.36	75.20	77.00	74.00	77.99	75.12
	AdaBest	66.77	72.29	69.84	75.89	**78.94**	76.12	**80.35**	79.35
	FedAvgM + ASAM	77.30	**84.89**	77.06	**84.92**	80.88	85.98	78.29	**86.03**
	FedProx + ASAM	73.74	75.76	75.32	77.03	76.89	75.92	76.65	76.95
	SCAFFOLD + ASAM	**77.78**	77.93	77.59	77.80	75.66	75.30	75.32	75.29
	FedDyn + SAM	77.38	81.00	**79.18**	81.70	**83.81**	**86.07**	**83.18**	85.57
	AdaBest + ASAM	77.48	78.43	78.41	79.72	82.00	80.80	81.87	80.81
CIFAR100	FedAvg	30.25	38.59	40.43	42.17	39.34	39.85	43.90	42.09
	FedSAM	31.04	38.56	44.73	46.05	39.30	39.24	47.96	46.47
	FedASAM	**36.04**	**40.81**	**45.61**	**47.78**	42.01	41.62	**49.17**	48.27
	FedAvgM	1.00	40.64	4.60	**47.88**	1.00	**53.50**	4.60	**53.69**
	FedProx	31.20	38.59	39.53	42.17	39.06	39.68	43.98	41.84
	SCAFFOLD	1.00	1.00	33.26	1.00	1.00	1.00	5.76	1.00
	FedDyn	1.00	1.40	22.03	24.75	1.00	1.40	8.27	35.15
	AdaBest	29.90	39.11	36.93	43.25	**44.48**	44.21	48.20	44.51
	FedAvgM + ASAM	1.00	39.61	4.60	**51.65**	1.00	**51.58**	4.60	**56.19**
	FedProx + ASAM	36.10	40.91	44.81	48.17	43.90	42.06	48.66	48.19
	SCAFFOLD + ASAM	**43.65**	42.61	**46.50**	46.76	40.63	39.07	44.87	44.28
	FedDyn + ASAM	22.16	23.51	38.43	38.60	17.51	19.22	38.60	31.06
	AdaBest + ASAM	39.75	**45.00**	45.25	49.56	**51.75**	47.42	**51.89**	51.47

reached by `FedDyn` on CIFAR100 are almost doubled. Lastly, the best results are obtained with `ASAM + SWA` which stabilizes the noisy learning trends and enables models to converge close to centralized performance with $\alpha = 0$.

5.2 ASAM and SWA in Real World Vision Scenarios

In this Section, we analyze our method in real world scenarios, *i.e.* large scale classification, Semantic Segmentation (SS) for autonomous driving [19] and Domain Generalization (DG) applied to both classification and SS.

Large-scale Classification. We extend our analysis on visual classification tasks to Landmarks-User-160k [31] to validate the effectiveness of `SAM`, `ASAM`, and `SWA` in the presence of real-world challenges such as *Non-Identical Class Distribution* (different distribution of classes per device), and *Imbalanced Client Sizes* (varying number of training data per device). Results confirm the benefits of applying client-side sharpness-aware optimizers, especially in combination with server-side weight averaging with an improvement in final accuracy of up to 7%.

Semantic Segmentation for Autonomous Driving. SS is a fundamental task for applications of autonomous driving. Due to the private nature of the data collected by self-driving cars, it is reasonable to study this task within a federated scenario. We refer to FedDrive [19] - a new benchmark for autonomous driving in FL - for both settings and baselines. The employed datasets are Cityscapes [14] and IDDA [2] with both uniform and heterogeneous settings. To test the generalization capabilities of the model when facing both semantic and appearance shift, the test domain of IDDA either contains pictures taken in the countryside, or in rainy conditions. The model is tested on both previously seen and unseen domains. As shown in Table 8, `ASAM` performs best both on Cityscapes and heterogeneous IDDA. The best performance is obtained combining `ASAM + SWA` with `SiloBN` [3], keeping the BatchNorm [32] statistics local to each client [53] while sharing the learnable parameters across domains.

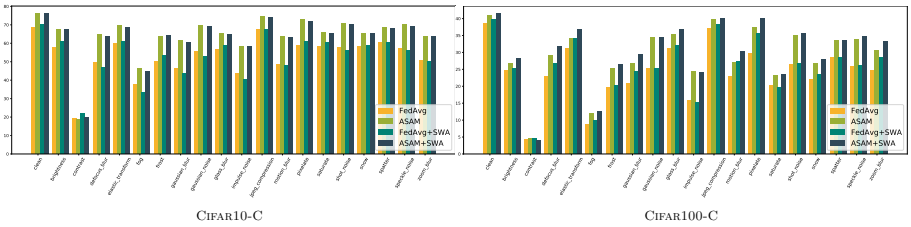
Table 7: Accuracy Results (%) on Landmarks-User-160k

	@5k rounds	w/ SWA 75	w/ SWA 100
FedAvg	61.91	66.05	67.52
FedSAM	63.72	67.11	68.12
FedASAM	64.23	67.17	**68.32**
Centralized		74.03	

Table 8: Federated SS on Cityscapes and IDDA. Results in mIoU (%) @ 1.5k rounds

	Algorithm	Uniform	Country seen	Country unseen	Rainy seen	Rainy unseen	mIoU
IDDA	FedAvg	✓	63.31	48.60	**65.16**	27.38	43.61
	FedSAM	✓	**64.22**	**49.74**	64.81	30.00	44.58
	FedASAM	✓	62.74	48.73	64.74	**31.32**	**45.86**
	FedAvg + SWA	✓	**63.91**	43.28	63.24	27.72	45.64
	FedSAM + SWA	✓	62.26	**46.26**	**63.69**	48.40	45.29
	FedASAM + SWA	✓	60.78	44.23	63.18	**51.76**	**45.69**
	FedAvg	✗	42.06	36.04	39.50	24.59	38.65
	FedSAM	✗	43.28	**37.83**	39.65	29.27	41.22
	FedASAM	✗	**43.67**	36.11	**41.68**	**30.07**	**42.27**
	FedAvg + SWA	✗	37.16	37.48	37.06	42.33	42.48
	FedSAM + SWA	✗	44.26	**40.45**	38.15	45.25	**43.42**
	FedASAM + SWA	✗	**45.23**	39.72	**42.09**	**45.40**	43.02
CityScapes	SiloBN	✗	45.86	32.77	48.09	39.67	45.96
	SiloBN + SAM	✗	**46.88**	33.71	48.22	40.08	49.10
	SiloBN + ASAM	✗	46.57	**35.22**	**48.33**	**40.76**	**49.75**

Domain Generalization. To further show the generalization performance acquired by the model trained with SAM, ASAM and SWA, we test it on the corrupted CIFAR datasets [27]. The test images are altered by 19 corruptions each with 5 levels of severity. Fig. 6 shows the results on the highest severity and once again validate the efficacy of seeking flat minima in FL (complete results in App. D).



CIFAR10-C CIFAR100-C

Fig. 6: Domain generalization in FL. Results with $\alpha = 0$, 20 clients, severity level 5.

6 Conclusions

Heterogeneous Federated Learning suffers from degraded performances and slowdown in training due to the poor generalization of the learned global model. Inspired by recent trends in deep learning connecting the loss landscape and the generalization gap, we analyzed the behavior of the model through the lens of the geometry of the loss surface and linked the lack of generalization to convergence towards sharp minima. As a solution, we introduced Sharpness-Aware Minimization, its adaptive version and Stochastic Weight Averaging in FL for encouraging convergence towards flatter minima. We showed the effectiveness of this approach in several vision tasks and datasets.

Acknowledgments. We thank L. Fantauzzo for her help with the SS experiments. We acknowledge the CINECA HPC infrastructure. Work funded by CINI.

References

1. Acar, D.A.E., Zhao, Y., Navarro, R.M., Mattina, M., Whatmough, P.N., Saligrama, V.: Federated learning based on dynamic regularization. International Conference on Learning Representations (2021) 2, 3, 7, 12
2. Alberti, E., Tavera, A., Masone, C., Caputo, B.: Idda: a large-scale multi-domain dataset for autonomous driving. IEEE Robotics and Automation Letters **5**(4), 5526–5533 (2020) 13, 26
3. Andreux, M., Terrail, J.O.d., Beguier, C., Tramel, E.W.: Siloed federated learning for multi-centric histopathology datasets. In: Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning, pp. 129–139. Springer (2020) 13
4. Bahri, D., Mobahi, H., Tay, Y.: Sharpness-aware minimization improves language model generalization. arXiv preprint arXiv:2110.08529 (2021) 5
5. Bello, I., Fedus, W., Du, X., Cubuk, E.D., Srinivas, A., Lin, T.Y., Shlens, J., Zoph, B.: Revisiting resnets: Improved training and scaling strategies. Advances in Neural Information Processing Systems **34** (2021) 11
6. Bercea, C.I., Wiestler, B., Rueckert, D., Albarqouni, S.: Feddis: Disentangled federated learning for unsupervised brain pathology segmentation. arXiv preprint arXiv:2103.03705 (2021) 4
7. Blanchard, G., Lee, G., Scott, C.: Generalizing from several related classification tasks to a new unlabeled sample. Advances in neural information processing systems **24** (2011) 3, 4
8. Bradbury, J., Frostig, R., Hawkins, P., Johnson, M.J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., Zhang, Q.: JAX: composable transformations of Python+NumPy programs (2018), http://github.com/google/jax 25
9. Briggs, C., Fan, Z., Andras, P.: Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp. 1–9. IEEE (2020) 3, 7
10. Caldarola, D., Mancini, M., Galasso, F., Ciccone, M., Rodolà, E., Caputo, B.: Cluster-driven graph federated learning over multiple domains. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop. pp. 2749–2758 (2021) 3, 7
11. Caruana, R.: Multitask learning. Machine learning **28**(1), 41–75 (1997) 6
12. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence **40**(4), 834–848 (2017) 3
13. Chen, X., Hsieh, C.J., Gong, B.: When vision transformers outperform resnets without pre-training or strong data augmentations. In: International Conference on Learning Representations (2022) 2, 5, 8
14. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3213–3223 (2016) 13, 26
15. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009) 25

16. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017) 12, 23, 28
17. Draxler, F., Veschgini, K., Salmhofer, M., Hamprecht, F.: Essentially no barriers in neural network energy landscape. In: International conference on machine learning. pp. 1309–1318. PMLR (2018) 5
18. Dziugaite, G.K., Roy, D.M.: Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. arXiv preprint arXiv:1703.11008 (2017) 2, 4, 7
19. Fantauzzo, L., Fani', E., Caldarola, D., Tavera, A., Cermelli, F., Ciccone, M., Caputo, B.: Feddrive: Generalizing federated learning to semantic segmentation in autonomous driving. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2022) 4, 13, 26, 29
20. Foret, P., Kleiner, A., Mobahi, H., Neyshabur, B.: Sharpness-aware minimization for efficiently improving generalization. International Conference on Learning Representations (2021) 2, 4, 5, 9, 11, 21
21. Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Garcia-Rodriguez, J.: A review on deep learning techniques applied to semantic segmentation. arXiv preprint arXiv:1704.06857 (2017) 4
22. Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D.P., Wilson, A.G.: Loss surfaces, mode connectivity, and fast ensembling of dnns. Advances in neural information processing systems **31** (2018) 5, 7, 29, 30
23. Gong, X., Sharma, A., Karanam, S., Wu, Z., Chen, T., Doermann, D., Innanje, A.: Ensemble attention distillation for privacy-preserving federated learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 15076–15086 (October 2021) 1
24. Guo, P., Wang, P., Zhou, J., Jiang, S., Patel, V.M.: Multi-institutional collaborations for improving deep learning-based magnetic resonance image reconstruction using federated learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2423–2432 (June 2021) 1
25. Heek, J., Levskaya, A., Oliver, A., Ritter, M., Rondepierre, B., Steiner, A., van Zee, M.: Flax: A neural network library and ecosystem for JAX (2020), http://github.com/google/flax 25
26. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al.: The many faces of robustness: A critical analysis of out-of-distribution generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8340–8349 (2021) 5
27. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. International Conference on Learning Representations (2019) 5, 14, 25
28. Hochreiter, S., Schmidhuber, J.: Flat minima. Neural computation **9**(1), 1–42 (1997) 2, 4
29. Hsieh, K., Phanishayee, A., Mutlu, O., Gibbons, P.: The non-iid data quagmire of decentralized machine learning. In: International Conference on Machine Learning. pp. 4387–4398. PMLR (2020) 25
30. Hsu, T.M.H., Qi, H., Brown, M.: Measuring the effects of non-identical data distribution for federated visual classification. NeurIPS Workshop (2019) 1, 3, 7, 10, 12, 24
31. Hsu, T.M.H., Qi, H., Brown, M.: Federated visual classification with real-world data distribution. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16. pp. 76–92. Springer (2020) 1, 3, 4, 6, 7, 10, 13, 24, 25, 27, 28

32. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. PMLR (2015) 13, 25
33. Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., Wilson, A.G.: Averaging weights leads to wider optima and better generalization. Uncertainty in Artificial Intelligence (UAI) (2018) 2, 5, 6, 7, 9, 21, 22, 28
34. Jastrzebski, S., Kenton, Z., Ballas, N., Fischer, A., Bengio, Y., Storkey, A.: On the relation between the sharpest directions of dnn loss and the sgd step length. International Conference on Learning Representations (2019) 2
35. Jastrzebski, S., Szymczak, M., Fort, S., Arpit, D., Tabor, J., Cho, K., Geras, K.: The break-even point on optimization trajectories of deep neural networks. arXiv preprint arXiv:2002.09572 (2020) 8
36. Jiang, Y., Neyshabur, B., Mobahi, H., Krishnan, D., Bengio, S.: Fantastic generalization measures and where to find them. arXiv preprint arXiv:1912.02178 (2019) 2, 4, 7
37. Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al.: Advances and open problems in federated learning. arXiv preprint arXiv:1912.04977 (2019) 3
38. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T.: Scaffold: Stochastic controlled averaging for federated learning. In: International Conference on Machine Learning. pp. 5132–5143. PMLR (2020) 2, 3, 7, 12
39. Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M., Tang, P.T.P.: On large-batch training for deep learning: Generalization gap and sharp minima. International Conference on Learning Representations (2017) 2, 4, 7
40. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. ICLR (2015) 8
41. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences **114**(13), 3521–3526 (2017) 6
42. Kleinberg, B., Li, Y., Yuan, Y.: An alternative view: When does sgd escape local minima? In: International Conference on Machine Learning. pp. 2698–2707. PMLR (2018) 2
43. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009) 6, 10
44. Kwon, J., Kim, J., Park, H., Choi, I.K.: Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. International Conference on Machine Learning (2021) 2, 5, 7, 9, 21, 22
45. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998) 24
46. Li, H., Xu, Z., Taylor, G., Studer, C., Goldstein, T.: Visualizing the loss landscape of neural nets. In: Neural Information Processing Systems (2018) 2, 4, 7, 30
47. Li, Q., Diao, Y., Chen, Q., He, B.: Federated learning on non-iid data silos: An experimental study. arXiv preprint arXiv:2102.02079 (2021) 12
48. Li, Q., He, B., Song, D.: Model-contrastive federated learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10713–10722 (2021) 1, 3, 7
49. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: Challenges, methods, and future directions. IEEE Signal Processing Magazine **37**(3), 50–60 (2020) 3

50. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. Proceedings of Machine Learning and Systems **2**, 429–450 (2020) 1, 2, 3, 7, 12
51. Li, W., McCallum, A.: Pachinko allocation: Dag-structured mixture models of topic correlations. In: Proceedings of the 23rd international conference on Machine learning. pp. 577–584 (2006) 25
52. Li, W., Milletarì, F., Xu, D., Rieke, N., Hancox, J., Zhu, W., Baust, M., Cheng, Y., Ourselin, S., Cardoso, M.J., et al.: Privacy-preserving federated brain tumour segmentation. In: International workshop on machine learning in medical imaging. pp. 133–141. Springer (2019) 4
53. Li, Y., Wang, N., Shi, J., Liu, J., Hou, X.: Revisiting batch normalization for practical domain adaptation. ICLR Workshop (2017) 13
54. Lin, T., Kong, L., Stich, S.U., Jaggi, M.: Ensemble distillation for robust model fusion in federated learning. arXiv preprint arXiv:2006.07242 (2020) 3
55. Liu, Q., Chen, C., Qin, J., Dou, Q., Heng, P.A.: Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1013–1023 (2021) 1, 4
56. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3431–3440 (2015) 3
57. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017) 1, 3, 5
58. Michieli, U., Ozay, M.: Prototype guided federated learning of visual feature representations. arXiv preprint arXiv:2105.08982 (2021) 4
59. Mirzadeh, S.I., Farajtabar, M., Gorur, D., Pascanu, R., Ghasemzadeh, H.: Linear mode connectivity in multitask and continual learning. NeurIPS (2018) 29
60. Noah, G., Zhewei, Y., Amir, G., Michael, M., Joseph, G.: pytorch-hessian-eigenthings: efficient pytorch hessian eigendecomposition (Oct 2018), https://github.com/noahgolmant/pytorch-hessian-eigenthings 30
61. Ouahabi, A., Taleb-Ahmed, A.: Deep learning for real-time semantic segmentation: Application in ultrasound imaging. Pattern Recognition Letters **144**, 27–34 (2021) 4
62. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019) 24
63. Qu, Z., Li, X., Duan, R., Liu, Y., Tang, B., Lu, Z.: Generalized federated learning via sharpness aware minimization. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., Sabato, S. (eds.) Proceedings of the 39th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 162, pp. 18250–18280. PMLR (17–23 Jul 2022) 4
64. Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., McMahan, H.B.: Adaptive federated optimization. International Conference on Learning Representations (2021) 3, 10, 25, 27, 28, 30
65. Ro, J.H., Suresh, A.T., Wu, K.: Fedjax: Federated learning simulation with jax. arXiv preprint arXiv:2108.02117 (2021) 25

66. Ruder, S.: An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747 (2016) 3, 8
67. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4510–4520 (2018) 25
68. Sheller, M.J., Reina, G.A., Edwards, B., Martin, J., Bakas, S.: Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation. In: International MICCAI Brainlesion Workshop. pp. 92–104. Springer (2018) 4
69. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 761–769 (2016) 29
70. Siam, M., Gamal, M., Abdel-Razek, M., Yogamani, S., Jagersand, M., Zhang, H.: A comparative study of real-time semantic segmentation for autonomous driving. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops. pp. 587–597 (2018) 4
71. Smith, S.L., Le, Q.V.: A bayesian perspective on generalization and stochastic gradient descent. International Conference on Learning Representations (2018) 2, 3, 7
72. Smith, V., Chiang, C.K., Sanjabi, M., Talwalkar, A.S.: Federated multi-task learning. Advances in neural information processing systems **30** (2017) 6
73. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015) 26
74. Tavera, A., Cermelli, F., Masone, C., Caputo, B.: Pixel-by-pixel cross-domain alignment for few-shot semantic segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1626–1635 (2022) 4
75. Tian, C.X., Li, H., Wang, Y., Wang, S.: Privacy-preserving constrained domain generalization for medical image classification. arXiv preprint arXiv:2105.08511 (2021) 4
76. Varno, F., Saghayi, M., Rafiee, L., Gupta, S., Matwin, S., Havaei, M.: Minimizing client drift in federated learning via adaptive bias estimation. arXiv preprint arXiv:2204.13170 (2022) 12
77. Weyand, T., Araujo, A., Cao, B., Sim, J.: Google landmarks dataset v2-a large-scale benchmark for instance-level recognition and retrieval. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2575–2584 (2020) 4, 25
78. Wu, Y., He, K.: Group normalization. In: Proceedings of the European conference on computer vision (ECCV). pp. 3–19 (2018) 25
79. Xie, C., Tan, M., Gong, B., Wang, J., Yuille, A.L., Le, Q.V.: Adversarial examples improve image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 819–828 (2020) 11
80. Xu, P., He, B., De Sa, C., Mitliagkas, I., Re, C.: Accelerated stochastic power iteration. In: International Conference on Artificial Intelligence and Statistics. pp. 58–67. PMLR (2018) 30
81. Yao, C.H., Gong, B., Qi, H., Cui, Y., Zhu, Y., Yang, M.H.: Federated multi-target domain adaptation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1424–1433 (2022) 1
82. Yi, L., Zhang, J., Zhang, R., Shi, J., Wang, G., Liu, X.: Su-net: an efficient encoder-decoder model of federated learning for brain tumor segmentation. In: International Conference on Artificial Neural Networks. pp. 761–773. Springer (2020) 4

83. Yu, C., Gao, C., Wang, J., Yu, G., Shen, C., Sang, N.: Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation. International Journal of Computer Vision **129**(11), 3051–3068 (2021) 26, 29
84. Yuan, H., Morningstar, W., Ning, L., Singhal, K.: What do we mean by generalization in federated learning? NeurIPS Workshop (2021) 3, 5
85. Yue, X., Nouiehed, M., Kontar, R.A.: Salr: Sharpness-aware learning rates for improved generalization. arXiv preprint arXiv:2011.05348 (2020) 2
86. Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., Gao, Y.: A survey on federated learning. Knowledge-Based Systems **216**, 106775 (2021) 3
87. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. International Conference on Learning Representations (2018) 11, 12, 23, 28
88. Zhang, L., Luo, Y., Bai, Y., Du, B., Duan, L.Y.: Federated learning for non-iid data via unified feature learning and optimization objective alignment. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 4420–4428 (October 2021) 1
89. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-iid data. arXiv preprint arXiv:1806.00582 (2018) 1, 3

Appendix

A Background

In this section, we briefly review the details of Sharpness-Aware Minimization (SAM) [20], its adaptive version (ASAM) [44] and Stochastic Weight Averaging (SWA) [33].

A.1 SAM and ASAM: Overview

SAM aims at finding the solution θ surrounded by a neighborhood having uniform low training loss $\mathcal{L}_{\mathcal{D}}(\theta)$, *i.e.* located in a flat minimum. The *sharpness* of a training loss function is defined as:

$$\max_{||\epsilon||_p \leq \rho} \mathcal{L}_{\mathcal{D}}(\theta + \epsilon) - \mathcal{L}_{\mathcal{D}}(\theta) \tag{1}$$

where ρ is an hyper-parameter defining the neighborhood size and $p \in [1, \infty)$. SAM aims at minimizing the sharpness of the loss solving the following minmax objective:

$$\min_{\theta \in \mathbb{R}^d} \max_{||\epsilon||_p \leq \rho} \mathcal{L}_{\mathcal{D}}(\theta + \epsilon) + \lambda ||\theta||_2^2 \tag{2}$$

where λ is a hyper-parameter weighing the importance of the regularization term. In [20], it is shown that $p = 2$ is typically the optimal choice, hence, without loss of generality, we use the ℓ_2-norm in the maximization over ϵ and omit the regularization term for simplicity. In order to obtain the exact solution of the inner maximization problem $\epsilon^* \triangleq \arg\max_{||\epsilon||_2 \leq \rho} \mathcal{L}(\theta + \epsilon)$, the authors propose to employ a first-order approximation of $\mathcal{L}(\theta + \epsilon)$ around 0:

$$\epsilon^* \approx \arg \max_{||\epsilon||_2 \leq \rho} \mathcal{L}_{\mathcal{D}}(\theta) + \epsilon^T \nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta) = \rho \frac{\nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta)}{||\nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta)||_2} =: \hat{\epsilon}(\theta) \tag{3}$$

Under this computationally efficient approximation, $\hat{\epsilon}(\theta)$ is nothing more than a scaled gradient of the current parameters θ. The sharpness-aware gradient is then defined as $\nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta)|_{\theta + \hat{\epsilon}(\theta)}$ and used to update the model as

$$\theta_{t+1} \leftarrow \theta_t - \gamma \nabla_{\theta_t} \mathcal{L}_{\mathcal{D}}(\theta_t)|_{\theta_t + \hat{\epsilon}_t}, \tag{4}$$

where γ is an appropriate learning rate and $\hat{\epsilon}_t = \hat{\epsilon}(\theta_t)$. This two-steps procedure is iteratively applied to solve Eq. 2. Intuitively, SAM performs a first step of gradient ascent to estimate the point $(\theta_t + \hat{\epsilon}_t)$ at which the loss is approximately maximized and then applies gradient descent at θ_t using the just computed gradient.

ASAM In [44], the authors point out that sharpness defined in a rigid region with a fixed radius ρ (Eq. 1) is sensitive to parameter re-scaling, negatively affecting the connection between sharpness and generalization gap. If A is a scaling operator acting on the parameters space without changing the loss function, two neural networks with weights θ and $A\theta$ can have different values of sharpness while maintaining the same generalization gap, *i.e.* the sharpness is *scale-dependent*. As a solution, they introduce the concept of *adaptive* sharpness, defined as

$$\max_{||T_\theta^{-1}\epsilon||_p \leq \rho} \mathcal{L}_\mathcal{D}(\theta + \epsilon) - \mathcal{L}_\mathcal{D}(\theta) \tag{5}$$

where T_θ^{-1} is the normalization operator of θ such that $T_{A\theta}^{-1} A = T_\theta^{-1}$. Eq. 2 can be rewritten to define the Adaptive Sharpness-Aware Minimization (ASAM) problem as follows:

$$\min_{\theta \in \mathbb{R}^d} \max_{||T_\theta^{-1}\epsilon||_p \leq \rho} \mathcal{L}_\mathcal{D}(\theta + \epsilon) + \lambda ||\theta||_2^2 \tag{6}$$

For improving stability, T_θ is substituted by $T_\theta + \eta I_w$, where $\eta > 0$ is a hyper-parameter controlling the trade-off between stability and adaptivity, while w is the number of weight parameters of the model.

A.2 Stochastic Weight Averaging: Overview

SWA averages weights proposed by SGD, while using a learning rate schedule to explore regions of the weight space corresponding to high performing networks. At each step i of a cycle of length c, the learning rate is decreased from γ_1 to γ_2:

$$\gamma(i) = \big(1 - t(i)\big)\gamma_1 + t(i)\gamma_2, \quad t(i) = \frac{1}{c}\big(mod(i-1, c) + 1\big) \tag{7}$$

If $c = 1$ the learning rate is constant (γ_1), otherwise for $c > 1$ the learning schedule is cyclical. Starting from a pre-trained model $f_{\hat{\theta}}$, SWA captures all the updates θ at the end of each cycle and averages them as:

$$\theta_{\text{SWA}} \leftarrow \frac{\theta_{\text{SWA}} \cdot n_{\text{models}} + \theta}{n_{\text{models}} + 1} \tag{8}$$

obtaining the final model $f_{\theta_{\text{SWA}}}$, where n_{models} keeps track of the number of completed cycles.

In our method, SWA is applied on the server-side to make the learning process more robust. Adapting the scenario of [33] to FL, from 75% of the training onwards, the server keeps two models, f_θ and $f_{\theta_{\texttt{SWA}}}$ (f and $f_{\texttt{SWA}}$ to simplify the notation). f follows the standard FedAvg paradigm, while $f_{\texttt{SWA}}$ is updated every c rounds (Eq. 8). At each round, the cycling learning rate is computed (Eq. 7) and used for the clients' local training.

A.3 Mixup and Cutout: Overview

Mixup and Cutout are recent methods for data augmentation, aiming to improve the learned models' generalization. We apply one of the two in the client-side training.

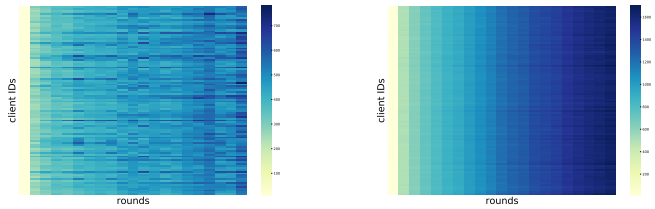


Fig. 7: CIFAR100. L2-norm of global classifier output features as rounds pass, after receiving as input each client's local data. **(a)** with $\alpha = 0$, the model tends to focus on a different client's distribution, *i.e.* on a single class, at each round. **(b)** when $\alpha = 1000$, the model gives the same attention to each distribution.

mixup [87] trains the neural network on convex combinations of images and their labels, exploiting the prior knowledge that linear interpolation of features leads to linear interpolations of their corresponding targets. Given two input images (x_i, x_j) and their corresponding one-hot label encodings (y_i, y_j) drawn from the k-th client's training data $\mathcal{D}_k$, virtual training examples are constructed as follows:

$$\begin{aligned} \bar{x} &= \lambda x_i + (1-\lambda)x_j \\ \bar{y} &= \lambda y_i + (1-\lambda)y_j \end{aligned} \tag{9}$$

with $\lambda \sim \text{Beta}(\alpha, \alpha)$ for $\alpha \in (0, \infty)$.

Cutout [16] regularizes learning by randomly masking out square regions of the input during training. At the implementation level, this corresponds to applying a fixed-size zero-mask to a random location of the image.

B Training in Heterogeneous Scenarios - Additional Material

In this section, we provide further analysis of the model's behavior in heterogeneous and homogeneous federated scenarios. As explained in Sec. 3.2, the model trained under a condition of statistical heterogeneity is subject to oscillations and loss in performance and generalization. Fluctuations in model predictions can also be noted by looking at its output features, defined as $f_\theta(x)\ \forall x \in \mathcal{X}$. Fig. 7 shows the L2-norm of the output features computed using the current global model $f_\theta^t\ \forall t \in [T]$, given as input the local clients' data $\mathcal{D}_k\ \forall k \in [K]$, where a higher norm value corresponds to greater attention paid to that class by the network. The uniformity of the features obtained in the homogeneous setting contrasts with the chaotic distribution of the ones resulting when $\alpha = 0$, which significantly vary over time without following a constant trend.

Table 9: Datasets statistics

Dataset	Task	Train clients	Size imbalance	Train samples	Test samples
CIFAR10	Classification	100	✗	50,000	10,000
CIFAR100	Classification	100	✗	50,000	10,000
CIFAR100-PAM	Classification	500	✗	50,000	10,000
CIFAR10-C	DG	-	-	-	10,000
CIFAR100-C	DG	-	-	-	10,000
LANDMARKS-USER-160K	Classification	1,262	✓	164,172	19,526
CITYSCAPES (uniform)	SS	146	✓	2,975	500
CITYSCAPES (heterogeneous)	SS	144	✓		
IDDA (country)	SS+DG	90	✗	4,320	1,920
IDDA (rainy)	SS+DG	69	✗	3,312	2,928

C Experiments Details

Here we provide a detailed description of the datasets and models used in the paper, together with information regarding the chosen hyper-parameters and their fine-tuning intervals. All results presented in both the main text and the Appendix are averaged over the last 100 rounds for increased robustness and reliability. Unless otherwise specified, the framework is PyTorch [62] and experiments were run on one NVIDIA GeForce GTX 1070.

C.1 Datasets and Models

Table 9 summarizes the tasks and the statistics of the number of clients and examples for each dataset.

CIFAR10 and CIFAR100 We replicate the federated version of the CIFAR datasets proposed by [30]. Each dataset is split among 100 clients, receiving 500 images each according to the latent Dirichlet distribution (LDA) applied to the labels. The client's examples are selected following a multinomial distribution drawn from a symmetric Dirichlet distribution with parameter α. The higher the value of α the larger the number of classes locally seen , *i.e.* the more similar and homogeneous the clients' distributions are. We test $\alpha \in \{0, 0.05, 100\}$ on CIFAR10 and $\alpha \in \{0, 0.5, 1000\}$ on CIFAR100. The task is image classification on 10 (CIFAR10) and 100 (CIFAR100) classes.

Model: We train a Convolutional Neural Network (CNN) similar to LeNet5 [45] on both datasets, following the setting of [31]. The network has two 64-channels convolutional layers with kernel of size 5×5, each followed by a 2×2 max-pooling layer, ended by two fully connected layers with 384 and 192 channels respectively and a linear classifier.

Data pre-processing: The 32×32 input images are pre-processed following the standard pipeline: the training images are randomly cropped applying padding 4 with final size 32×32, randomly horizontally flipped with probability 0.5 and finally the pixel values are normalized with the dataset's mean and standard deviation; normalization is applied to test images as well.

CIFAR100-PAM We further extend our experiments to a more complex version of CIFAR100, *i.e.* CIFAR100-PAM proposed by [64], reflecting the "coarse" and "fine" label structure of the dataset for a more realistic partition. The dataset is split among 500 clients - with 100 images each - following the Pachinko Allocation Method (PAM) [51], on the result of which LDA is applied.

Model: We train a modified ResNet18, replacing Batch Normalization [32] layers with group normalization (GN) ones [78], as suggested by [29]. We use two groups for each GN layer. Experiments have been run using `FedJAX` [65] on a cluster with NVIDIA V100 GPUs.

Data pre-processing: CIFAR100-PAM images are pre-processed as the CIFAR LDA versions described above.

CIFAR10-C and CIFAR100-C are the corrupted versions of the CIFAR datasets. They are part of the benchmark proposed by [27], used for testing the image classifiers' robustness. The $10k$ images-test set is modified according to a given *corruption* and a corresponding level of *severity*. There are 19 possible corruptions (brightness, contrast, elastic blur, elastic transform, fog, frost, Gaussian blur, Gaussian noise, glass blur, impulse noise, JPEG compression, motion blur, pixelate, saturate, short noise, snow, spatter, speckle noise, zoom blur), while the severity ranges from 1 (low) to 5 (high).

Model: The same model described for CIFAR10 and CIFAR100 is used here. To test the generalization ability of our method, we test the model trained with CIFAR10/100 on the corresponding corrupted dataset.

Landmarks-User-160k Introduced by [31], the Landmarks-User-160k dataset comprises 164,172 training images belonging to 2,028 landmarks. The dataset is created according to the authorship information from the large-scale dataset Google Landmarks v2 (GLv2) [77]. Each author owns at least 30 pictures depicting 5 or more landmarks, while each location is depicted by at least 30 images and was visited by no less than 10 users. The authors in the test set do not overlap with the ones appearing in the training split.

Model: We follow a setting similar to the one proposed by [31] and use a MobileNetV2 [67] network pre-trained on ImageNet [15] with with GroupNorm layers in place of BatchNorm. Since no details on the model are available, we set the network feature multiplier $\alpha = 1$ and use 8 groups for the GN layers. We did not apply a bottleneck layer before the classifier as specified in [31]. To reduce training time, we use `Flax` [25] for both pre-training and centralized baselines, and `FedJAX` [65] for the implementation of the federated algorithms. Both libraries are based on `JAX` [8] and allow for efficient data parallelization. Implementation of the MobileNetV2 backbone used for all the experiments is

available here[1]. All large-scale classification experiments have been performed using an NVIDIA DGX A100 40GB.

The model trained on ImageNet reaches $\approx 68\%$ top-1 accuracy on the validation set. In our experience, GroupNorm tends to perform slightly worse than BatchNorm when trained on ImageNet. However, since we did not extensively tune the hyper-parameters, getting better final performance is possible. For the ImageNet training, we used 8 GPUs with a total batch size of 2048 images.

Data pre-processing: We applied the same data augmentation for training the model on ImageNet and fine-tuning on GLv2: we crop and resize the input images to 224×224 with random scale and aspect ratio as described in [73]. The data augmentation pipeline used for the experiments can be found here[2]. We also adapted the GLv2 TensorFlow Federated data pipeline[3] to be compatible with `FedJAX`.

Cityscapes [14] is a popular dataset for Semantic Segmentation and contains 2,975 real photos taken in the streets of 50 different cities under good weather conditions. Annotations are provided for 19 semantic classes. We refer to the federated splits proposed in the FedDrive benchmark [19]. The *uniform* version of the dataset randomly assigns each image to one of the 146 users. In order to account for the distribution heterogeneity appearing in real-world scenarios, an ulterior version is proposed, referred to as *heterogeneous*: every client only accesses images from one of the 18 training cities. In both cases, the test set contains pictures of unseen cities.

Model: As proposed by the authors of FedDrive, we employ the lightweight network BiSeNetv2 [83] for training, accounting for possible lower computational capabilities of the edge devices.

Data pre-processing: The images are randomly scaled in the range (0.5, 1.5) and cropped to a 512×1024 shape.

IDDA [2] is a synthetic dataset for semantic segmentation, specific for the field of autonomous driving. In addition to the annotations for 16 semantic classes, the driving conditions are further characterized by three axes: a city among the 7 available, ranging from Urban to Rural environments; one of 5 viewpoints, simulating different vehicles; an atmospheric condition among 3 possible choices (Noon, Sunset, Rainy), for a total of 105 *domains*. As done for Cityscapes, we refer to FedDrive [19] for the federated splits. In the *uniform* distribution of IDDA, each client has access to 48 images randomly drawn from the whole dataset. The

[1] https://github.com/rwightman/efficientnet-jax/tree/a65811fbf63cb90b9ad0724792040ce93b749303

[2] https://github.com/google/flax/blob/571018d16b42ce0a0387515e96ba07130cbf79b9/examples/imagenet/input_pipeline.py

[3] https://www.tensorflow.org/federated/api_docs/python/tff/simulation/datasets/gldv2/load_data

Table 10: Best performing training parameters

Dataset	Client learning rate	Batch size	Weight decay	Epochs	Client momentum	Rounds	Clients per round
CIFAR10	0.01	64	$4 \cdot 10^{-4}$	1	0	$10k$	$\{5, 10, 20\}$
CIFAR100	0.01	64	$4 \cdot 10^{-4}$	1	0	$20k$	$\{5, 10, 20\}$
CIFAR100-PAM	0.01	20	$4 \cdot 10^{-4}$	1-2	0.9	$10k$	$\{10, 20\}$
LANDMARKS-USER-160K	0.1	64	$4 \cdot 10^{-5}$	5	0	$5k$	10
CITYSCAPES (unif.)	0.05	8	$5 \cdot 10^{-4}$	2	0.9	$1.5k$	5
CITYSCAPES (het.)	0.05	8	$5 \cdot 10^{-4}$	2	0.9	$1.5k$	5
IDDA (country)	0.1	8	0	2	0.9	$1.5k$	5
IDDA (rainy)	0.1	8	0	2	0.9	$1.5k$	5

heterogeneous version is built so that every user only sees a single domain. Two distinct testing scenarios are proposed to assess the generalization abilities of the learned model: one with images belonging to domains likely already seen at training time ("seen" in Table 8 of the main text) and another one containing a never-seen one ("unseen"). The unseen domain either contains images taken in the countryside ("country") to analyze the *semantic* shift or in rainy conditions ("rainy") for studying the shift in *appearance*.

Model: As done for Cityscapes, BiSeNetv2 is the model of choice.

Data pre-processing: The images are randomly scaled in the range (0.5, 2.0) and cropped to a 512×928 shape.

C.2 Hyper-parameters Tuning

We consider a different hyper-parameters setup for each dataset. The final choices of training hyper-parameters are summarized in Table 10. Table 11 and 12 respectively show the values used for `SAM`/`ASAM` and `SWA`.

CIFAR10 and CIFAR100 For both datasets, the training hyper-parameters follow the choice of [31]. The client learning rate is tuned between the values $\{0.01, 0.1\}$ and set to 0.01, the batch size is 64, $E \in \{1, 2\}$ is tested for the number of local epochs and the former is chosen. As for the weight decay the value $4 \cdot 10^{-4}$ leads to better performances than 0. The local optimizer is SGD with no momentum. No learning rate scheduler is used for simplicity. We optimize the cross-entropy loss. As for the server-side, we compare the behavior of different optimizers (*i.e.* SGD, Adam, AdaGrad) with learning rates in $\{0.001, 0.01, 0.1, 1\}$ (results in Appendix E.1), following the setup of [64], and find out that `FedAvg`, *i.e.* SGD with learning rate 1, is the best choice. When testing `FedAvgM`, the server-side momentum $\beta = 0.9$. As for the other SOTAs, we choose $\mu = 0.1$ in `FedProx` and $\alpha = 0.01$ in `FedDyn` from $\{0.001, 0.01, 0.1\}$; in `AdaBest`, we tune $\beta \in \{0.8, 0.9\}$ and $\mu \in \{0.01, 0.02\}$ and pick $(0.9, 0.02)$ for CIFAR10 and $(0.8, 0.02)$ for CIFAR100. The training proceeds for $10k$ rounds on CIFAR10 and $20k$ rounds on CIFAR100.

Table 11: FedSAM and FedASAM hyper-parameters

Dataset	Distribution	SAM	ASAM	
		ρ	ρ	η
CIFAR10	$\alpha = 0$	0.1	0.7	0.2
	$\alpha = 0.05$	0.1	0.7	0.2
	$\alpha = 100$	0.02	0.05	0.2
CIFAR100	$\alpha = 0$	0.02	0.5	0.2
	$\alpha = 0.5$	0.05	0.5	0.2
	$\alpha = 1000$	0.05	0.5	0.2
CIFAR100-PAM	$\alpha = 0.1$	0.05	0.5	0/0.2
LANDMARKS-USER-160K	-	0.05	0.5	0/0.2
CITYSCAPES	het/unif	0.01	0.1	0.2
IDDA	het/unif	0.01	0.5	0.2

Mixup/Cutout: Following the setup of [87], we fix $\alpha_{\text{mixup}} = 1$, resulting in λ uniformly distributed between 0 and 1. As for Cutout instead, we select a cutout size of 16×16 pixels for CIFAR10 and 8×8 for CIFAR100, as done by [16].

SAM/ASAM: The parameter ρ of SAM is searched in $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5\}$. As for ASAM, the value of ρ is tuned in $\{0.05, 0.1, 0.2, 0.5, 0.7, 1.0, 2.0\}$ and $\eta \in \{0.0, 0.01, 0.1, 0.2\}$. The choices made for each dataset and α are shown in Table 11. There is no distinction of values as clients vary per round.

SWA: We test SWA's starting round in $\{5\%, 25\%, 50\%, 75\%\}$ of the rounds budget and as expected [33] the best contribution is given if applied from 75% of the training onwards (see Appendix E). We set the value of the learning rate γ_1 to 0.01 and test $\gamma_2 \in \{10^{-5}, 10^{-4}, 10^{-3}\}$, selecting $\gamma_2 = 10^{-4}$. The cycle length c is tested in $\{5, 10, 20\}$ and set to 10 for CIFAR10 and 20 for CIFAR100. Table 12 summarizes the choices.

CIFAR100-PAM The hyper-parameters follow the same choice of [64] (see Table 10). We report accuracy at $5K$ and $10K$ communication rounds.

Mixup/Cutout: Same as CIFAR100.

SAM/ASAM: We search hyperpameters in the same values as CIFAR100. For ρ we found 0.05 and 0.5 to be the best values respectively for SAM and ASAM in all configurations. For ASAM we found that $\eta = 0.2$ is working fine when cutout or no augmentations are applied, while $\eta = 0$ works best in the case of Mixup.

SWA: Same as CIFAR100.

Landmarks-User-160k We start from the hyper-parameters proposed by [31]. In contrast with the original paper, we found that FedAvgM with momentum $\beta = 0.9$ is unstable with 10 participating clients and requires reducing the server learning rate to 0.1 to train the model. Better performance and faster convergence can be obtained with 50 clients per round and $\beta = 0.9$. However, we use 10 clients per round and FedAvg as the baseline because of our limited resources and to maintain consistency with other experiments. All hyper-parameters are described in Table 10.

Table 12: SWA hyper-parameters

Dataset	c	γ_1	γ_2	Start round
CIFAR10	10	10^{-2}	10^{-4}	7500
CIFAR100	20	10^{-2}	10^{-4}	15000
CIFAR100-PAM	5	10^{-2}	10^{-4}	15000
LANDMARKS-USER-160K	5	10^{-1}	10^{-3}	3750/5000
CITYSCAPES	5	$5 \cdot 10^{-2}$	$5 \cdot 10^{-4}$	1125
IDDA	5	10^{-1}	10^{-3}	1125

SAM/ASAM: The parameter ρ of SAM is searched in $\{0.01, 0.05, 0.1\}$. As for ASAM, the value of ρ is tuned in $\{0.1, 0.3, 0.5\}$ and $\eta \in \{0.0, 0.1, 0.2\}$.

SWA: We tested both SWA starting at the 75% and 100% of training, *i.e.* the 3750-*th* and 5000-*th* rounds. We tested different combinations of cycle lengths $c \in \{5, 10, 20\}$ and learning rate $\gamma_2 \in \{10^{-2}, 10^{-3}, 10^{-4}\}$. The best performing learning rates (γ_1, γ_2) are respectively $(10^{-1}, 10^{-3})$ and the cycle length is 5.

Cityscapes and IDDA For both Cityscapes and IDDA, we maintain the choice of hyper-parameters of [19]. The clients' initial learning rate is 0.05 on Cityscapes and 0.1 on IDDA, the weight decay is $5 \cdot 10^{-4}$ on Cityscapes, while it is not used on IDDA, 2 local epochs, the client optimizer is SGD with momentum 0.9. Differently from [19], we do not use mixed precision, thus the batch size is reduced from 16 to 8. A polynomial learning rate scheduler is applied locally, following [83]. The optimization is based on the Online Hard-Negative Mining [69], which selects the 25% of the pixels having the highest cross-entropy loss. The training is spanned across $1.5k$ rounds.

SAM/ASAM: The parameter ρ of SAM is searched in $\{0.01, 0.05, 0.1\}$. As for ASAM, the value of ρ is tuned in the set $\{0.05, 0.1, 0.5\}$ and $\eta \in \{0.0, 0.1, 0.2\}$.

SWA: Following the setup established for the CIFAR datasets, SWA starts at the 75% of training, *i.e.* the $1125th$ round. The learning rates (γ_1, γ_2) are respectively $(10^{-1}, 10^{-3})$ for IDDA and $(5 \cdot 10^{-2}, 5 \cdot 10^{-4})$ for Cityscapes. The cycle length is 5 for both datasets.

C.3 Plotting the Loss Landscapes

In the main text, we introduced both 2-D (Fig. 2 of the main text) and 3-D plots of the loss landscapes (Fig. 1 of the main text). Implementation details follow.

2D Loss Landscape Following the indications of [22,59]:

1. We choose three weight vectors $\theta_1, \theta_2, \theta_3$ and use them to obtain two basis vectors $\vec{u} = (\theta_2 - \theta_1)$ and $\vec{v} = (\theta_3 - \theta_1) - \frac{\langle \theta_3 - \theta_1, \theta_2 - \theta_1 \rangle}{||\theta_2 - \theta_1||^2} \cdot (\theta_2 - \theta_1)$.
2. Then, the normalized vectors $\hat{u} = u/||u||$ and $\hat{v} = v/||v||$ form an orthonormal basis in the plain containing $\theta_1, \theta_2, \theta_3$.

3. We now define a Cartesian grid of $N \times N$ points in the basis $\hat{u}, \hat{v}$. In our case, $N = 21$.
4. For each point of the grid, the corresponding weights are computed and the loss is consequently evaluated with the resulting network. For each point P of the grid having coordinates (x, y), the corresponding weights are computed as $P = \theta_1 + x \cdot \hat{u} + y \cdot \hat{v}$. As a consequence, θ_1 is the reference and can be found in the origin $(0, 0)$.

We adapted the code of [22][4] to our scenario.

3D Loss Landscape The plots in Fig. 1 in the main text are generated using the code of [46][5], modified to fit our datasets and models. Given a network architecture and its pre-trained parameters, the loss surface is computed along random directions near the optimal parameters.

C.4 Computing Hessian Eigenvalues

We refer to [60] for computing both the local and the top 50 Hessian eigenvalues (Figs. 4,5 in the main text) with Stochastic Power Iteration method [80] with maximum 20 iterations per run.

D Results on Corrupted CIFAR10 and CIFAR100

In Fig. 8, we compare the performance obtained by `FedAvg`, `FedSAM`, `FedASAM`, `FedAvg` + `SWA`, `FedSAM` + `SWA` and `FedASAM` + `SWA` on CIFAR10-C and CIFAR100-C as α varies. All results tell us that `ASAM` (alone or combined with `SWA`) is the algorithm with the best generalization capabilities, as already seen in Sec. 5.2 of the main text.

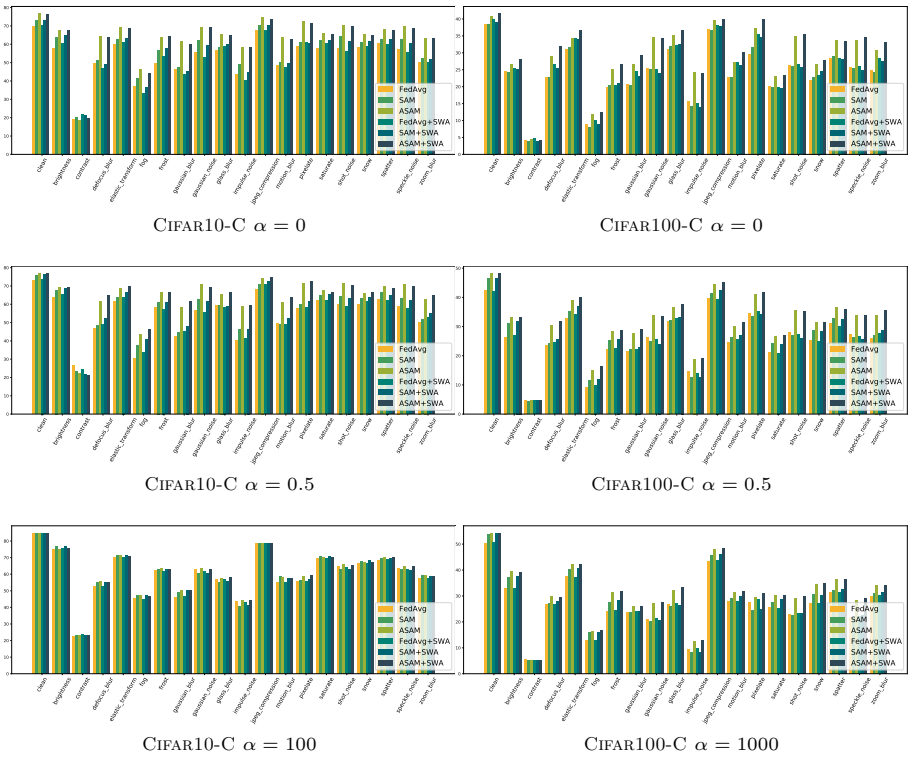
E Ablation Studies

In this Section, we present our ablation studies on server-side optimizers, `SAM`, `ASAM` and `SWA`, moved from the main text due to space constraints.

E.1 Ablation Study on Server-Side Optimizers

To choose the best server-side optimizer, we test SGD, Adam and AdaGrad on the heterogeneous ($\alpha = 0$) and homogeneous ($\alpha = 1k$) versions of CIFAR100 with 5 clients per round. Following [64], we set $\beta_1 = \beta_2 = 0$ for AdaGrad and $\beta_1 = 0.9, \beta_2 = 0.99$ for Adam. As Table 13 shows, SGD with learning rate 1, *i.e.* `FedAvg`, is certainly the best choice to have acceptable performances both in the homogeneous scenario and above all in the heterogeneous one.

[4] https://github.com/timgaripov/dnn-mode-connectivity
[5] https://github.com/tomgoldstein/loss-landscape



CIFAR10-C $\alpha = 0$ CIFAR100-C $\alpha = 0$

CIFAR10-C $\alpha = 0.5$ CIFAR100-C $\alpha = 0.5$

CIFAR10-C $\alpha = 100$ CIFAR100-C $\alpha = 1000$

Fig. 8: Domain generalization in FL. Results with 20 clients, severity level 5 on CIFAR10-C and CIFAR100-C.

E.2 Ablation Study on SAM and ASAM

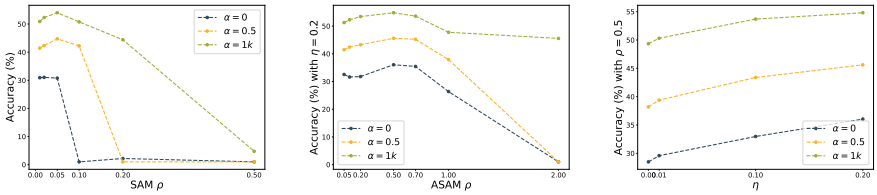
We present here an analysis on the sensitivity of the model to the hyper-parameters ρ and η in ASAM and ρ in SAM (Fig. 9), having as a reference the setting with 5% clients participation on CIFAR100. Regardless of the distribution, we can see that high values of SAM's ρ lead to a fast decline in performance (Fig. 9a), meaning that the algorithm handles smaller neighborhoods better. On the other hand, ASAM allows us to have more freedom and expand the size of the neighborhood up to the value of $\rho = 0.5$ (Fig. 9b), index of the greater robustness of the method. In Fig. 9c, we notice that the performances improve linearly as η increases, where η is a hyper-parameter balancing the trade-off between stability and adaptivity.

E.3 Ablation Study on SWA

SWA adds two new concepts to the standard federated training: the average of stochastic weights collected along the trajectory of SGD (Eq. 8) and the cyclical learning rate (Eq. 7), which decreases from γ_1 to γ_2 according to the cycle length c, transmitted as additional information to the clients of each round. Our ablation

Table 13: Final accuracy (%) using different server-side optimizers with varying learning rate (LR) on CIFAR100 @ $20k$ rounds. 5% clients participation. In bold the best results on both $\alpha = 0$ and $\alpha = 1k$.

Optimizer	LR	$\alpha = 0$	$\alpha = 1k$
`SGD`	1	**30.25**	**49.92**
	0.1	14.09	40.43
	0.01	2.67	11.35
	0.001	1.20	1.12
`Adam`	1	1.00	51.73
	0.1	29.75	51.62
	0.01	13.72	40.12
	0.001	2.60	11.31
`AdaGrad`	1	1.00	1.00
	0.1	1.77	46.74
	0.01	26.25	51.44
	0.001	9.70	32.01



(a) (b) (c)

Fig. 9: Results on CIFAR100, 5% clients participation. **(a)** Sensitivity to `SAM`'s parameter ρ. **(b)-(c)** Sensitivity to `ASAM`'s parameters ρ (with fixed $\eta = 0.2$) and η (with fixed $\rho = 0.5$) as α varies.

studies aim to understand which of these two components has the greatest impact on the achieved stability and increased model performance. We compare the results obtained by `SWA` with $c > 1$ with those reached when the learning rate is kept constant, *i.e.* $c = 1$, and when the server-side average of the collected weights is not applied while maintaining $c > 1$, *i.e.* changing only the clients' learning rate cyclically (Table 14). We point out that using $c = 1$ and not applying the average brings us back to the standard federated setting. We discover that the server-side average gives the major contribution, which helps in stabilizing learning, while the cycle length does not particularly affect the results. Since the best results in the most difficult scenarios (*i.e.* low value of both α and number of participating clients on CIFAR100) are reached when $c > 1$, we prefer the cyclical learning rate to the constant one in further experiments.

In addition, in Table 15 we report the differences in results when applying `SWA` from $\{5\%, 25\%, 50\%, 75\%\}$ of the training onwards on `FedAvg` with 5 clients per round, showing that a longer pre-training of the network leads to the greater effectiveness of this algorithm.

Table 14: SWA ablation study: comparison between cyclical ($c > 1$) and constant learning rate ($c = 1$) and contribution given by averaging stochastic weights. Highlighted in bold the best result for each combination (Algorithm, α, participating clients).

Dataset	Algorithm	WeightsAvg	c	$\alpha = 0$			$\alpha = 0.5/0.05$			$\alpha = 1k/100$		
				$5cl$	$10cl$	$20cl$	$5cl$	$10cl$	$20cl$	$5cl$	$10cl$	$20cl$
CIFAR100	FedAvg	✓	20	**39.34**	39.74	39.85	**43.90**	**44.02**	42.09	50.98	50.87	50.92
	FedSAM			**39.30**	**39.51**	39.24	**47.96**	**46.76**	**46.47**	53.90	53.67	**54.36**
	FedASAM			42.01	**42.64**	41.62	**49.17**	**48.72**	48.27	53.86	54.79	54.10
	FedAvg	✓	1	38.86	**39.82**	**40.19**	43.86	43.93	**42.67**	**51.33**	**51.05**	**51.11**
	FedSAM			38.58	39.20	**39.37**	47.29	46.34	46.40	53.88	**53.70**	**54.36**
	FedASAM			**42.50**	42.40	**41.76**	48.67	48.50	47.95	54.16	**55.07**	54.19
	FedAvg	✗	20	30.68	34.86	37.42	40.34	42.40	41.89	50.06	50.21	50.81
	FedSAM			31.51	35.87	37.81	44.08	45.80	46.43	53.76	53.46	54.28
	FedASAM			36.85	39.76	41.03	46.34	48.06	**48.38**	54.21	55.06	54.22
	FedAvg	✗	1	30.25	36.74	38.59	40.43	41.27	42.17	49.92	50.25	50.66
	FedSAM			31.04	36.93	38.56	44.73	44.84	46.05	**54.01**	53.39	53.97
	FedASAM			36.04	39.76	40.81	45.61	46.58	47.78	**54.81**	54.97	**54.50**
CIFAR10	FedAvg	✓	10	69.71	69.54	70.19	73.48	72.80	**73.81**	84.35	84.32	84.47
	FedSAM			74.97	73.73	73.06	76.61	75.84	76.22	84.23	84.37	84.63
	FedASAM			76.44	**75.51**	76.36	76.12	76.16	76.86	84.88	84.80	**84.79**
	FedAvg	✓	1	**69.88**	**69.83**	**70.72**	**73.91**	**73.12**	73.07	**84.90**	84.47	**84.67**
	FedSAM			**75.17**	**74.00**	**73.53**	**76.93**	**76.06**	**76.55**	**84.53**	84.54	84.77
	FedASAM			**76.80**	75.48	**76.84**	**76.87**	**76.30**	**77.55**	**85.09**	**85.06**	84.73
	FedAvg	✗	10	61.41	63.96	67.39	67.17	69.88	72.19	84.18	84.15	84.45
	FedSAM			70.66	71.14	73.04	73.93	74.96	76.20	84.23	84.40	84.69
	FedASAM			75.07	74.87	76.37	75.37	76.17	77.14	84.68	84.72	84.71
	FedAvg	✗	1	65.00	65.54	68.52	69.24	72.50	73.07	84.46	**84.50**	84.59
	FedSAM			70.16	71.09	72.90	73.52	74.81	76.04	84.58	**84.67**	**84.82**
	FedASAM			73.66	74.10	76.09	75.61	76.22	76.98	84.77	84.72	84.75

Table 15: SWA ablation study: comparison between SWA starting rounds when using FedAvg with 5 clients per round

Dataset	c	Start round	Test Accuracy (%)		
			$\alpha = 0$	$\alpha = 0.5/0.05$	$\alpha = 1k/100$
CIFAR100	20	1000	24.53	34.52	49.38
		5000	30.66	39.71	**51.52**
		10000	36.21	42.55	51.01
		15000	**39.34**	**43.90**	50.98
CIFAR10	10	500	55.57	60.50	79.09
		2500	60.34	65.72	81.49
		5000	66.22	70.55	83.79
		7500	**69.71**	**73.48**	**84.35**

F Tables Omitted in the Main Text

F.1 Heterogeneous FL Benefits Even More from Flat Minima - Additional Material

Table 16 completes the analysis introduced in Sec. 5.1 regarding the gains obtained in the federated scenario w.r.t. the centralized one. Here we report the results for $\alpha \in \{0, 1k\}$. As noted for $\alpha = 0$ (Table 5 in the main text), data augmentations fail in the federated heterogeneous scenarios ($\alpha \in \{0, 0.5\}$), but reasonably work in the homogeneous ones.

F.2 Data Augmentations with CIFAR10

Here we show the results obtained when applying Mixup and Cutout to CIFAR10 as the value of α, clients participation and algorithm change (Table 17). As demonstrated for CIFAR100 (Sec. 5.1), data augmentations do not improve generalization in a federated context, but on the contrary they seem to inhibit learning, leading to sometimes even worse results than FedAvg.

Table 16: Comparison of improvements (%) in centralized and federated scenarios ($\alpha \in \{0.5, 1k\}$, 5 clients) on CIFAR100, computed w.r.t. the reference at the bottom

Algorithm	Accuracy			Absolute Improvement			Relative Improvement		
	Centr.	$\alpha = 0.5$	$\alpha = 1k$	Centr.	$\alpha = 0.5$	$\alpha = 1k$	Centr.	$\alpha = 0.5$	$\alpha = 1k$
SAM	55.22	44.73	54.01	+3.02	+4.30	+4.01	+5.79	+10.64	+8.03
ASAM	55.66	45.61	54.81	+3.46	+5.18	**+4.89**	+6.63	+12.81	**+9.80**
SWA	52.72	43.90	50.98	+0.52	+3.47	+1.06	+1.00	+8.58	+2.12
SAM + SWA	55.75	47.96	53.90	+0.55	+7.53	+3.98	+1.06	+18.63	+7.97
ASAM + SWA	55.96	49.17	53.86	+3.76	**+8.74**	+3.94	+7.20	**+21.62**	+7.89
Mixup	58.01	35.10	55.34	+5.81	-5.33	+5.42	+11.13	-13.18	+10.86
Cutout	55.30	37.72	53.48	+3.10	-2.71	+3.56	+5.94	-6.70	+7.13

Centralized: **52.20** - FedAvg $\alpha = 0.5$: **40.43**, $\alpha = 1k$: **49.92**

Table 17: FedAvg, SAM, ASAM and SWA w/ strong data augmentations (Mixup, Cutout) on CIFAR10

	Algorithm	SWA	Aug	$\alpha = 0$			$\alpha = 0.5/0.05$			$\alpha = 1000/100$		
				5cl	10cl	20cl	5cl	10cl	20cl	5cl	10cl	20cl
CIFAR10	FedAvg	✗	None	65.00	65.54	68.52	69.24	72.50	73.07	84.46	84.50	84.59
	FedSAM	✗	None	70.16	71.09	72.90	73.52	74.81	76.04	84.58	84.67	**84.82**
	FedASAM	✗	None	73.66	74.10	76.09	75.61	76.22	76.98	84.77	84.72	84.75
	FedAvg	✓	None	69.71	69.54	70.19	73.48	72.80	73.81	84.35	84.32	84.47
	FedSAM	✓	None	74.97	73.73	73.06	76.61	75.84	76.22	84.23	84.37	84.63
	FedASAM	✓	None	**76.44**	**75.51**	**76.36**	**76.12**	**76.16**	**76.86**	**84.88**	**84.80**	**84.79**
	FedAvg	✗	Mixup	62.26	63.61	65.54	65.63	68.44	68.21	**82.38**	**84.46**	**83.58**
	FedSAM	✗	Mixup	67.35	69.32	69.78	70.34	72.98	72.54	81.88	82.24	82.25
	FedASAM	✗	Mixup	70.61	71.31	71.62	72.19	**72.84**	**72.72**	82.36	82.75	83.08
	FedAvg	✓	Mixup	66.31	66.89	66.26	69.79	69.12	68.80	82.27	82.88	82.67
	FedSAM	✓	Mixup	**72.42**	70.65	69.75	**73.36**	72.29	72.44	81.04	81.18	81.15
	FedASAM	✓	Mixup	72.37	**72.40**	**71.89**	72.54	72.36	72.32	81.86	81.70	81.92
	FedAvg	✗	Cutout	61.12	64.47	64.20	66.45	69.09	68.99	**83.77**	83.91	**84.31**
	FedSAM	✗	Cutout	63.69	66.30	67.25	67.66	71.39	70.67	83.03	83.84	83.49
	FedASAM	✗	Cutout	68.50	69.26	69.75	69.23	**71.91**	**71.28**	83.73	**84.10**	84.00
	FedAvg	✓	Cutout	65.54	65.60	65.79	69.94	69.55	69.63	83.35	83.39	83.64
	FedSAM	✓	Cutout	69.40	68.45	67.36	71.36	71.56	70.99	82.61	82.75	82.52
	FedASAM	✓	Cutout	**71.30**	**71.12**	**70.91**	**72.79**	71.76	71.09	83.06	83.31	83.11

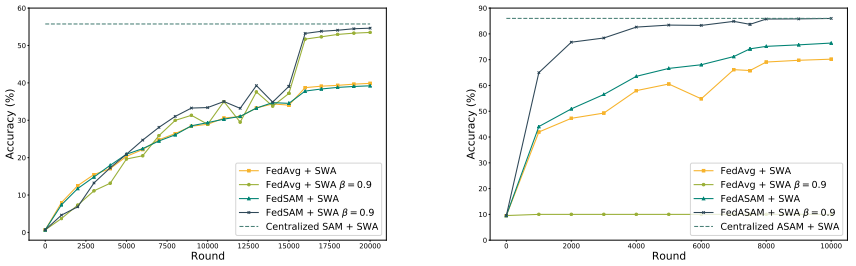
G Figures Omitted in the Main Text

All plots are best seen in colors.

Convergence plots As shown in Sec. 5.1, once combined with FedAvgM- *i.e.* server-side momentum $\beta = 0.9$ - SAM and ASAM allow to reach convergence even in the most heterogeneous scenarios on both CIFAR10 and CIFAR100. Fig. 10 shows the convergence plots of those runs. In addition, Fig. 11 compares the behavior of FedAvg, FedSAM, FedASAM and their combination with SWA on the most difficult setting, *i.e.* $\alpha = 0$ and 5 clients per round on both CIFAR datasets, highlighting the stability and the positive gap in performance introduced by SWA.

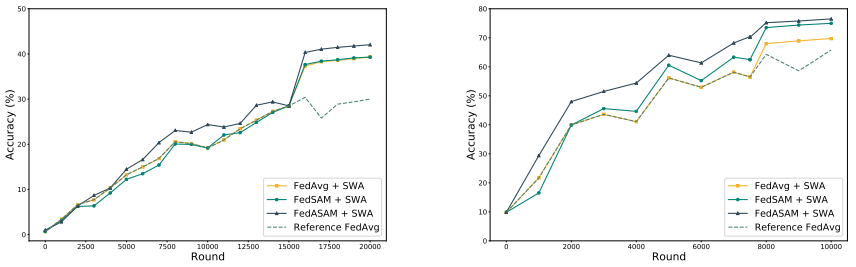
Loss Surfaces Fig. 12 shows the convergence points of three local models trained with $\alpha = 0.5$ on the corresponding test error surface, while Fig. 13 displays the train loss surfaces with $\alpha \in \{0, 0.5, 1000\}$. In addition, in Fig. 14 we compare the convergence points of FedAvg, FedSAM and FedASAM in the heterogeneous scenarios of CIFAR100, *i.e.* $\alpha \in \{0, 0.5\}$, proving that ASAM reaches the best local minimum.

Hessian Eigenvalues The top 50 eigenvalues of the global model trained with $\alpha = 0.5$ are showed in Fig. 15. Fig. 16 shows the complete comparison of the local Hessian eigenvalues partially shown in Sec. 3.2, introducing the values of λ^k_{max} $\forall k \in [K]$ resulting with SAM and $\alpha = 0.5$.



CIFAR100 CIFAR10

Fig. 10: Convergence plots with $\alpha = 0$, 20 clients. When combining `FedAvgM` or `FedSAM` (CIFAR100)/`FedASAM` (CIFAR10) with `SWA`, convergence is reached even in the most heterogeneous scenarios. `FedAvgM + SWA` applied to CIFAR10 fails to learn, while adding momentum to `FedASAM` significantly speeds up training.



CIFAR100 CIFAR10

Fig. 11: Convergence plots with $\alpha = 0$, 5 clients, highlighting the positive gap in performance and the stability introduced by `SWA` (both if applied on `FedAvg` but especially on `FedASAM`) in the most difficult setting.

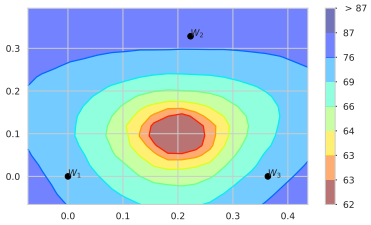
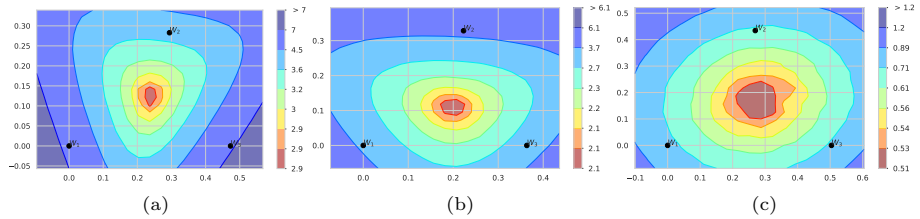
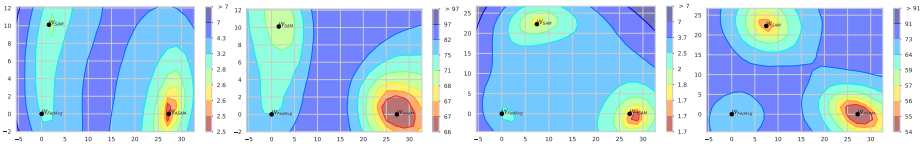


Fig. 12: Test error surface computed on CIFAR100 using three distinct local models trained with $\alpha = 0.5$ for $20k$ rounds.



(a) (b) (c)

Fig. 13: Train cross-entropy loss surfaces computed with three local models after $20k$ training rounds on CIFAR100. **(a)** $\alpha = 0$ **(b)** $\alpha = 0.5$ **(c)** $\alpha = 1000$.



(a) (b) (c) (d)

Fig. 14: Loss surfaces comparing the convergence points of `FedAvg`, `FedSAM` and `FedASAM` after $20k$ training rounds on CIFAR100. The minima reached by `SAM` and `ASAM` are found within low-loss neighborhoods. **(a)** Train loss surface $\alpha = 0$. **(b)** Test error surface $\alpha = 0$. **(c)** Train loss surface $\alpha = 0.5$. **(d)** Test error surface $\alpha = 0.5$.

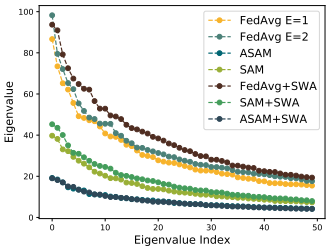
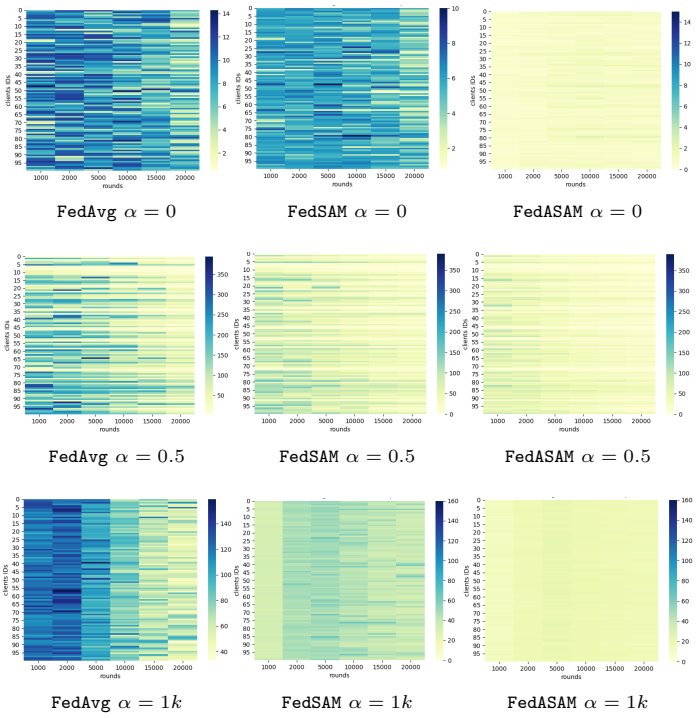


Fig. 15: Top 50 eigenvalues of the global model with $\alpha = 0.5$ on CIFAR100.



FedAvg $\alpha = 0$ FedSAM $\alpha = 0$ FedASAM $\alpha = 0$

FedAvg $\alpha = 0.5$ FedSAM $\alpha = 0.5$ FedASAM $\alpha = 0.5$

FedAvg $\alpha = 1k$ FedSAM $\alpha = 1k$ FedASAM $\alpha = 1k$

Fig. 16: Maximum Hessian eigenvalue computed for each client as rounds pass.