DecentNeRFs: Decentralized Neural Radiance Fields from Crowdsourced Images

Zaid Tasneem[1], Akshat Dave[2], Abhishek Singh[2], Kushagra Tiwary[2], Praneeth Vepakomma[2], Ashok Veeraraghavan[1], and Ramesh Raskar[2]

[1] Rice University
[2] Massachusetts Institute of Technology
{ztasneem,vashok}@rice.edu, {ad74,abhi24,vepakom,ktiwary,raskar}@mit.edu

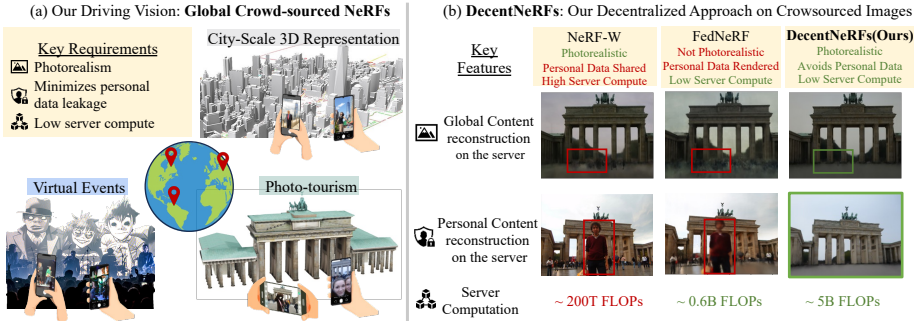


Fig. 1: Overview of our approach: Our framework leverages geotagged images of locations siloed worldwide in user photo galleries. It constructs neural radiance fields (NeRFs) for immersive 3D viewing in a decentralized manner, minimizing the reconstruction of personal content at the server. (**Left**): We showcase the potential to generate global-scale 3D scene representations for applications like city-scale NeRFs, immersive past event experiences, and virtual photo-tourism. We highlight the capture of diverse viewpoints across users. (**Right**): We compare our approach against centralized and existing decentralized baselines for a photo-tourism example. We demonstrate our approach's optimal tradeoffs between low-server compute and photorealism while minimizing personal content reconstruction on the server.

Abstract. Neural radiance fields (NeRFs) show potential for transforming images captured worldwide into immersive 3D visual experiences. However, most of this captured visual data remains siloed in our camera rolls as these images contain personal details. Even if made public, the problem of learning 3D representations of billions of scenes captured daily in a centralized manner is computationally intractable. Our approach, DecentNeRF, is the first attempt at decentralized, crowd-sourced NeRFs that require $\sim 10^4\times$ less server computing for a scene than a centralized approach. Instead of sending the raw data, our approach requires users to send a 3D representation, distributing the high computation cost of training centralized NeRFs between the users. It learns photorealistic scene representations by decomposing users' 3D views into personal and global NeRFs and a novel optimally weighted aggregation of only the latter. We validate the advantage of our approach to learn NeRFs with photorealism and minimal server computation cost on structured synthetic and real-world photo tourism datasets. We further analyze how secure aggregation of global NeRFs in DecentNeRF minimizes the undesired reconstruction of personal content by the server.

1 Introduction

Every day, more than 5 billion photos are captured worldwide, comprising multiple viewpoints of every monument, skyscraper, cafe, and concert on Earth. Neural radiance fields (NeRFs) present an exciting opportunity to process this massive data into immersive visual experiences at a global scale. However, most of these images remain siloed in personal camera rolls. Less than 2% of these captured photos are ever posted on the internet [28]. Even if these personal images were made public, learning NeRFs for billions of scenes captured daily at a global scale in a centralized fashion is computationally intractable. Therefore, to build immersive visual experiences at a global scale, NeRFs must be decentralized to handle high compute needs and avoid the undesired reconstruction of personal content by a central entity, all the while ensuring photorealism (Fig. 1(a)).

Our framework, DecentNeRF, is the first attempt towards decentralized NeRFs from crowd-sourced images. Images in public spaces are often composed of *global* content,e.g., a monument that we would like to share with the world and a *personal* content, such as a friend posing in front of the monument that we would like to keep to ourselves. Our key insight is that often global content is *static* across users, and the personal content is *dynamic*, i.e., varies from user to user. This association of global as static and personal as dynamic allows our approach to perform global-personal separation in the captured images. Our approach enables sharing only the global scene-specific 3D representations across users instead of sharing the combined (global+dynamic) image as in a conventional NeRF pipeline. In doing so, the server avoids the cost of centrally training NeRFs by distributing the NeRF training computation across users. It also minimizes the reconstruction of the undesired occlusions of personal user-specific content at the server.

For a particular scene, we model the multi-view visual data as a combination of a *global radiance field* for the 3D scene of interest and a *personal radiance field* for the user's personal information (transient across users). We propose federated learning [17] procedure to learn the global radiance field across users by aggregating only the user's global radiance field model (locally trained). Instead of uniformly averaging the users' weights as typical in federated learning [18], we propose a novel federation procedure where the per-user scaling is learned implicitly to maximize visual fidelity. To prevent the server from accessing the individual user's global radiance fields, we use a secure multi-party computation (SMPC) protocol [7] for aggregation. The secure aggregation minimizes the reconstruction of personal content by the server compared to existing approaches during initial rounds of federation.

Our proposed framework, DecentNeRF is the first work to analyze the decentralization aspects of radiance fields for real-world crowd-sourced images. NeRF-W [22] achieves the high visual quality of the public global scene from in-the-wild crowd-sourced images but requires all the personal user images to be transferred to a central server for training. This results in personal content directly being accessed by the server and high server compute(Fig. 1(b)). We address both these limitations in our work. Works on federated learning of NeRFs [15, 35] demon-

strate model compression and large scene modeling capabilities but assume the images of static scenes devoid of users' personal information. When dealing with crowd-sourced data, these approaches not only allow the server to access personal information without a secure aggregation protocol but also result in inaccurate reconstructions (Fig. 1(b)). Our approach demonstrates high visual quality in the global content compared to existing decentralized approaches, with $\sim 10^4\times$ lower compute than the centralized approach(Fig. 1(b)).

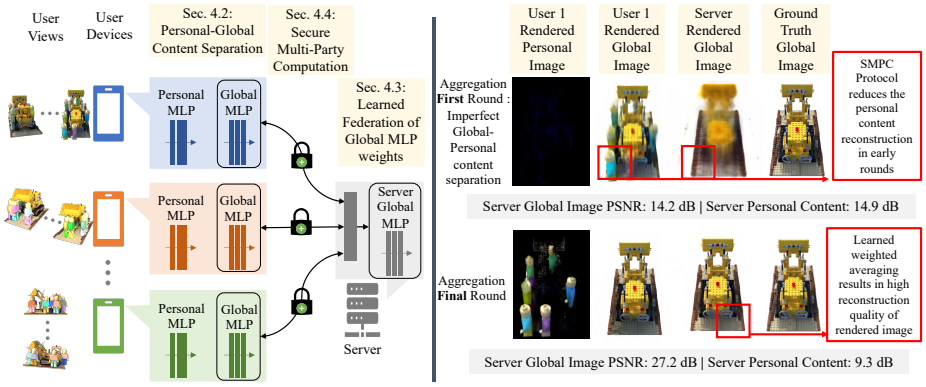
To summarize, we make the following contributions.

- We introduce DecentNeRF, the first approach to address the challenges of learning global 3D scene representations at scale from crowd-sourced images in a decentralized manner.
- Our method uses public-global separation and a novel learned federation scheme to achieve high-quality reconstruction of 3D scenes with very low server computing compared to prior works (Fig. 1(b)).
- Our method outperforms existing federated learning NeRF techniques on scene reconstruction quality while requiring $\sim 10^4\times$ lower compute than centralized approaches. We demonstrate this in simulated (Table 1) and real-world crowdsourced scenes (Table 2).
- We provide an analysis on how secure aggregation of user's global NeRFs in DecentNeRF reduces the reconstruction of personal content on the server Fig. 7 for both datasets compared to prior works.

2 Related Work

3D from Unstructured Image Collection. Generating 3D scene representations for novel-view synthesis from an extensive collection of unstructured images [32] has been explored in [9, 20, 22, 40]. However, these prior works are centralized by nature - all the captured images are sent to the server for training NeRFs. Our work, DecentNeRF, diverges from traditional work by focusing on decentralization to 1) distribute the training compute to the users and thus scale to billions of scenes and 2) avoid aggregating users' images, which could contain personal details. Our vision assumes NeRFs can be trained without users' attrition of mobile devices. Recent advancements in mobile NeRF renderings, such as [8, 10], demonstrate the promise for eventual mobile deployment.

Federated Learning. In Federated learning (FL) [17, 18], each client device trains the model parameters on-device using its own local data. The server then performs a weighted average of the models to obtain a global model and this process continues till convergence. The works in [15, 35] apply federated learning in the context of NeRFs for achieving the utility of 3D scene modeling. Since, NeRFs are 3D representations sharing them instead of raw data would still lead to reconstruction of the personal data. We instead model the local 3D scene representation as a combination of a global radiance field for the 3D scene-specific detaisls and a personal radiance field for the user-specific information. This decoupling approach is reminiscent of parameter decoupling in personalized federated learning [36]. However, unlike personalized federated learning, we



(a) DecentNeRF Overview (b) DecentNeRF rendering results for first and final federation round.

Fig. 2: Key features of our approach (a) Overview: Personal and global MLPs are trained on user devices to separate personal and global content from local images. After each training round, the server performs a learned federation of users' global MLPs using a secure MPC protocol and distributes the updated global MLP back to each user. (b) Results on reducing personal content leakage: We notice that users' global MLPs contain personal content during the initial rounds. Our secure MPC protocol ensures the server only sees the averaged global MLP from which the rendering of users' personal content is minimal. Over federation rounds, global and personal MLPs separate content through learned weighted averaging, enabling high-fidelity rendering from the server's global MLP.

consider each sample to be composed of personal and non-personal information. Hence, the goal is still to learn a single global consistent scene across multiple users.

Secure Aggregation. Accessing individual model updates from the users in FL could lead to data reconstruction attacks [16, 41]. Therefore, secure aggregation has been proposed to average model updates such that only the final averaged weights are revealed to the untrusted central server. This is accomplished by encrypting individual model updates by each user such that only the final average can be decrypted. To compute the average of encrypted user models, existing techniques rely on primitives such as secure multi-party computation(SMPC). It has been studied extensively in FL since the first practical construction by [6]. Recent works have explored different dimensions of secure aggregation such as communication [11] and synchronization [27,34]. We use SMPC-based secure aggregation [21] as a building block to perform weighted averaging over encrypted model updates.

3 Key Insights

Consider a scenario where users visit a famous restaurant in town at different times over months capturing images that are now saved on their personal photo galleries. In addition to visual content depicting the restaurant scene, these images likely contain personal content that users would not want to share publicly.

We aim to develop a decentralized approach where a server can learn a 3D representation of the restaurant, given such a cluster of users and their captured images. This is a challenging problem as the learned scene representation must encode both the appearance and 3D structure of the scene while not revealing the personal image content to the server. To create a global-level 3D representation, we can repeat this process for millions of users and locations (restaurants, monuments, etc.) which puts a compute constraint on the server learning the scene representation. This section highlights the key insights enabling our approach to achieve photorealistic 3D reconstruction with minimal server computing and undesired reconstruction of personal content.

Decentralization through Federated 3D Scene Representations. Neural radiance fields (NeRFs) excel at encoding 3D scene information using multilayer perceptrons (MLPs) [24]. Existing decentralized solutions collaboratively learn a shared NeRF MLP representation with each user's local views [15]. The users can refine the MLPs locally and in parallel, offloading compute from the server. The server only needs to aggregate user MLP updates into a combined shared MLP and transmit this back to users for further refinement. Over multiple federation rounds, this approach aims to reconstruct a 3D scene with the shared MLP. Such a federated method requires orders of magnitude less server computation than centralized approaches, aligning with our decentralization goal.

Existing Federated NeRF performs poorly on crowdsourced images. However, existing federated NeRF approaches assume input view consistency - that any 3D point observed from users' images is static. The underlying assumption is that all users took all images at the same instant, only capturing the global scene content and avoiding personal data. These assumptions do not hold for crowdsourced images taken over months and contain personal content like users, their food, or credit cards which are transient across users. Violations of these assumptions would hamper reconstruction quality (Fig. 5) and leak personal content from the shared MLPs (Fig. 7). We now provide key insights into how we can exploit the structure of these violations to learn photorealistic global 3D scene content in a decentralized manner.

1. **Personal-Global Content Separation.** Only the global scene-specific content is 3D view-consistent (static) across users such as the columns and most of the restaurant's interior. By definition, all other 3D content would be transient across users, be it non-personal like the wait staff or personal and sensitive like the user, their, or the credit card on the table. To leverage the juxtaposition between scene-specific and user-specific content, we propose encoding 3D appearance between personal and global MLPs in Sec. 4.2, capturing personal and global content, respectively. Only the global MLP is federated at the server to form the combined global MLP. This allows for high-quality reconstruction of global content over multiple rounds of federation as shown in Fig. 2b (bottom).
2. **Learned Federation.** Users likely have different data distributions - number of views, disparity, and user/scene content ratios. Naive federated averaging of global MLP [18] is suboptimal as shown in Sec. 6 and Fig. 4.

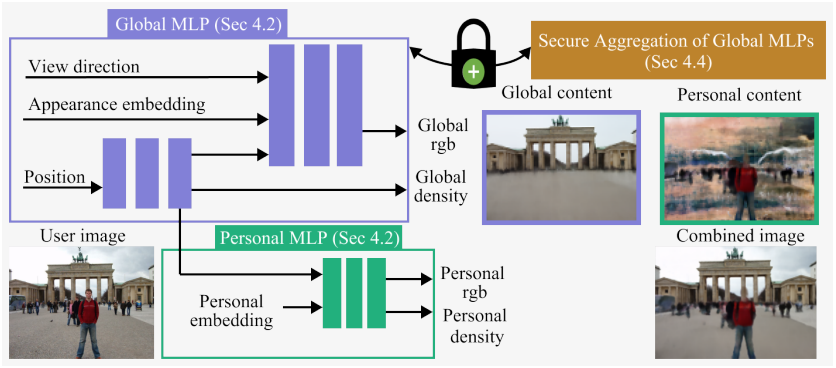


Fig. 3: DecentNeRF architecture: On user devices, we consider NeRFs with the following architecture where the personal MLP is always local to the user and the weights of global MLP are securely aggregated at the server. We also highlight what DecentNeRF learns to represent in its Global and Personal MLPs.

We instead learn aggregation weights over federation rounds in Sec. 4.3 for optimal reconstruction quality.

3. **Secure Multi-Party Computation (SMPC).** We notice that during the initial rounds, the user global MLPs encode both personal and global content as highlighted in Fig. 2b (top). This is because users initially have no notion of global or personal content without federation across users. If the server has access direct access to the users' global MLPs, which is the case for existing FL NeRF methods [15], it can faithfully render the personal content as shown in user 1 rendered global image in Fig. 2(b). To prevent the server from outright accessing the individual user's global MLPs, we use SMPC aggregation described in Sec. 4.4. Now, the server can only access averaged global MLP at each round. We analyze how these securely aggregated server global MLPs have minimal reconstruction of personal content during initial rounds in Fig. 2(b) (Top row-third column). We further analyze the reconstruction of personal content from aggregated server global MLPs for the case of varying numbers of users and overlap of users' views in Section 6.3.

Overall Compute. It is important to note that decentralized NeRF approaches, including ours, would have total computing (server+user) similar to a centralized approach (only server) for training NeRFs. However, in the case of decentralized systems, the training of NeRFs is distributed to the users to reduce server computing significantly. Our approach's compute and communication overheads are discussed in detail in the supplement.

4 Our Approach

In this section, we discuss the assumptions our approach makes, develop each of the three key insights described in Sec. 3, and show how they come together in our algorithm.

4.1 Preliminaries

We state the assumptions necessary for our approach:
1. Personal content is transient across users. We assume not all clients have taken images at the same instant, which is a realistic assumption for crowd-sourced images. It ensures that the personal content is transient and not static across users like the global content. **2. Sufficient overlap in views.** Our approach assumes that no single user has views that don't overlap with another user. We expand more on this in Sec. 6.3. **3. Known camera poses.** Our NeRF pipeline assumes known, accurate camera poses. In centralized settings, structure from motion (SfM) on image feature descriptors like SIFT can derive relative camera poses using COMLAP [31]. Decentralized SfM is an active research area [1, 12] and beyond the scope of this work.
Threat model. From a user's perspective, the server and other users are untrusted. However, we assume no collusion between users to leak more information than possible individually. Unlike differential privacy [13], which prevents identity leakage, we focus on preventing the reconstruction of user-specific semantic personal information by the server. The server can try to reconstruct a target user's personal information from the averaged model updates. We currently limit our scope to honest-but-curious servers who follow protocols but attempt to reconstruct raw data based on observations, leaving other attack types for future work.
NeRF Backbone [24] At its core, NeRF trains a multi-layer perceptron (MLP) that takes a 5D input vector: a 3D spatial coordinate $\mathbf{x} = (x, y, z)$ within the scene, and pitch θ and yaw ϕ parameters of the viewing direction $\mathbf{d}$. The MLP outputs a scalar density σ and an RGB color vector $\mathbf{c} = (r, g, b)$. NeRF rendering synthesizes images via volume rendering, sampling predicted color and density at many points along rays traversing the scene. For a particular ray corresponding to a pixel, the MLP predicts a color vector $\mathbf{c} = (r, g, b)$ and compares it with the ground truth RGB value, using the loss to train the MLP weights. We refer the reader to [24] for more details.

4.2 Personal-Global Content Separation

Consider a scene with K known users. For each user k, we model the scene using global and personal MLPs (Fig. 3) with the MLP weights denoted as $\mathbf{g}_k$ and $\mathbf{p}_k$ respectively. The personal MLPs are kept native to the user's device. In contrast, the global MLP captures shared global content (such as the landmark) across users via federation. Each MLP outputs its scalar density σ and an RGB color $\mathbf{c} = (r, g, b)$, so the rendered image is an alpha-composite of both output images.

Our architecture is inspired by NeRF in the Wild (NeRF-W) works in [9, 22], with each user essentially running NeRF-W locally. The global and personal MLPs are analogous to the static and transient MLPs in the context of a single user. However, the global-personal definition better suits the decentralized setting since NeRF-W's transient MLP will not separate all of the personal content from the user. Imagine a subject being recorded in front of a monument with

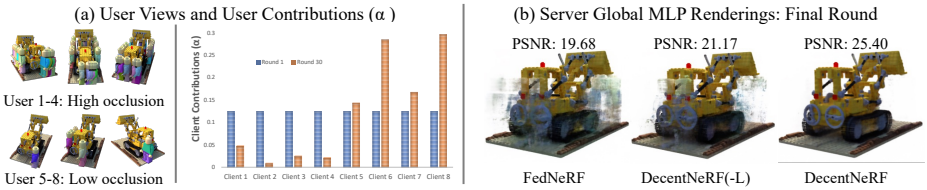


Fig. 4: Ablation on Learned Federation: We demonstrate that with Learned Federation of Global MLPs, our approach (DecentNeRF) learns to weigh clients with less occlusion over 30 rounds of training which leads to better reconstruction quality overall compared to one with FedAvg aggregation scheme i.e. DecentNeRF(-L) and FedNeRF.

multiple images. Since the subject would be static across all views within the user's images, it will not appear in the transient MLP even though it is transient across users. Our federation of global MLP across users over multiple rounds can separate this user-static but transient across user-specific content into the personal MLP branch (Fig. 3 (inset)) and lead to high-quality reconstructions
Prior works removing content from NeRFs use segmenting and inpainting personal content (e.g. people) [39]. In our case, the personal-global MLP separation helps us implicitly separate the personal and global content.

4.3 Learned Federation of Global MLPs

The global MLP weights at any aggregation round m, are defined as $\mathbf{g}^{(m)} = \sum \alpha_k^m \cdot \mathbf{g}_k^m$. In a naive federated learning approach, all users are weighted equally, $\alpha_k = \frac{1}{K}$. FedAvg [18] defines the users based on the number of pixels in their image data, $\alpha_k = \frac{p_k}{\sum_{k=1}^{K} p_k}$, where p_k is the total number of image pixels in the user data.

These existing definitions lead to suboptimal reconstruction performance as shown in Fig. 4 because all users have different data distributions - number of views, view disparity, user/scene content ratios and would be able to do different levels of global-personal content decomposition to benefit the server's global representation. We aim to learn the α implicitly over different merge rounds by the following equation

$$\alpha_k^{(m)} = \alpha_k^{(m-1)} - \eta \frac{\partial L}{\partial \alpha_k^{(m-1)}} \tag{1}$$

where η is the weighted averaging learning rate, and the cumulative loss L is defined as the sum of losses for each user k, $L = \sum_k L_k$, where $L_k(g_k^{(m-1)}, p_k^{(m-1)}, I_{\text{GT}})$ is the training loss defined in [22].

The loss implicitly rewards the user's weightage, α_k when they do a better global-personal decomposition in the previous round's aggregation. Combining the above with the update equation for α_k,

$$\frac{\partial L}{\partial \alpha_k}^{(m-1)} = \left(\sum \frac{\partial L_k}{\partial \mathbf{g}^{(m-1)}} \right) \cdot \mathbf{g}_k^{m-1} \tag{2}$$

Beyond the first round of aggregation, each user sends its own $\frac{\partial L_k}{\partial \mathbf{g}^{(m-1)}}$, which is aggregated by the server and sent back to the users. This aggregated value

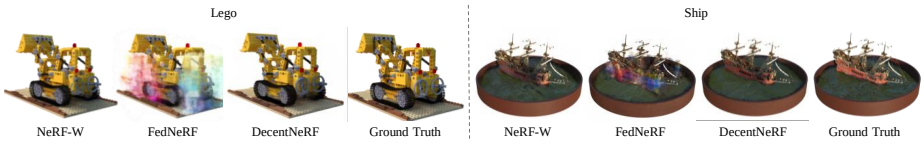


Fig. 5: Qualitative results on Novel Blender dataset: DecentNeRF simultaneously removes unwanted personal content ('lego persons') while reconstructing fine details of global content ('lego excavator' and 'ship'). In contrast, FedNeRF [15] hallucinates 'blobs' where personal content exists in user views.

gets multiplied with the previous round's user weights at the user to update each user's α_k at every round.

	Lego				Ship			
	Photorealism			Decentralization	Photorealism			Decentralization
	PSNR↑	SSIM↑	LPIPS↓	Server FLOPs↓	PSNR↑	SSIM↑	LPIPS↓	Server FLOPs↓
NeRF-W	27.09	0.934	0.0513	≈ 130 T	26.35	0.8661	0.1765	≈ 220T
FedNeRF	17.7	0.858	0.2003	≈ 0.6 B	20.92	0.8281	0.2368	≈ 1 B
DecentNeRF	26.25	0.92	0.0652	≈ 4.8 B	24.58	0.8554	0.1901	≈ 8 B

Table 1: Quantitative results on Novel Blender dataset: Best and second best results for are highlighted in blue and orange respectively. A centralized approach [22] achieves the best photorealism performance while using 10^5 more server compute than decentralized approaches. FedNerf [15] even though it only is 8x server compute cheaper than DecentNeRF and performs poorly on photorealism metrics, In contrast, DecentNeRF obtains comparable photorealism to the centralized approach with several orders of magnitude lower sever compute.

4.4 Secure Multi-Party Computation (SMPC)

From the user's perspective, in every round, there are exactly two kinds of messages that each user k communicates with the server - 1) weights of the global MLP, $\mathbf{g}_k$, and 2) gradients with respect to the per-user loss, $\frac{\partial L_k}{\partial \mathbf{g}_k^{(m-1)}}$. They also compute 3) gradient updates of $\alpha\left(\frac{\partial L^{(m-1)}}{\partial \alpha_k}\right)$. We aim to obfuscate the user communications such that only the final result aggregated over all the users is visible to the server. From the server's perspective, it needs to compute - 4) the weighted average of the obfuscated weights sent by the users $\mathbf{g}^{(m)}$, and the cumulative loss $\sum \frac{\partial L_k}{\partial \mathbf{g}^{(m-1)}}$. Each of these four functional requirements is satisfied using secure multi-party computation [21]. We now describe the protocols in more detail.

$\mathsf{send\_wts}(\mathbf{g}_\mathbf{k}^{(\mathbf{m})})$: Each user obfuscates its local weights (multiplies with α_k) by applying a random mask or encrypting data using additively homomorphic encryption techniques. This randomization ensures that when the obfuscated weights are aggregated, the original values can not be recovered from those weights while providing the means to calculate the weighted average accurately.

$\mathsf{weight\_avg}(\mathbf{g})$: This function executes the computation of the weighted average of the obfuscated weights sent by the users $(\mathbf{g}^{(\mathbf{m})})$. By employing secure multi-party computation techniques, the server aggregates the masked weights received from all users.

send_grads() This function is similar to send_wts in its design as each user communicates its local gradients by performing randomized obfuscation to its gradients in a similar way it performs obfuscation of the weights.

compute_grads() This function is relatively simple where each user calculates the update step described in equation (2) and then updates α_k accordingly.

4.5 Summarizing our algorithm

Putting together the building blocks developed above, our algorithm can be summaries as (We refer the reader to the supplementary material for pseudo-code of the detailed algorithm):

- Assumes random initialization of global $\mathbf{G^{(0)}}$ and personal MLP weights $\mathbf{P^{(0)}}$ at all K users.
- (On the user) For each round the users return the current $\mathbf{g}_k^m$ and previous rounds global MLP along with loss gradients of previous round global MLP weights $\frac{\partial L_k}{\partial \mathbf{g}^{(m-1)}}$. They also calculate the gradient updates for α_k.
- (On the server) For each round the weights and previous rounds gradients are securely aggregated and sent back to the user.

5 Experimental details

5.1 Datasets

Novel Blender Dataset. To do a controlled study of the effects of varying occluding personal content, number of users, and overlap of user views, we introduce our own synthetic dataset. We do it by adding personal content in the form of Lego people to the original blender dataset from [24]. The equivalent of this would be actual people/occlusions in a real-world setting. Unless mentioned otherwise, the dataset comprises 360-degree rotation with non-IID partitioning of 18 degrees in yaw angle rotation of 100 images distributed equally among 20 users. Each view contains six Lego people, with one static person per user's view representing personal content. More details and examples from the novel dataset are shown in the supplementary.

Phototourism Dataset. To evaluate the decentralization benefits of our approach on a real-world dataset we also evaluate 3 scenes on novel-view synthesis from unconstrained image collections: Brandenburg Gate, Trevi Fountain, and Sacre Couer [32]. Our setup has 20 clients each with 10 images of 2x downsampled resolution. Please refer to the supplementary for more details.

5.2 Baselines

We compare our solution with a centralized approach (NeRF-W [22]) and an existing decentralized approach (FedNeRF [15]). We also run an ablation of our approach without the learned federation federation.

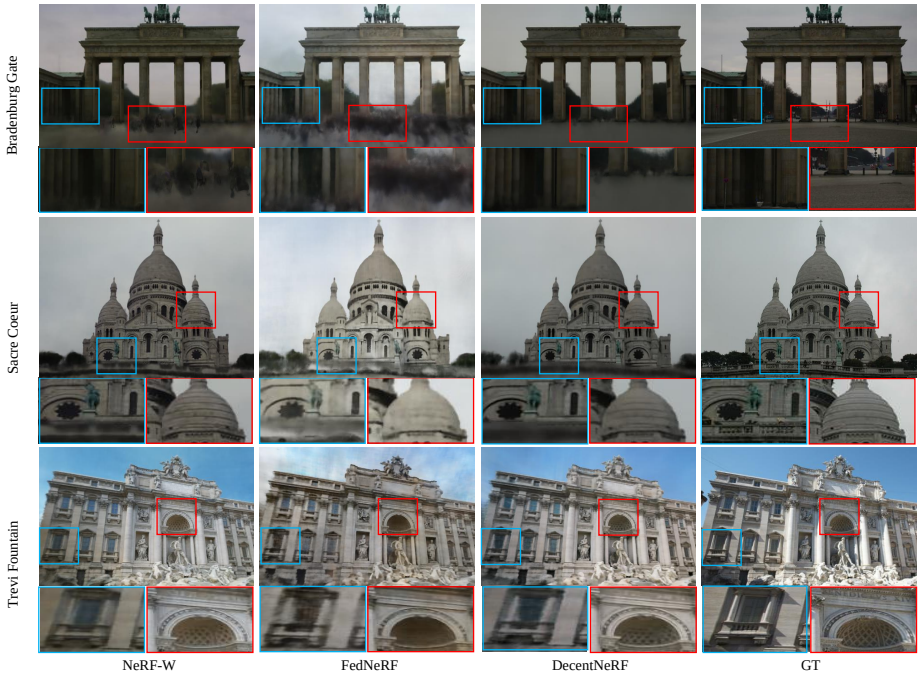


Fig. 6: Qualitative results on Phototourism [32] dataset: DecentNeRF is simultaneously able to reduce personal content (top row) from the scenes and capture fine details (middle-bottom). It does well in reconstruction in areas where personal content is present in user images highlighted by blue squares specially in top and middle rows.

5.3 Evaluation Metrics

Photorealism. We use PSNR, SSIM, and LPIPS on the global MLP renders as described in prior works [15,24] for our novel blender dataset. For Phototourism, we use the same protocol as in [22] for fitting the appearance of the left half of the validation images and evaluating on the right half for DecentNeRF and NeRFW.

Decentralization. To do a fair comparison we use the FLOPs computation from the SOTA NeRF training [25] and calculate server and per-user compute. Server compute includes aggregation, reconstruction cost for SMPC protocols, and user weight update. For users, it is NeRF training compute. For our application only server compute is relevant and analyzed in the main paper. Please refer to the supplementary material for more details on server and client costs including communication costs

Personal content. As we know the ground-truth mask of the personal content, we use NeRF-centric reconstruction metrics like PSNR to measure personal content for the novel blender dataset. These empirical evaluations have been used in other scene inversion works in [26, 29, 37]. For the real-world phototourism dataset, we go one step further and use a pre-trained Faster R-CNN ResNet-50 FPN [14] object detector and report the number of people detected as a metric

	Bradenburg Gate				Trevi Fountain				Sacre Coeur			
	Photorealism			Decentralization	Photorealism			Decentralization	Photorealism			Decentralization
	PSNR↑	SSIM↑	LPIPS↓	Server FLOPs↓	PSNR↑	SSIM↑	LPIPS↓	Server FLOPs↓	PSNR↑	SSIM↑	LPIPS↓	Server FLOPs↓
NeRF-W	25.18	0.901	0.1927	≈ 200 T	21.49	0.7206	0.2866	≈ 250 T	20.58	0.7835	0.2305	≈ 280 T
FedNeRF	18.92	0.8158	0.3528	≈ 0.6 B	16.55	0.5774	0.4352	≈ 1.2 B	15.41	0.6555	0.4072	≈ 1.5 B
DecentNeRF	24.62	0.8802	0.2571	≈ 5 B	20.61	0.6501	0.3963	≈ 9 B	19.22	0.7259	0.3126	≈ 10 B

Table 2: Quantitative performance on Phototoursim dataset [33]: Best and second best results are highlighted in blue and orange respectively. DecentNeRF is effective for real-world crowd-sourced datasets compared to other decentralized approaches [15] while using 10^4 less compute than centralized approaches [22].

of personal content. We render views for the server-accessible MLPs which would be users' Global MLPs for FedNeRF and Server Global MLP for DecentNeRF.

5.4 Implementation Details

We use a PyTorch Lightning implementation of NeRF-W [19] for centralized evaluation. We implement the secure aggregation protocol, SecAgg [6], using Flower [4]. Our approach differs from FedNeRF [15] as we employ a Personal MLP, appearance embedding, secure aggregation, and learned weighted averaging. Please refer to our supplementary material for training and model details.

6 Results and Analysis

6.1 Ablation on Learned Federation

To demonstrate the advantage of Learned Federation of weights in terms of photorealism we created a Blender scene with 8 users - 4 with heavy occlusion, and 4 with less occlusion overlapping the former. In Fig. 4 we compare our novel learned federation approach with ablation using FedAvg aggregation strategy trained on this scene. We also compare with FedNeRF. Our learned federation learns to weigh users with less occlusion, boosting PSNR on global content (GT). In real-world scenarios, DecentNeRF would access clients with varying occlusion levels and learn to weigh them explicitly, without predefined heuristics. This allows for better optimization than existing FedAvg.

6.2 Photorealism and Decentralization analysis

Novel Blender Dataset. We do a quantitative and qualitative analysis of photorealism and decentralization on our novel Blender dataset in Table 1 and in Fig 5. Our results highlight that our approach requires $\sim 10^5\times$ server compute than a centralized approach with only a minimal decrease in photorealism as opposed to FedNeRF which breaks in the presence of occlusion/personal content.
Phototourism Dataset. We do a quantitative comparison of DecentNeRF reconstruction performance on real-world crowdsourced Phototourism Dataset for FedNerF, NeRFW, and our approach. In Fig. 6 and Table 2 the results tell a compelling story that we are able to perform as well as the best performing centralized approach (NeRFW) in real-world scenes while several orders of magnitude less server compute.

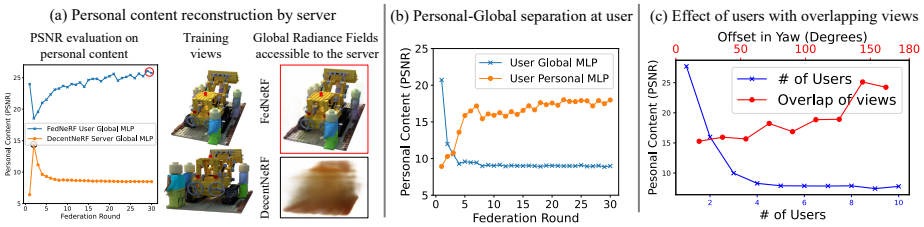


Fig. 7: Personal content reconstruction: (a) PSNR evaluation on personal content for server-accessible Radiance fields for FedNeRF and DecentNeRF per round, rendering the worst performance rounds for each on the right. We notice in the server renderings that FedNeRF cannot prevent the server from reconstructing the personal content. At the same time, DecentNeRF reduces the reconstruction of personal content(by 10 dB) even in initial rounds. (b) PSNR evaluation of personal content for users' global and personal MLP for each round demonstrates DecentNeRFs ability to separate personal and global content over multiple rounds of federation. (c) We evaluate the effect of (1) the number of users and (2) the overlap in the views of two users views on the faithful rendering of personal content by the server for DecentNeRF.

6.3 Analysis of Personal Content Reconstruction by the Server

Personal content reconstruction on Blender. PSNR curves for server-accessible MLPs, i.e., user global MLP for FedNeRF and server global MLP for DecentNeRF in Fig 7 (a) demonstrates a significant reduction of reconstruction (by 10dB) of personal content by the server for DecentNeRF. Visual inspection of server-accessible renderings for DecentNeRF shows the inability of the server to reconstruct the personal content faithfully.

Personal-global separation. The PSNR curves of users' Global and Personal MLPs on personal content rendering, as shown in Fig 7 (b), demonstrate that personal content initially appears in the users' global MLP. However, over multiple rounds of federation, our two MLP approaches shift the personal content onto the users' personal MLPs. We strongly encourage the reader to inspect the supplemental video. It shows free-view renderings of our personal and global MLPs for different rounds and demonstrates our ability to separate content.

Worst case reconstruction of personal content. We notice that the users' global MLP contains personal content during initial rounds. However, because the server can only access the securely aggregated MLPs for DecentNeRF, it cannot reconstruct personal content faithfully due to averaging. We further analyze this worst-case reconstruction of personal content in Fig. 7 (c) by (1) *varying the number of users* with the same overlap in views and (2) *changing the overlap of two users* by changing their view in yaw for the blender "lego" scene. Our analysis highlights two extremes: (1) if there are two users with a significant overlap of views, then the server cannot reconstruct the personal content of either user. (2) If two users have no overlapping views, i.e., they capture entirely separate areas of the scene, then the server's ability to reconstruct personal content increases considerably.

Personal content reconstruction on Phototourism. As noted in Fig 8 (a) a person detection NN [14] fails to detect people (personal content) from the server

Fig. 8: Personal content reconstruction on Phototourism Dataset: (a) Number of persons detected for server-accessible Radiance fields for FedNeRF and DecentNeRF per round for all users (b) Rendering server-accessible Radiance Fields for FedNeRF and DecentNeRF for training views. We note how DecentNeRF reduces the reconstruction of personal content on the server quantitatively and qualitatively.

global MLP renderings of DecentNeRF. Corresponding renderings shown in Fig 7 (b) validate that the server renderings do not reconstruct personal content faithfully for DecentNeRF.

7 Discussion

We present DecentNeRF, a decentralized framework for learning NeRFs, that we believe is crucial for enabling global-scale crowd-sourced NeRFs. DecentNeRF decomposes multi-view images into personal and global visual content. It then securely aggregates the global content across users with high visual fidelity while minimizing the reconstruction of personal content on the server.

While we analyze DecentNeRF on phototourism scenes, it would enable decentralized NeRF approaches for capturing large-scale 3D scenes and live events, facilitating widespread adoption of NeRFs. DecentNeRF's photorealism and decentralization advantages can be further enhanced by leveraging future advancements in NeRF architectures, training, and rendering schemes. DecentNeRF would pave the way for cross-collaborations between neural rendering and decentralized learning communities.

Limitations. While we do not demonstrate DecentNeRF on actual user mobile devices, there are recent advancements in mobile NeRF renderings [10] that indicate the promise for eventual mobile deployment. DecentNeRF assumes the server can't access the individual user radiance fields from the securely aggregated radiance fields. However, there can be attacks on SMPC [2] that attempt to break this protection, for which defenses [7] have been explored in differential privacy and federated learning literature. Please refer to the supplementary material on details of these attacks. Incorporating privacy guarantees and more efficient secure decentralization strategies would be exciting future directions.

References

1. Asadi, M., Zareinia, K., Saeedi, S.: Di-nerf: Distributed nerf for collaborative learning with unknown relative poses. arXiv preprint arXiv:2402.01485 (2024) 7
2. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning. In: International conference on artificial intelligence and statistics. pp. 2938–2948. PMLR (2020) 14, 3
3. Bell, J.H., Bonawitz, K.A., Gascón, A., Lepoint, T., Raykova, M.: Secure single-server aggregation with (poly) logarithmic overhead. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. pp. 1253–1269 (2020) 2
4. Beutel, D.J., Topal, T., Mathur, A., Qiu, X., Fernandez-Marques, J., Gao, Y., Sani, L., Li, K.H., Parcollet, T., de Gusmão, P.P.B., et al.: Flower: A friendly federated learning research framework. arXiv preprint arXiv:2007.14390 (2020) 12, 2
5. Bhowmick, A., Duchi, J., Freudiger, J., Kapoor, G., Rogers, R.: Protection against reconstruction and its applications in private federated learning. arXiv preprint arXiv:1812.00984 (2018) 3
6. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., Seth, K.: Practical secure aggregation for privacy-preserving machine learning. In: proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1175–1191 (2017) 4, 12, 2
7. Burkhart, M., Strasser, M., Many, D., Dimitropoulos, X.: {SEPIA}:{Privacy-Preserving} aggregation of {Multi-Domain} network events and statistics. In: 19th USENIX Security Symposium (USENIX Security 10) (2010) 2, 14
8. Cao, J., Wang, H., Chemerys, P., Shakhrai, V., Hu, J., Fu, Y., Makoviichuk, D., Tulyakov, S., Ren, J.: Real-time neural light field on mobile devices. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8328–8337 (2023) 3
9. Chen, X., Zhang, Q., Li, X., Chen, Y., Feng, Y., Wang, X., Wang, J.: Hallucinated neural radiance fields in the wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12943–12952 (2022) 3, 7
10. Chen, Z., Funkhouser, T., Hedman, P., Tagliasacchi, A.: Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In: The Conference on Computer Vision and Pattern Recognition (CVPR) (2023) 3, 14
11. Choi, B., Sohn, J.y., Han, D.J., Moon, J.: Communication-computation efficient secure aggregation for federated learning. arXiv preprint arXiv:2012.05433 (2020) 4
12. Dusmanu, M., Schonberger, J.L., Sinha, S.N., Pollefeys, M.: Privacy-preserving image features via adversarial affine subspace embeddings. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14267–14277 (2021) 7
13. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3. pp. 265–284. Springer (2006) 7
14. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 1440–1448 (2015) 11, 13
15. Holden, L., Dayoub, F., Harvey, D., Chin, T.J.: Federated neural radiance fields. arXiv preprint arXiv:2305.01163 (2023) 2, 3, 5, 6, 9, 10, 11, 12

16. Jeon, J., Lee, K., Oh, S., Ok, J., et al.: Gradient inversion with generative image prior. Advances in neural information processing systems **34**, 29898–29908 (2021) 4
17. 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. Foundations and Trends® in Machine Learning **14**(1–2), 1–210 (2021) 2, 3
18. Konečnỳ, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D.: Federated learning: Strategies for improving communication efficiency. arXiv preprint arXiv:1610.05492 (2016) 2, 3, 5, 8
19. kwea123: nerf_pl implementation of nerf (neural radiance fields) in pytorch lightning (2023), https://github.com/kwea123/nerf_pl/tree/nerfw, accessed: 2023-11-17 12, 2
20. Li, Z., Xian, W., Davis, A., Snavely, N.: Crowdsampling the plenoptic function. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16. pp. 178–196. Springer (2020) 3
21. Lindell, Y.: Secure multiparty computation (mpc). Cryptology ePrint Archive (2020) 4, 9
22. Martin-Brualla, R., Radwan, N., Sajjadi, M.S., Barron, J.T., Dosovitskiy, A., Duckworth, D.: Nerf in the wild: Neural radiance fields for unconstrained photo collections. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7210–7219 (2021) 2, 3, 7, 8, 9, 10, 11, 12
23. 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
24. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In: The European Conference on Computer Vision (ECCV) (2020) 5, 7, 10, 11, 2
25. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (ToG) **41**(4), 1–15 (2022) 11, 2
26. Ng, T., Kim, H.J., Lee, V.T., DeTone, D., Yang, T.Y., Shen, T., Ilg, E., Balntas, V., Mikolajczyk, K., Sweeney, C.: Ninjadesc: content-concealing visual descriptors via adversarial learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12797–12807 (2022) 11
27. Nguyen, J., Malik, K., Zhan, H., Yousefpour, A., Rabbat, M., Malek, M., Huba, D.: Federated learning with buffered asynchronous aggregation. In: International Conference on Artificial Intelligence and Statistics. pp. 3581–3607. PMLR (2022) 4
28. Photutorial: How many photos are there? (statistics & trends in 2023) (2023), https://photutorial.com/photos-statistics/ 2
29. Pittaluga, F., Koppal, S.J., Kang, S.B., Sinha, S.N.: Revealing scenes by inverting structure from motion reconstructions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 145–154 (2019) 11
30. Sanyal, S., Addepalli, S., Babu, R.V.: Towards data-free model stealing in a hard label setting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15284–15293 (2022) 3
31. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4104–4113 (2016) 7

32. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: Exploring photo collections in 3d. ACM Trans. Graph. **25**(3), 835–846 (jul 2006). https://doi.org/10.1145/1141911.1141964, https://doi.org/10.1145/1141911.1141964 3, 10, 11
33. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3d. In: ACM siggraph 2006 papers, pp. 835–846 (2006) 12
34. So, J., He, C., Yang, C.S., Li, S., Yu, Q., E Ali, R., Guler, B., Avestimehr, S.: Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning. Proceedings of Machine Learning and Systems **4**, 694–720 (2022) 4
35. Suzuki, T.: Federated learning for large-scale scene modeling with neural radiance fields. arXiv preprint arXiv:2309.06030 (2023) 2, 3
36. Tan, A.Z., Yu, H., Cui, L., Yang, Q.: Towards personalized federated learning. IEEE Transactions on Neural Networks and Learning Systems (2022) 3
37. Tasneem, Z., Milione, G., Tsai, Y.H., Yu, X., Veeraraghavan, A., Chandraker, M., Pittaluga, F.: Learning phase mask for privacy-preserving passive depth estimation. In: European Conference on Computer Vision. pp. 504–521. Springer (2022) 11
38. Truong, J.B., Maini, P., Walls, R.J., Papernot, N.: Data-free model extraction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4771–4780 (2021) 3
39. Weder, S., Garcia-Hernando, G., Monszpart, A., Pollefeys, M., Brostow, G.J., Firman, M., Vicente, S.: Removing objects from neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16528–16538 (2023) 8
40. Yang, Y., Zhang, S., Huang, Z., Zhang, Y., Tan, M.: Cross-ray neural radiance fields for novel-view synthesis from unconstrained image collections. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15901–15911 (2023) 3
41. Yin, H., Mallya, A., Vahdat, A., Alvarez, J.M., Kautz, J., Molchanov, P.: See through gradients: Image batch recovery via gradinversion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16337–16346 (2021) 4

Supplementary Material

DecentNeRFs: Decentralized Neural Radiance Fields from Crowdsourced Images

A Supplementary Video

In the supplementary material, we have included a video that visually illustrates the key features and results of our DecentNeRF approach, providing compelling evidence to support our claims. This video serves as a powerful complement to the written content, offering a more immersive and intuitive understanding of our work.

B Compute Comparisons

B.1 Complexity Comparison

Offline Communication and Computation For each round of aggregation, each user prepares offline coded random masks and distributes them to the users.
Reconstruction Decoding process to reconstruct the weighted average of the obfuscated weights sent by the users.
Online Communication Sending and receiving of obfuscated and receiving of weights and gradients.

In Table 3 we note that DecentNeRF has the same server complexity as FedAvg [23] at the cost of incurring more user complexity.

	FedNeRF	**DecentNeRF**
offline comm. (U)	$O(sN)$	$O(sN^2 + dN)$
offline comp. (U)	$O(dN + sN^2)$	$O(dN^2)$
online comm. (U)	$O(d + sN)$	$O(dN + sN^2)$
online comm. (S)	$O(dN + sN^2)$	$O(dN + sN^2)$
online comp. (U)	$O(d)$	$O(dN^2)$
reconstruction (S)	$O(dN^2)$	$O(dN^2)$

Table 3: Complexity comparison between FedAvg [23] and DecentNeRF. In terms of server computing, DecentNeRF has the same complexity but utilizes higher user computing. Here **N** is the total number of users, **d** is the model size, and **s** is the length of the secret keys as the seeds for Pseudo-Random Generator ($\mathbf{s} \ll \mathbf{d}$). U is for User and S is for Server

B.2 Server Compute

Here we show how server compute (in terms of FLOPs) is estimated for one of the example blender scenes in Table 1.

NeRF-W: For rendering one batch of rays (4096), SOTA NeRF algorithms such as the one by Müller et al. [25] take about $\approx 350\times10^6$ FLOPs (approximately 10^4 times faster than vanilla NeRF [24]). For the Lego dataset, which has 100 images of size 800×800 pixels, this leads to 12,500 batches for one epoch, resulting in a total cost of approximately 4.8×10^{12} FLOPs per epoch. Training the scene for 30 epochs requires around 130×10^{12} FLOPs of computing on the server. Here, we note that the compute cost for the server scales with the number of pixels in the complete dataset.

FedNeRF: It requires the addition of 20 (number of users) Global MLP weights per round, each MLP having around 1 million parameters. 1 addition is equivalent to 1 FLOP. For 30 federation rounds, the total computation cost at the server is approximately 0.6×10^9 FLOPs. In a decentralized setting, we note that the computing cost scales by number of Users.

DecentNeRF: In addition to the aggregation of user weights, similar to FedNeRF, the server computes the aggregate of gradients, which incurs the same computational cost as the aggregation of weights in FedNeRF, approximately 0.6×10^9 FLOPs. The computation of the dot product between each user's weights and the aggregated sum of gradients incurs an additional 1.2×10^9 FLOPs (accounting for parameter-wise multiplication and summation). For the reconstruction cost, we utilize Secure Aggregation (SecAgg+) [3], which has $O(dN \log N)$ complexity for the previous 4 computations, incurring an additional 3×10^9 FLOPs.

C Implementation Details

Training Details We use a PyTorch implementation of NeRF-W [19] for centralized evaluation, which is a variant of vanilla NeRF [24]. As shown in Figure 3, the Global Radiance Field **G**, consists of eight layers with 256 hidden units each, followed by an MLP that outputs RGB. This MLP comprises four layers of 128 hidden units each. The Personal MLP, consisting of four layers of 128 hidden units, outputs both σ and color. Details on positional encoding, encoding viewing direction, and other aspects are in line with NeRF-W and are mentioned in the supplementary material. For DecentNeRF, we implement the same architecture on the user device, which we simulate using the open-source Flower library [4]. This implementation includes the secure aggregation protocol of SecAgg+ [6], a feature of Flower. The FedNeRF implementation is akin to DecentNeRF but omits the Personal MLP, appearance embedding, secure aggregation, and learned weighted averaging.

During training, we sample 64 points for coarse and 64 for fine part of the network while for inference we sample 128 fine samples for all implementations. Models are trained with Adam for the same number of total iterations (sum of all user iterations for decentralized) for NeRF-W and DecentNeRF while FedNeRF is trained for lower rounds as the validation loss doesn't improve beyond certain rounds.

D Model Extraction Attacks and Defenses

There are model-extraction attacks that attempt to recover individual weights of the model reported by any client [30, 38] even if the weights were to be securely aggregated to compute the averaged weight. These attacks have been studied in the context of federated learning along with other attacks such as backdoor attacks [2] for adversarially poisoning a federated learning system. Similarly, there have been several defenses based on differential privacy, such as the PrivUnit mechanism [5] to prevent such attacks. The study of federated learning systems in the context of various such attacks is out of scope of this paper. The contributions of our paper hold independently to help enable a real-world vision application. That said, our method is in fact forward-compatible with the PrivUnit defense to prevent reconstruction of client weights.

E Dataset Details

Phototourism dataset Even though the scenes from the phototourism dataset vary in number of images we only use the first 200 images for all approaches since for our decentralized approaches we need to simulate multiple clients and a single machine. The real implementation of our decentralized wouldn't have this restriction and can learn on more than 200 images.

F DecentNeRF pseudocode

Algorithm 1 DecentNeRF algorithm. Data transfers are marked by *.

Require: Hyperparameters:
Weighted Averaging Learning Rate η,
Number of merge rounds M,
Number of training epochs per merge Γ.

$\{g^{(0)}, p^{(0)}\}$ are initial Shared and Public MLP weights
$\alpha^{(0)} \rightarrow \{\alpha_1, \alpha_2, \ldots, \alpha_K\}$ are initial client contributions
Distribute $\{g^{(0)}, p^{(0)}\}$ to all clients $k = 1, \ldots, K$ *
for merge round $m = 1$ to M **do**
  for client $k = 1$ to K in Parallel **do**
    $g^{(m-1)}$ is received from the server *
    if $m > 1$ **then**
      freeze $\{g_k^{(m-1)}, p_k^{(m-1)}\}$
      $\frac{\partial L_k}{\partial g_k^{(m-1)}} \leftarrow$ *Calc. Gradients*
      $\frac{\partial L_k}{\partial g_k^{(m-1)}}$ transmitted to the server *
      $\sum \frac{\partial L_k}{\partial \mathbf{g}^{(m-1)}}$ is recieved from the server *
      $\alpha_k^{(m)} \leftarrow$ Alpha Update$(\left\{ \frac{\partial L_k}{\partial g_k^{(m-1)}}, g_k^{(m-1)}, \eta \right\})$
    end if
    $\{g_k^{(m)}, p_k^{(m)}\} \leftarrow$ NeRF$(g_k^{(m-1)}, p_k^{(m-1)}, \Gamma)$
    $\alpha_k^{(m)} \times g_k^{(m)}$ is transmitted to the server *
  end for
  for server in Parallel **do**
    $g^{(m)} \leftarrow$ Secure Aggregation$(\{\alpha_k \times g_k\}_{k=1,\ldots,K}^{(m)})$
    $g^{(m)}$ is transmitted to the clients *
    $\sum \frac{\partial L_k}{\partial \mathbf{g}^{(m-1)}} \leftarrow$ Secure Aggregation$(\{\frac{\partial L_k}{\partial g_k^{(m-1)}}\}_{k=1,\ldots,K})$
    $\sum \frac{\partial L_k}{\partial \mathbf{g}^{(m-1)}}$ is transmitted to the clients *
  end for
end for
return $g^{(M)}$