Accelerating Heterogeneous Federated Learning with Closed-form Classifiers

Accelerating Heterogeneous Federated Learning with Closed-form Classifiers

Eros Fanì [1] **Raffaello Camoriano** [1 2] **Barbara Caputo** [1 3] **Marco Ciccone** [1]

Abstract

Federated Learning (FL) methods often struggle in highly statistically heterogeneous settings. Indeed, non-IID data distributions cause client drift and biased local solutions, particularly pronounced in the final classification layer, negatively impacting convergence speed and accuracy. To address this issue, we introduce *Federated Recursive Ridge Regression* (FED3R). Our method fits a Ridge Regression classifier computed in closed form leveraging pre-trained features. FED3R is immune to statistical heterogeneity and is invariant to the sampling order of the clients. Therefore, it proves particularly effective in cross-device scenarios. Furthermore, it is fast and efficient in terms of communication and computation costs, requiring up to two orders of magnitude fewer resources than the competitors. Finally, we propose to leverage the FED3R parameters as an initialization for a softmax classifier and subsequently fine-tune the model using any FL algorithm (FED3R with Fine-Tuning, FED3R+FT). Our findings also indicate that maintaining a fixed classifier aids in stabilizing the training and learning more discriminative features in cross-device settings. Official website: https://fed-3r.github.io/.

1. Introduction

Federated Learning (FL) (McMahan et al., 2017) provides a practical framework for training machine learning models collaboratively across distributed *clients* while ensuring privacy. This decentralized approach involves multiple communication rounds between clients and a central server. During each round, clients leverage their private data to improve their local models. Then, they send the model updates to the server, which aggregates them and transmits the improved model to the next set of clients for further improvement.

[1]Department of Computing and Control Engineering, Polytechnic University of Turin, Italy [2]Istituto Italiano di Tecnologia, Genoa, Italy [3]CINI Consortium, Rome, Italy. Correspondence to: Eros Fanì <eros.fani@polito.it>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

While appealing, limiting the optimization on the client side presents several challenges. In real-world scenarios, billions of clients might be involved (Kairouz et al., 2021), and data are often collected based on user preferences (Tan et al., 2022), availability (Gu et al., 2021), geographical location (Hsu et al., 2020; Fantauzzo et al., 2022), or personal habits (Fallah et al., 2020; Yang et al., 2018). This leads to data distributions across clients with inherent *statistical heterogeneity* in the form of *quantity skewness* (Li et al., 2020b; Wang et al., 2020; Hsu et al., 2020), *label skewness* (Karimireddy et al., 2020b; Li et al., 2022; Caldarola et al., 2022; Fanì et al., 2023), or *domain shift* (Fantauzzo et al., 2022; Nguyen et al., 2022; Liu et al., 2021).

As a result, training models that generalize well across the global underlying data distribution presents a major challenge. Specifically, convergence speed is hampered due to clients' sparse sampling and partial participation in successive rounds (Li et al., 2020b; Karimireddy et al., 2020a). Furthermore, the process of aggregating model updates becomes particularly challenging in strongly heterogeneous settings. This difficulty arises because biased local updates from individual clients can potentially steer the model away from global minimizers (Karimireddy et al., 2020b; Li et al., 2020a; Acar et al., 2021).

Most of the approaches addressing such issues focus on regularizing the local objective to reduce model parameters drift (Li et al., 2020a; Karimireddy et al., 2020b; Acar et al., 2021; Ozfatura et al., 2021) or leveraging momentum to incorporate knowledge from previous updates and align the local optimization to the global direction (Karimireddy et al., 2020a; Xu et al., 2021; Kim et al., 2022; Liu et al., 2023).

In particular, Luo et al. (2021) shows that model parameter drift in FL mainly involves neural network prediction heads. Indeed, deeper layers tend to be more susceptible to bias towards the individual client data distributions, while initial layers maintain better consistency in terms of representation similarity. As mainly studied in other areas such as Continual Learning (Ratcliff, 1990; McCloskey & Cohen, 1989), in classification this phenomenon occurs due to the inherent nature of the softmax classifier, which is prone to forgetting if updated by sampling data in a non-i.i.d. or class-imbalanced manner (Kirkpatrick et al., 2017), as the most recently acquired knowledge tends to be more relevant than the older one, resulting in *recency bias* (or *catastrophic forgetting*) (Mai et al., 2021; Masana et al., 2022; Wu et al., 2019; Lyu

et al., 2024). Similarly, in FL, the local optimization biases the classifier towards the local distribution, which can result in overwriting past clients' knowledge (Legate et al., 2023b; Caldarola et al., 2022). This problem is exacerbated in realistic *cross-device* scenarios with large number of devices, where clients may not be revisited during training (Ruan et al., 2021), making the optimization slower and unstable.

To address this issue, we propose *Federated Recursive Ridge Regression* (FED3R), a novel approach to FL leveraging pre-trained representations to train classifiers that are immune to statistical heterogeneity by design. The FED3R classifier can be efficiently trained and incrementally updated in closed form by repurposing the Ridge Regression (RR) online formulation for FL. Each client computes its local RR statistics using the feature maps generated by the pre-trained feature extractor and sends them to the server, where they are aggregated and used to compute the RR classifier. The FED3R solution is equivalent to the centralized RR solution and invariant to the sampling order of the clients. Our strategy allows aggregating client models exactly, efficiently, and without the need for backpropagation. Moreover, each client necessitates only a single round of communication with the server, contrary to traditional gradient-based methods.

Furthermore, to address non-linearities of the latent space and the distribution shift of the target task from the pre-trained representation, we present FED3R with Random Features (FED3R-RF), a kernelized version of FED3R, and FED3R with Fine-Tuning (FED3R+FT), which can update both feature extractor and the classifier jointly.

Finally, we repurpose RR as a tool to quantitatively assess the quality of the feature extractors. Indeed, learning an RR classifier on the features of a trained feature extractor allows decoupling the contributions of the feature extractor and the classifier on the final performance. With this, we find that fine-tuning only the feature extractor while keeping the FED3R classifier fixed not only helps to counteract client drift and destructive interference during the aggregation phase but also improves the quality of the features in settings with strong statistical heterogeneity.

Contributions

- We propose FED3R, a federated version of Ridge Regression, to efficiently learn a linear classifier that is immune to statistical heterogeneity. We also propose FED3R-RF, a kernelized version of the algorithm based on random features to handle the non-linearities of the input space.
- We demonstrate that FED3R significantly accelerates training, converging faster than FL other methods. Additionally, we show that FED3R reduces communication and computational costs by up to two orders of magnitude compared to other methods.
- We show that FED3R can also be employed as a classifier initialization for fine-tuning representations and how it can stabilize the training in highly heterogeneous settings. We evaluate the effectiveness of our proposed algorithms on the Landmarks and iNaturalist datasets (Hsu et al., 2020), two realistic and cross-device FL scenarios for visual classification with thousands of clients and classes.
- We show how to repurpose RR as a tool to discern the contributions of the feature extractor and the classifier to the final model performance. We find that fine-tuning the feature extractor while keeping the FED3R classifier fixed greatly improves the features' quality, robustness to client drift, and destructive interference.

2. Related Works

Statistical heterogeneity in FL. Despite the effectiveness of current FL approaches in homogeneous scenarios with i.i.d. data, addressing statistically heterogeneous and realistic settings remains challenging (Kairouz et al., 2021). Private data exhibits biases due to factors such as personal habits and geographical locations (Hsu et al., 2020; Kairouz et al., 2021; Fallah et al., 2020), causing variations among clients in categories, domains, and dataset sizes. This bias induces *client drift* (Karimireddy et al., 2020b), leading local models to converge toward different minima, deviating from the global direction, resulting in noisy, unstable learning trends (Li et al., 2020b; Caldarola et al., 2022).

Optimization-based methods for heterogeneous FL. To reduce the impact of heterogeneity, simple solutions involve limiting the drift of the local models with regularization techniques. For instance, FedProx (Li et al., 2020a) introduces a penalization in the local objectives to prevent divergence from the global model. Other methods, such as Scaffold (Karimireddy et al., 2020b), leverage stochastic variance reduction (Reddi et al., 2016) to correct the local direction with the global one. FedDyn (Acar et al., 2021) aligns local and global stationary points at convergence, enjoying the same convergence properties of Scaffold. Still, its practical effectiveness in cross-device settings is limited since it is often prone to parameter explosion (Varno et al., 2022).

Other approaches aim to reduce client drift by exploiting the history of previous updates, incorporated with momentum or adaptive optimizers (Wang et al., 2019; Reddi et al., 2021). In particular, FedAvgM (Hsu et al., 2019) employs momentum in the server-side aggregation, demonstrating effectiveness in realistic settings (Hsu et al., 2020). Other works introduce client-side momentum (Kim et al., 2022; Xu et al., 2021; Karimireddy et al., 2020a) to guide the local updates in the direction followed by the global model. Finally, Mime (Karimireddy et al., 2020a) aims at replicating the behavior of models trained on i.i.d. data by combining stochastic variance reduction and client-side momentum.

Despite these methods being theoretically principled, our

work empirically reveals their inherent instabilities in real-world cross-device FL scenarios. In contrast, we demonstrate that training classifiers with closed-form solutions and exact aggregation can be dramatically faster and communication efficient than gradient-based optimization in practical cross-device scenarios. Additionally, we illustrate that integrating our method with FL optimizers can further expedite convergence through a final fine-tuning stage.

Classifier bias and destructive interference. A natural direction to study the effect of heterogeneity is to analyze its impact on different parts of the model. On this matter, Luo et al. (2021) showed that the bias of the clients towards local data distributions is significantly more pronounced in the deeper layers, with a peak in the last one, *i.e.*, the prediction head. This phenomenon has also been observed in Continual Learning on sequences of heterogeneous task distributions (Ramasesh et al., 2020; Davari et al., 2022; Kim & Han, 2023). Other works observed that biased classifiers and misaligned features create a vicious cycle (Zhou et al., 2022; Li et al., 2023). On one side, discriminative features are needed to train models effectively, as convergence speed improves when gradients are more aligned (Nguyen et al., 2023). However, heterogeneity heavily affects the prediction head, which suffers from destructive interference during the aggregation phase and hampers the features learning process. Indeed, Yu et al. (2022) shows that learning the feature extractor and the classifier in two separate phases may be beneficial in FL, as also observed in other fields (Kang et al., 2020; Wang et al., 2022).

Previous research attempted to mitigate classifier biases at the end of FL training by retraining only the classifier on the server with generated virtual features (Luo et al., 2021; Shang et al., 2022; Nguyen et al., 2023). However, this approach remains sub-optimal as it is based on the quality of the feature representation and generative process, which could negatively affect the retrained classifiers. More recently, Li et al. (2023) proposed a fixed synthetic classifier, motivated by the simplex geometry of the logits space induced by the neural collapse (Papyan et al., 2020).

In this work, we take a different turn and tackle the classifier bias problem with a principled approach. Starting from a pre-trained representation, we employ a *one-vs-rest* Ridge Regression (RR) classifier that can be trained in a distributed setting with exact aggregation. This efficiency stems from an online formulation that recovers the closed-form solution of the centralized problem by effectively providing a classifier that does not suffer from statistical heterogeneity and is invariant to the actual federated split. Legate et al. (2023a) adopts a similar rationale for a Nearest Class Mean (NCM) classifier, avoiding gradient updates using class centroids. While NCM may be effective on simpler datasets, we demonstrate its weakness in realistic scenarios, contrasting the consistent performance of RR.

We refer to Appendix A for additional related works on existing RR-based methods in distributed learning and vertical FL and a broader overview of the existing literature on transfer learning methods with pre-trained models in FL.

3. Background

In this section, we provide a concise overview of the FL framework and the fundamental concepts of Ridge Regression before formally describing our algorithm.

3.1. FL Problem Formulation

Let $\mathcal{K}$ be the set of all the clients involved in the training with cardinality $|\mathcal{K}| = K$, and let $\mathcal{S}$ be the server that orchestrates the training procedure. Each client $k \in \mathcal{K}$ has access to a private local dataset $\mathcal{D}_k$ of size $n_k = |\mathcal{D}_k|$; neither the server nor the other clients can access $\mathcal{D}_k$. Each local dataset $\mathcal{D}_k$ is composed of n_k pairs $(x, y) \sim P_k$, where $x \in \mathcal{X}, y \in \mathcal{Y}$. Here, $\mathcal{X}$ and $\mathcal{Y}$ represent the input and output spaces, and P_k is the joint data distribution associated with client k.

The global federated objective is given by:

$$\theta^* = \arg\min_{\theta} \sum_{k \in \mathcal{K}} \mathcal{L}_k(\mathcal{M}; \mathcal{D}_k), \tag{1}$$

where $\mathcal{L}_k = \sum_{(x,y)\in\mathcal{D}_k} \ell(\mathcal{M}(x;\theta), y)$ is the Local Empirical Risk associated to the client k, computed according to a loss function ℓ (e.g., cross-entropy), and $\mathcal{M}$ is a model parameterized by θ. At each round t, a subset of selected clients $\mathcal{K}' \subseteq \mathcal{K}$ receives the global parameters θ^{t-1} of the previous round from the server, initializes the local parameters $\theta_k = \theta^{t-1}$ and optimizes them using the private datasets $\mathcal{D}_k$, obtaining the new parameters θ_k^t. Then, the locally optimized model parameters θ_k^t are shared with the server $\mathcal{S}$, which aggregates them according to the specific FL algorithm. For instance, the FedAvg (McMahan et al., 2017) aggregation rule is a weighted average of clients' models $\theta^t = \sum_{k\in\mathcal{K}'} \frac{n_k}{n} \theta_k^t$, where $n = \sum_{k\in\mathcal{K}'} n_k$. The server broadcasts the aggregated model θ^t to the new active clients. The process is repeated for several rounds until convergence.

3.2. Closed-form Ridge Regression (RR)

Our work is based on the idea of using one-vs-rest classifiers such as *least-squares regressors* (Stigler, 1981; Björck, 1996; Rifkin et al., 2003) that admit a closed-form solution and can be computed efficiently. We first define the problem for the *centralized setting*, where samples from a dataset $\mathcal{D}$ can be accessed simultaneously.

Although simple least-squares empirical risk minimization is generally prone to overfitting (Bishop, 2006), it can easily be augmented with $\mathcal{L}_2$ regularization (controlled by a Tikhonov hyper-parameter $\lambda \in \mathbb{R}^+$), obtaining a Ridge Regression (Boyd & Vandenberghe, 2004) problem:

$$W^* = \arg\min_{W \in \mathbb{R}^{p\times C}} \|Y - XW\|^2 + \lambda \|W\|^2, \tag{2}$$

where $X \in \mathbb{R}^{n \times p}$ is the matrix of the n stacked input samples, $Y \in \mathbb{R}^{n \times C}$ is the matrix of the stacked one-hot-encoding vectors of the corresponding C classes, and p is the input dimensionality. The solution W^* constitutes the optimal parameters for the linear predictor $f(x; W) = W^\top x$.

The problem in Eq. (2) admits a closed-form solution:

$$W^* = (X^\top X + \lambda I_p)^{-1} X^\top Y, \tag{3}$$

where I_p is the $p \times p$ identity matrix.

Since $X^\top X + \lambda I_p \succ 0$ for any $\lambda > 0$, no additional assumptions on the rank or the dimensions of the matrix X are required to prove that the optimal solution W^* exists (Boyd & Vandenberghe, 2004). Moreover, RR can be directly applied to classification, as introduced in Rifkin et al. (2003), and it converges in probability to the optimal Bayes classifier as n tends to infinity (Steinwart & Christmann, 2008; Bartlett et al., 2006; Shawe-Taylor & Cristianini, 2004).

3.3. Handling Non-linear Input Spaces in RR

While simple and powerful, RR is a linear classifier whose performance is tied to the separability of the input space. To handle the non-linearities of the input space, we map $\mathcal{X}$ onto a latent feature space $\mathcal{Z} \subseteq \mathbb{R}^d$ using a pre-trained feature extractor $\varphi : \mathcal{X} \to \mathcal{Z}$ and apply Eq. (3) directly on the feature maps to obtain the optimal predictor.

For clarity, we express Eq. (3) for a linear classifier $W^* \in \mathbb{R}^{d \times C}$ whose input space is $\mathcal{Z}$, as:

$$W^* = (A + \lambda I_d)^{-1} b \tag{4}$$

Here, $A := Z^\top Z = \sum_{(x,y)\in\mathcal{D}} \varphi(x)\varphi(x)^\top \in \mathbb{R}^{d \times d}$ is the covariance matrix of samples in the mapped space, where $Z \in \mathbb{R}^{n \times d}$ is the matrix of mapped input samples with each row $Z_i = \varphi(X_i)$. Also, $b := Z^\top Y = \sum_{(x,y)\in\mathcal{D}} \varphi(x) e_y^\top \in \mathbb{R}^{d \times C}$, and e_y is the one-hot encoding vector for class y.

Although a pre-trained feature extractor φ can be used for handling non-linearities in the input space, the performance depends on the quality of φ on the target task. To further improve the latent space's separability, we also consider employing Kernel Ridge Regression (KRR) in our method. KRR is a nonparametric learning algorithm that uses kernel functions to implicitly address the non-linearity of the input space (Hastie et al., 2009; Shawe-Taylor & Cristianini, 2004). However, the kernel matrix's space complexity is $\mathcal{O}(n^2)$, in contrast to the covariance matrix A with space complexity of $\mathcal{O}(d^2)$. For sizable datasets ($n \gg d$), storing the kernel matrix and computing the exact KRR solution becomes impractical. To overcome this bottleneck, we employ Random Features KRR (Rahimi & Recht, 2007), a data-independent subsampling scheme enabling optimal generalization properties while reducing the computational complexity of KRR (Rudi & Rosasco, 2017) through an approximate nonlinear mapping of the input features. Its properties are particularly suitable for the FL setting, enabling us to keep the same formulation as our algorithm's linear version, as shown in the next section.

4. Method

We now present Federated Recursive Ridge Regression (FED3R), and show how recursive least squares can be elegantly repurposed to the FL setting. Each client contributes to the A and b matrices of Eq. (4) by independently computing local statistics, which are then collected and aggregated by the server and used to compute the global RR classifier in closed form. Moreover, we introduce FED3R-RF, a kernelized version of our algorithm that uses random features to approximate the KRR solution.

4.1. Federated Recursive Ridge Regression (FED3R)

While the least-squares problem can be effectively solved in a closed form via Eq. (3), in principle it needs access to the entire dataset $\mathcal{D}$, which is not always available if data is accessed sequentially (Camoriano et al., 2017; Wang et al., 2022) or is distributed across devices, as in FL.

Luckily, thanks to the linearity of Eqs. (3) and (4), when new samples become available, the optimal solution can be exactly updated by recursive least squares (Kailath et al., 2000). This method computes solutions recursively and efficiently via Sherman-Morrison-Woodbury or Cholesky updates (Sherman & Morrison, 1950; Hager, 1989).

Alternatively, the RR statistics A and b can be cumulatively updated without re-computing them from scratch on the entire dataset. The solution can then be computed using the updated statistics by solving a linear system. Our method is based on the observation that the matrices A and b can be incrementally computed, for instance, by simply summing over the samples of the dataset $\mathcal{D}$. Exact and efficient incremental RR updates enable several continual and incremental classification methods (Camoriano et al., 2017; Wang et al., 2022), as RR admits an equivalent exact incremental solution (Björck, 1996; Sayed, 2008) that we reformulate specifically for the FL context, finally leading to our FED3R algorithm.

In practice, thanks to the associative property of the sum, we can break the matrices A and b into the contributions of the clients' local datasets $\mathcal{D}_k$:

$$\begin{aligned} A &= \sum_{(x,y)\in\mathcal{D}} \varphi(x)\varphi(x)^\top = \sum_{k\in\mathcal{K}} \sum_{(x,y)\in\mathcal{D}_k} \varphi(x)\varphi(x)^\top \\ &= \sum_{k\in\mathcal{K}} Z_k^\top Z_k = \sum_{k\in\mathcal{K}} A_k, \end{aligned} \tag{5}$$

$$\begin{aligned} b &= \sum_{(x,y)\in\mathcal{D}} \varphi(x) e_y^\top = \sum_{k\in\mathcal{K}} \sum_{(x,y)\in\mathcal{D}_k} \varphi(x) e_y^\top \\ &= \sum_{k\in\mathcal{K}} Z_k^\top Y_k = \sum_{k\in\mathcal{K}} b_k. \end{aligned} \tag{6}$$

Algorithm 1 - **FED3R** and **FED3R-RF**

Require:
Server $\mathcal{S}$, clients $\mathcal{K}$
Fixed pre-trained feature extractor $\varphi : \mathcal{X} \to \mathbb{R}^d$
Random features $\omega \in \mathbb{R}^{d \times D}$
Hyper-parameter $\lambda > 0$
Clients $k \in \mathcal{K}$:
for each client $k \in \mathcal{K}$ in parallel **do**
$Z_k = \varphi(X_k)$
Map Z_k to a D-dimensional space using the RF ω
$A_k = Z_k^\top Z_k, \quad b_k = Z_k^\top Y_k$
end for
Server $\mathcal{S}$:
Collect all the clients' statistics
Compute $A = \sum_{k \in \mathcal{K}} A_k, \;\; b = \sum_{k \in \mathcal{K}} b_k$
Apply Eq. (4) to get W^*
Normalize W^*: $W_c^* \leftarrow W_c^* / \|W_c^*\| \quad \forall c \in [C]$

This is true for all the possible partitions of the underlying dataset $\mathcal{D}$ such that $\mathcal{D} = \bigcup_{k \in \mathcal{K}} \mathcal{D}_k$ is the union of the local datasets $\mathcal{D}_k$, with $\mathcal{K}$ the set of all the clients.

In FED3R, each client k computes its local A_k and b_k statistics and shares them with the server, where they are aggregated and employed to calculate W^*. Hence, the FED3R solution is mathematically equivalent to the centralized RR solution, independently of the federated split. Therefore, it inherits all the generalization properties of RR. In particular, it achieves optimal convergence rates in probability (Caponnetto & De Vito, 2007).

Finally, to address possible class unbalanced distributions over $\mathcal{D}$, we normalize W^* by dividing each column by its class norm, similar to the approach used by the authors of (Legate et al., 2023a): $W_c^* \leftarrow W_c^* / \|W_c^*\|$.

4.2. FED3R with Random Features (FED3R-RF)

As pre-trained feature extractors may not be expressive enough to separate features for complex learning problems linearly, we also introduce FED3R-RF, which first performs a nonlinear random features mapping of the latent feature space to a new D-dimensional feature space by approximating the corresponding kernel feature map, where $D > d$ is a hyper-parameter. Consequently, all the dimensionalities of the statistics that depended on d here depend on D. FED3R and FED3R-RF are summarized in Algorithm 1.

4.3. FED3R and FED3R-RF Properties

The FED3R (FED3R-RF) solution computed using all the local datasets $\mathcal{D}_k$ is mathematically equivalent to the corresponding centralized RR (RR with Random Features) solution using $\mathcal{D}$. Consequently, the federated classifiers inherit all the properties and guarantees of the centralized ones. Additionally, both methods exhibit three fundamental and desirable properties related to the FL setting, which we list below. Finally, for a discussion on the privacy guarantees of our method, we refer the reader to Appendix B.

Immunity to statistical heterogeneity. As Eqs. (5) and (6) show, due to the associative and commutative properties of the sum, once all the clients have shared their statistics with the server, the matrices A and b are the same for all possible partitions of the dataset. Hence, the FED3R solution is invariant to the particular data split across the clients; in other words, FED3R is immune to statistical heterogeneity and is invariant to the clients' sampling order. Consequently, FED3R guarantees the same final solution given any FL split of the same dataset $\mathcal{D}$.

Clients are sampled only once. In FED3R, each client only needs to communicate its statistics once, meaning it only needs to be sampled once. If we assume that, as for classical FL algorithms, κ clients are sampled during each round without replacement, FED3R requires exactly $\lceil K/\kappa \rceil$ rounds to converge to its final optimal solution, and no asymptotic convergence proof is required. The convergence is exact and guaranteed after $\lceil K/\kappa \rceil$ rounds. Therefore, the higher the participation rate, the faster FED3R converges. This is not generally guaranteed for gradient-based FL algorithms.

Differences with gradient-based FL algorithms. Unlike gradient-based FL algorithms, FED3R does not rely on common assumptions such as the smoothness of clients' objectives or the unbiasedness and bounded variance of stochastic gradients (Kairouz et al., 2021; Karimireddy et al., 2020a;b; Acar et al., 2021). In addition, FED3R does not require assuming bounded gradient dissimilarity among clients, which formalizes the effect of heterogeneous local datasets.

4.4. FED3R with Fine-Tuning (FED3R+FT)

The proposed FED3R algorithm is a fast and efficient solution to learn a classifier with guarantees of being immune to statistical heterogeneity. However, FED3R performance relies on the quality of the pre-trained feature extractor, which is frozen. Similar to the approaches from Wang et al. (2022) and Legate et al. (2023a), we propose FED3R with Fine-Tuning (FED3R+FT), a two-stage algorithm where a fine-tuning stage follows the classifier initialization.

First, FED3R+FT learns a FED3R classifier using a pre-trained feature extractor. Then, it initializes a softmax classifier using the parameters of the FED3R classifier. Finally, the whole model is fine-tuned using a traditional FL algorithm. As the FED3R classifier is the optimal Regularized Least Squares classifier obtained using the pre-trained feature extractor, it provides a stable starting point that can mitigate client drift and destructive interference during aggregation.

However, due to the FED3R classifier's derivation from the mean squared loss, the entropy of its predictions distribution

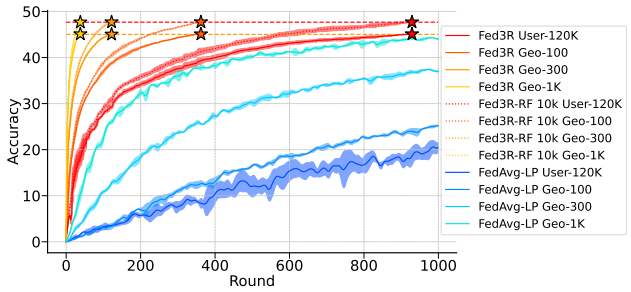


Figure 1: FED3R and FED3R-RF invariance to different iNaturalist splits. All the curves converge to the same values, showing how both methods are immune to statistical heterogeneity.

may not directly correspond to that of the cross-entropy (CE) loss employed in the fine-tuning phase. Consequently, the shape of the two loss landscapes can significantly vary. To solve this issue, we calibrate the entropy of the FED3R initialization by adjusting the temperature of the softmax function (see more details in Appendix C).

We propose three different fine-tuning strategies for FED3R+FT. The first involves fine-tuning the entire model, which we actually refer to as FED3R+FT[1]. In some cases, the pre-trained features may already be robust enough, so it is reasonable only to fine-tune the classifier. We call this variant FED3R+FTLP. On the other hand, keeping the FED3R classifier constant while fine-tuning the feature extractor could help minimize destructive interference, especially in cross-device scenarios with high statistical heterogeneity, where the classifier is often the most affected layer (Luo et al., 2021; Li et al., 2023). We refer to this last variant as FED3R+FTFEAT.

5. Experiments

In this section, we empirically evaluate the performances of our proposed methods in terms of accuracy, convergence speed, communication, and computational costs. First, we empirically show that FED3R and FED3R-RF are both immune to statistical heterogeneity and their performances are equivalent to the ones of the corresponding centralized RR solutions. Then, we compare FED3R and FED3R-RF with FedAvg (McMahan et al., 2017), FedAvgM (Hsu et al., 2019), and Scaffold (Karimireddy et al., 2020b). Finally, we show how FED3R can effectively bootstrap training and use it as powerful initialization when combined with other optimization methods using FED3R+FT.

Datasets. For the evaluation we choose two large-scale image classification datasets, Landmarks (Weyand et al., 2020) and iNaturalist (Van Horn et al., 2018), both partitioned as proposed in (Hsu et al., 2020) [2]. These datasets emulate realistic FL scenarios, as they offer over 10^5 training images and involve thousands of heterogeneous clients (see Table 4 in Appendix C for additional details). We select 10 clients per round in all our experiments, except when differently declared, simulating a participation rate of $\approx 0.8\%$ for Landmarks and $\approx 0.1\%$ for iNaturalist.

Models and baselines. All the experiments are conducted using a MobileNetV2 architecture (Sandler et al., 2018) pre-trained on ImageNet-1k (Deng et al., 2009). As baselines we included FedAvg (McMahan et al., 2017), FedAvgM (Hsu et al., 2019) and Scaffold (Karimireddy et al., 2020b). We do not include FedDyn (Acar et al., 2021), Mime, and MimeLite (Karimireddy et al., 2020a) because they fail to converge in most of the FED3R+FT setting. Moreover, we do not include Scaffold in all the iNaturalist experiments, as it fails to converge. We refer to Appendix C for additional implementation details.

Additional details. Appendices D and E offer supplementary information regarding the estimation of communication costs and computation costs, respectively. Further exploration into the efficacy of random feature approximation and the performance of our methods on the small-scale Cifar100 dataset are presented in Appendix F and Appendix G, respectively. Appendix H provides additional plots that compare the best methods for Landmarks and iNaturalist.

5.1. FED3R Equivalence to (Centralized) RR

In this section, we evaluate the results of FED3R and FED3R-RF on several splits of the iNaturalist dataset, simulating various levels of statistical heterogeneity as proposed in Hsu et al. (2020). Both algorithms rely only on the pre-trained feature extractor to train the classifiers and do not adjust the representation. Hence, to ensure a fair comparison, we compare them with the *Linear Probing* version (LP) of the FedAvg baseline, where we keep the parameters of the feature extractor frozen and train only the softmax classifier. For these experiments, we choose FedAvg as the baseline to compare with because it shows similar performances to FedAvgM and Scaffold fails to converge.

Specifically, Figure 1 compares FED3R, FED3R-RF with 10k random features, and FedAvg-LP, using four different iNaturalist splits (for details on the splits, refer to Table 4 in Appendix C). All the FED3R and FED3R-RF 10k experiments converge to 45.1% and 47.6% accuracy, respectively, which are equivalent to RR and RR-RF with 10k random features in the centralized scenario (the dashed lines). This confirms the invariance to statistical heterogeneity and the equivalence of the FL and centralized solution, as discussed

[1]We sometimes use FED3R+FT FEAT+LP for convenience, meaning that we fine-tune both feature extractor and classifier.

[2]For Landmarks and iNaturalist, we always mean the Landmark-Users-160K and iNaturalist-Users-120K partition, respectively, except when differently declared.

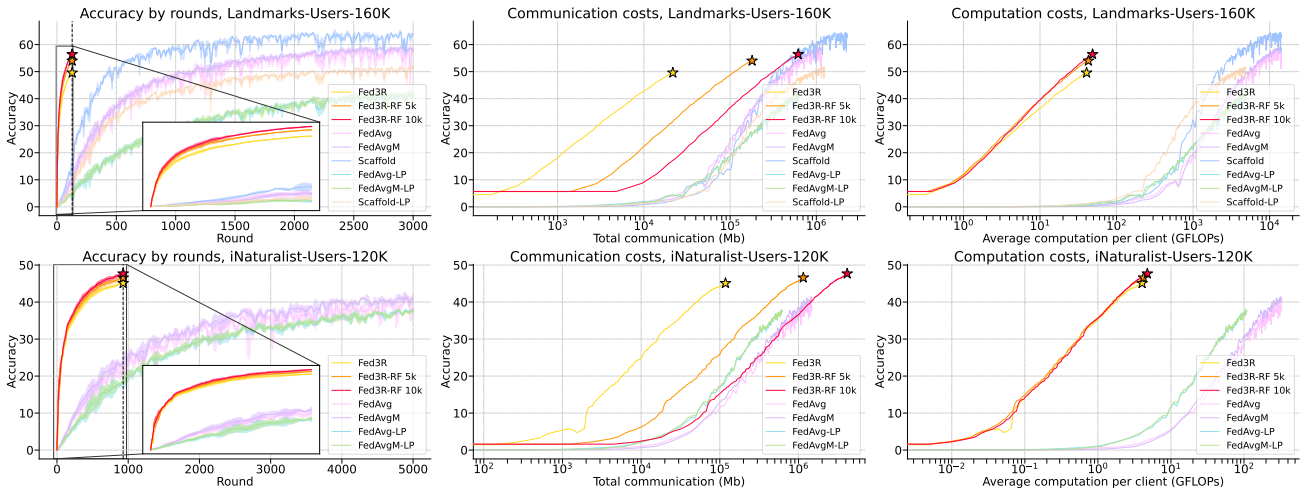


Figure 2: Comparison between FED3R and the baselines. From left to right: accuracy vs rounds, accuracy vs communication budget, accuracy vs average computation per client. Top row: Landmarks results, Bottom row: iNaturalist results. Fed3R shows clear advantages regarding convergence speed, communication, and computation budget required.

in Section 4.3 from a theoretical perspective. Finally, as noticeable, convergence is much faster than FedAvg-LP, as its speed is proportional to the number of clients in the specific split, as discussed in Section 4.3.

5.2. FED3R vs. Gradient-based FL Baselines

Figure 2 shows how the methods perform in terms of *accuracy* and *convergence speed* (left), *communication costs* to reach the target accuracy (center), and *average computation* needed per client to reach the target accuracy (right). This is shown for both the Landmarks dataset (first row) and the iNaturalist dataset (second row). The communication and computation costs can be interpreted as the budget needed for an FL system to reach a specific accuracy using the respective method.

As shown in Figure 2, FED3R outperforms all the LP baselines in terms of speed and computational efficiency. Notably, on the Landmarks dataset, FED3R achieves comparable results to Scaffold-LP – the best of the LP baselines – while requiring two orders of magnitude less communication and computations, with FED3R-RF even surpassing it at the expense of communication cost, but still being more computationally efficient.

Remarkably, FED3R exhibits even greater efficacy on the iNaturalist dataset, surpassing all LP and full-training baselines by a substantial margin across all evaluation criteria, including rounds, total communication, and computation costs. This underscores the significant impact of exact aggregation classifiers and the challenges faced by optimization-based FL methods in heterogeneous cross-device settings.

Discussion on the convergence speed. Both FED3R and FED3R-RF (with $D = 5k$ and $D = 10k$) are much faster than the baselines and require up to two orders of magnitude less communication and average computation budget. For instance, FED3R-RF with $D = 10k$ on the Landmarks dataset hits 40% accuracy after 27.3 rounds on average, compared to the 528.7 (speedup $\times$19.3) needed by FedAvg, 285.7 needed by Scaffold (speedup $\times$10.5), and 2251.3 (speedup $\times$82.4) and 690.33 (speedup $\times$25.3) needed by their corresponding LP versions. Similar consistent speedups can be observed for higher values of accuracy and on iNaturalist.

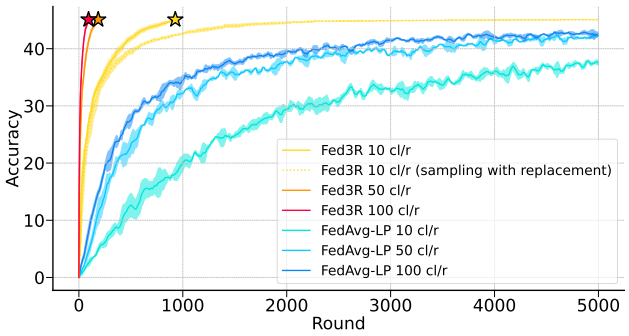


Figure 3: Accuracy vs Rounds with three different participation rates (indicated in the legend by x cl/r, where cl/r stands for *sampled clients per round*) and two sampling strategies (*without replacement* for FED3R and *with replacement* for FedAvg-LP, if not differently specified), iNaturalist dataset.

Indeed, since each client needs to be sampled only once, the convergence speed of FED3R depends only on the total number of clients and the participation rate. With 10 clients sampled per round and a total of 1262 clients for Landmarks and 9275 clients for iNaturalist, FED3R always needs exactly 127 and 928 rounds to converge, respectively, which

Table 1: Final accuracy (%) achieved by the FED3R family of classifiers and by the FedNCM classifier.

	FED3R	FED3R-RF 5k	FED3R-RF 10k	FEDNCM
Landmarks	49.6 ± 0.0	53.9 ± 0.0	**56.6 ± 0.0**	36.2 ± 0.0
iNaturalist	45.1 ± 0.0	46.8 ± 0.0	**47.6 ± 0.0**	32.2 ± 0.0

is an important advantage over gradient-based optimization methods as they require multiple passes over the data.

Performance with different sampling rates. Figure 3 shows how FED3R final performance is invariant to the number of clients sampled at each round by construction (as explained in Section 4.3). As a worst-case analysis, we also show that even sampling *with replacement*, as in FedAvg and the other classical algorithms, proves to be faster than the LP methods. Notably, FED3R, with a sampling rate of 10 clients per round, converges faster than FedAvg-LP with a sampling rate of 100 clients per round. Indeed, FED3R almost achieves convergence performance after just 1.5k rounds. Therefore, FED3R does not really need to wait for all the clients in the federation to be available. For further investigation on how many rounds are needed to sample with replacement a given percentage of distinct clients, we refer the reader to the Appendix I.

Ablation on Fed3R vs. FedNCM. Similarly to FED3R, Legate et al. (2023a) propose fitting a closed-form classifier using Nearest Class Means (FEDNCM). Table 1 compares the performance of FEDNCM, FED3R, and FED3R-RF at convergence for both the Landmarks and iNaturalist datasets, without the fine-tuning stage. FED3R clearly emerges as a more powerful and robust approach that can deal with complex datasets, outperforming FedNCM by a significant margin - up to 20 accuracy points with the kernelized version. Consequently, as our method yields superior classifiers, we omit the FEDNCM experiments from the FT discussions in Section 5.3. This decision is based on the assumption that employing a weaker classifier as initialization would result in a lower final accuracy.

5.3. FED3R+FT experiments

In the previous section, we showed the efficacy of the FED3R algorithm in terms of speed and efficiency. However, sometimes its performance may not surpass baseline methods, particularly when utilizing FED3R instead of FED3R-RF, as shown in Figure 2, top row. However, while employing FED3R-RF incurs a minimal computational overhead, it substantially escalates communication costs, rivaling other baseline methods. Moreover, both FED3R and FED3R-RF rely only on pre-trained features, as the pre-trained feature extractor is not optimized on the target datasets of the clients. Therefore, we propose also FED3R+FT and its variants, as discussed in Section 4.4.

Table 2: FED3R+FT final performance (Acc. %).

Dataset	FT alg.	Classifier Initialization	FTFEAT	FTLP	FT
Landmark-Users-160K	FedAvg	✗	-	41.0 ± 1.6	57.7 ± 1.2
		FED3R	59.6 ± 0.2	56.7 ± 0.4	**64.1 ± 0.4**
	FedAvgM	✗	-	40.8 ± 1.2	58.7 ± 0.8
		FED3R	59.0 ± 0.2	56.2 ± 0.5	**64.1 ± 0.2**
	Scaffold	✗	-	51.7 ± 0.3	63.4 ± 0.9
		FED3R	63.4 ± 0.1	58.0 ± 1.5	**67.4 ± 0.3**
iNaturalist-Users-120K	FedAvg	✗	-	36.7 ± 0.4	39.5 ± 3.2
		FED3R	**50.8 ± 0.2**	42.0 ± 0.3	49.0 ± 0.6
	FedAvgM	✗	-	37.6 ± 0.2	39.3 ± 0.7
		FED3R	**51.5 ± 0.2**	43.5 ± 1.9	49.8 ± 0.8

Table 2 shows the final accuracy values for the different strategies. Moreover, Figure 4 shows the performance and the costs of FED3R+FT and baseline methods for the Landmarks dataset, while Figure 5 compares the three variants of FED3R+FT for the iNaturalist dataset. FedAvgM serves as the FT algorithm in both scenarios. Notably, at least one of our FED3R+FT variants significantly outperforms the baselines for both datasets, achieving accuracies of 67.4 ± 0.3 and 51.5 ± 0.2 in the Landmarks and iNaturalist experiments, respectively, and the curves associated with our methods are consistently and significantly above the comparisons across every x-axis value.

Fine-tuning the entire model shows benefits on Landmarks, which is more similar to cross-silo FL than iNaturalist. On the other hand, in federated settings with more clients, such as in iNaturalist, there is a significant negative impact during the aggregation phase for the FED3R+FT and FED3R+FTLP experiments, as the classifier is fine-tuned and becomes susceptible to the classifier bias phenomenon, as discussed in Section 1. Conversely, keeping the classifier fixed and only fine-tuning the feature extractor as in the FED3R+FTFEAT experiments eliminates this phenomenon for the classifier, ensuring performance improvement and clearly indicating that the pre-trained features were not sufficiently good for the target task.

5.4. Features Quality Evaluation via Ridge Regression

In this section, we show that RR is a useful tool for quantifying features' quality and linear separability in an FL scenario. Indeed, RR provides a closed-form deterministic solution that solely depends on the feature space and that can be computed in federated settings through the FED3R equivalent formulation, unaffected by statistical heterogeneity. Moreover, RR is independent of the specific training hyper-parameters, conversely to softmax classifiers trained with gradient-based methods.

Table 3 provides this quantitative analysis, where we compute RR on the feature extractor of the model after fine-tuning, at convergence. The findings demonstrate an en-

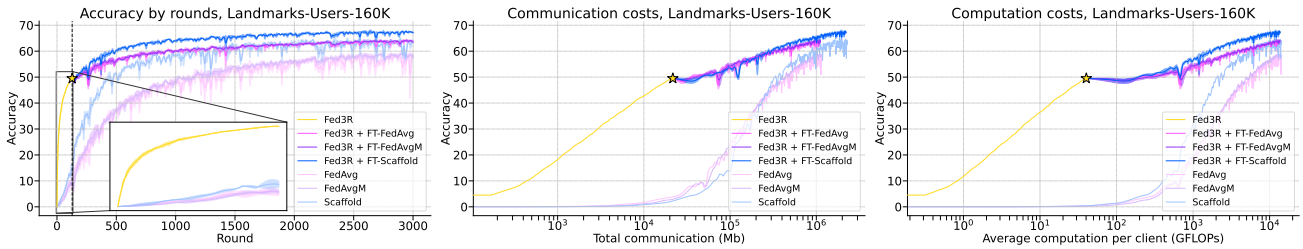


Figure 4: Comparison between FED3R+FT and the baselines Landmarks dataset. At the convergence point of FED3R, we substitute the parameters of the FED3R classifier to the ones of the softmax and then use another algorithm for fine-tuning.

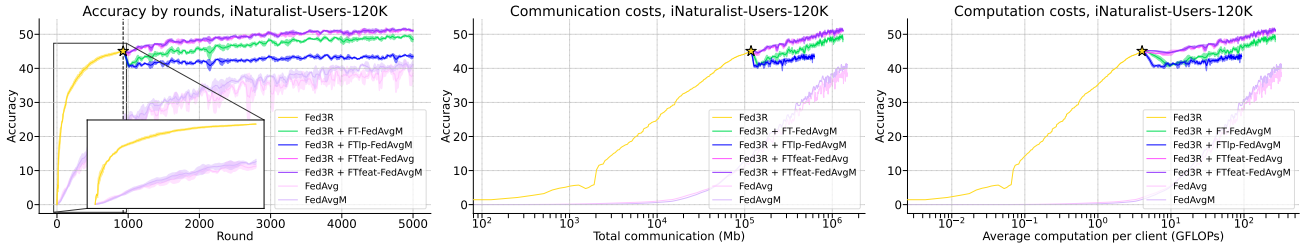


Figure 5: Comparison between FED3R+FT in all its variants and the baselines, iNaturalist dataset.

Table 3: Quality of the feature extractors (acc. %) at convergence measured with Ridge Regression.

Dataset	FT alg.	FT strategy	Cls Init.	Softmax	RR
Landmarks-Users-160K	-	-	FED3R	-	49.6 ± 0.0
	FedAvg	FEAT + LP	-	57.7 ± 1.2	58.8 ± 2.1
		FEAT + LP	FED3R	**64.1** ± **0.4**	59.6 ± 0.2
		FEAT	FED3R	59.6 ± 0.2	**62.1** ± **0.1**
	Scaffold	FEAT + LP	-	63.4 ± 0.9	57.0 ± 1.9
		FEAT + LP	FED3R	**67.4** ± **0.3**	61.8 ± 0.1
		FEAT	FED3R	63.4 ± 0.1	**64.3** ± **0.2**
iNaturalist-Users-120K	-	-	FED3R	-	45.1 ± 0.0
	FedAvg	FEAT + LP	-	39.5 ± 3.2	53.1 ± 0.9
		FEAT + LP	FED3R	49.0 ± 0.6	52.2 ± 0.3
		FEAT	FED3R	**50.8** ± **0.2**	**54.6** ± **0.1**

hancement in the quality of the learned features with the robust FED3R initialization. This initialization aids in stabilizing the training process, reducing destructive interference and forgetting caused by heterogeneity. Specifically, FED3R+FT and FED3R+FTFEAT consistently yield higher RR accuracy than the corresponding baseline with the same fine-tuning algorithm, for both the Landmarks and iNaturalist datasets. Moreover, FED3R+FTFEAT consistently outperforms FED3R+FT in all cases, as keeping the classifier fixed completely prevents the classifier bias.

Furthermore, in realistic cross-device scenario as iNaturalist, RR at convergence achieves even higher accuracy than the softmax classifier in all the fine-tuning strategies. This observation suggests the possibility of executing FED3R after the training process to further improve performance.

6. Conclusion

In this work, we introduce FED3R, a family of Federated Learning algorithms based on Recursive Ridge Regression. FED3R is designed to minimize communication and computation costs and accelerate convergence speed while adhering to the privacy constraints of FL. Unlike gradient-based FL algorithms where statistical heterogeneity is a significant challenge, FED3R is immune to statistical heterogeneity by design and can also serve as a robust initialization for further fine-tuning with optimization-based FL algorithms. Results show that our algorithm requires up to two orders of magnitude less communication and computation costs to convergence than the baselines (see Figure 2) and improves the accuracy up to 12% in challenging cross-device FL scenarios (see Table 2, iNaturalist results). Finally, our findings reveal that the features produced during the fine-tuning stage are more robust than those achieved by other methods at convergence see Table 3). This underscores the notion that in challenging cross-device settings, the quality of the feature extractor may serve as a bottleneck alongside the classifier's quality. Future works may extend FED3R to streaming data or personalized learning scenarios within the FL framework.

Impact Statement

FED3R significantly enhances training efficiency by providing remarkable speed and minimal computational and communication costs. By lightening the FL training load, our algorithm not only improves efficiency but also reduces the energy required for training. This not only benefits cost savings but also contributes to reducing environmental pollu-

tion associated with training models, as the energy required for training may still be generated using unsustainable methods. This has the potential to significantly impact various applications across industries, making them more accessible and cost-effective. The rapid execution and resource efficiency of our method could lead to increased adoption of FL techniques, enabling advancements in fields ranging from healthcare and finance to autonomous systems and beyond.

Acknowledgments

This study was carried out within the FAIR - Future Artificial Intelligence Research and received funding from the European Union Next-GenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.3 – D.D. 1555 11/10/2022, PE00000013). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them. We acknowledge the CINECA award under the ISCRA initiative for the availability of high-performance computing resources and support. We also thank the reviewers and area chair for their valuable comments.

References

Acar, D. A. E., Zhao, Y., Navarro, R. M., Mattina, M., Whatmough, P. N., and Saligrama, V. Federated learning based on dynamic regularization. *International Conference on Learning Representations*, 2021.

Afonin, A. and Karimireddy, S. P. Towards model agnostic federated learning using knowledge distillation. *arXiv preprint arXiv:2110.15210*, 2021.

Babakniya, S., Elkordy, A. R., Ezzeldin, Y. H., Liu, Q., Song, K.-B., El-Khamy, M., and Avestimehr, S. Slora: Federated parameter efficient fine-tuning of language models. *arXiv preprint arXiv:2308.06522*, 2023.

Bartlett, P. L., Jordan, M. I., and McAuliffe, J. D. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Bishop, C. Pattern recognition and machine learning. *Springer google schola*, 2:531–537, 2006.

Björck, Å. *Numerical methods for least squares problems*. SIAM, 1996.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. Practical secure aggregation for federated learning on user-held data. *arXiv preprint arXiv:1611.04482*, 2016.

Boyd, S. P. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Cai, J., Liu, X., Yu, Z., Guo, K., and Li, J. Efficient vertical federated learning method for ridge regression of large-scale samples. *IEEE Transactions on Emerging Topics in Computing*, 2022.

Caldarola, D., Caputo, B., and Ciccone, M. Improving generalization in federated learning by seeking flat minima. In *European Conference on Computer Vision*, pp. 654–672. Springer, 2022.

Camoriano, R., Pasquale, G., Ciliberto, C., Natale, L., Rosasco, L., and Metta, G. Incremental robot learning of new objects with fixed update time. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3207–3214. IEEE, 2017.

Caponnetto, A. and De Vito, E. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7:331–368, 2007.

Chen, J., Xu, W., Guo, S., Wang, J., Zhang, J., and Wang, H. Fedtune: A deep dive into efficient federated fine-tuning with pre-trained transformers. *arXiv preprint arXiv:2211.08025*, 2022.

Cho, Y. J., Liu, L., Xu, Z., Fahrezi, A., and Joshi, G. Heterogeneous low-rank approximation for federated fine-tuning of on-device foundation models. *arXiv preprint arXiv:2401.06432*, 2024.

Davari, M., Asadi, N., Mudur, S., Aljundi, R., and Belilovsky, E. Probing representation forgetting in supervised and unsupervised continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16712–16721, 2022.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and 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.

Fallah, A., Mokhtari, A., and Ozdaglar, A. Personalized federated learning: A meta-learning approach. *Advances in Neural Information Processing Systems*, 2020.

Fanì, E., Ciccone, M., and Caputo, B. Feddrive v2: an analysis of the impact of label skewness in federated semantic segmentation for autonomous driving. *5th Italian Conference on Robotics and Intelligent Machines (I-RIM)*, 2023.

Fantauzzo, L., Fanì, E., Caldarola, D., Tavera, A., Cermelli, F., Ciccone, M., and Caputo, B. Feddrive: Generalizing federated learning to semantic segmentation in autonomous driving. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 11504–11511. IEEE, 2022.

Ferrante, M. and Frigo, N. A note on the coupon-collector's problem with multiple arrivals and the random sampling. *arXiv preprint arXiv:1209.2667*, 2012.

Ferrante, M. and Saltalamacchia, M. The coupon collector's problem. *Materials matemàtics*, pp. 0001–35, 2014.

Gu, X., Huang, K., Zhang, J., and Huang, L. Fast federated learning in the presence of arbitrary device unavailability. *Advances in Neural Information Processing Systems*, 34: 12052–12064, 2021.

Guo, T., Guo, S., Wang, J., Tang, X., and Xu, W. Promptfl: Let federated participants cooperatively learn prompts instead of models-federated learning in age of foundation model. *IEEE Transactions on Mobile Computing*, 2023.

Hager, W. W. Updating the inverse of a matrix. *SIAM review*, 31(2):221–239, 1989.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Hsu, T.-M. H., Qi, H., and Brown, M. Measuring the effects of non-identical data distribution for federated visual classification. *Neurips Workshop on Federated Learning*, 2019.

Hsu, T.-M. H., Qi, H., and 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.

Huang, L., Li, Z., Sun, J., and Zhao, H. Coresets for vertical federated learning: Regularized linear regression and k-means clustering. *Advances in Neural Information Processing Systems*, 35:29566–29581, 2022.

Kailath, T., Sayed, A. H., and Hassibi, B. *Linear estimation*. Prentice Hall, 2000.

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.

Kang, B., Xie, S., Rohrbach, M., Yan, Z., Gordo, A., Feng, J., and Kalantidis, Y. Decoupling representation and classifier for long-tailed recognition. In *Eighth International Conference on Learning Representations (ICLR)*, 2020.

Karimireddy, S. P., Jaggi, M., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. Mime: Mimicking centralized stochastic algorithms in federated learning. *Advances in Neural Information Processing Systems*, 2020a.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020b.

Kim, D. and Han, B. On the stability-plasticity dilemma of class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20196–20204, 2023.

Kim, G., Kim, J., and Han, B. Communication-efficient federated learning with acceleration of global momentum. *arXiv preprint arXiv:2201.03172*, 2022.

Kim, T., Lin, E., Lee, J., Lau, C., and Mugunthan, V. Navigating data heterogeneity in federated learning: A semi-supervised approach for object detection. *Advances in Neural Information Processing Systems*, 36, 2024.

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.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Legate, G., Bernier, N., Caccia, L., Oyallon, E., and Belilovsky, E. Guiding the last layer in federated learning with pre-trained models. In *Advances in Neural Information Processing Systems*, volume 36, 2023a.

Legate, G., Caccia, L., and Belilovsky, E. Re-weighted softmax cross-entropy to control forgetting in federated learning. *arXiv preprint arXiv:2304.05260*, 2023b.

Li, Q., Diao, Y., Chen, Q., and He, B. Federated learning on non-iid data silos: An experimental study. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pp. 965–978. IEEE, 2022.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020a.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. On the convergence of fedavg on non-iid data. *International Conference on Learning Representations*, 2020b.

Li, Z., Shang, X., He, R., Lin, T., and Wu, C. No fear of classifier biases: Neural collapse inspired federated learning with synthetic and fixed classifier. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5319–5329, October 2023.

Liu, Q., Chen, C., Qin, J., Dou, Q., and Heng, P.-A. Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space. *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Liu, Y., Sun, Y., Ding, Z., Shen, L., Liu, B., and Tao, D. Enhance local consistency in federated learning: A multi-step inertial momentum approach. *arXiv preprint arXiv:2302.05726*, 2023.

Luo, M., Chen, F., Hu, D., Zhang, Y., Liang, J., and Feng, J. No fear of heterogeneity: Classifier calibration for federated learning with non-iid data. *Advances in Neural Information Processing Systems*, 34:5972–5984, 2021.

Lyu, Y., Wang, L., Zhang, X., Sun, Z., Su, H., Zhu, J., and Jing, L. Overcoming recency bias of normalization statistics in continual learning: Balance and adaptation. *Advances in Neural Information Processing Systems*, 36, 2024.

Mai, Z., Li, R., Kim, H., and Sanner, S. Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3589–3599, 2021.

Masana, M., Liu, X., Twardowski, B., Menta, M., Bagdanov, A. D., and Van De Weijer, J. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5513–5533, 2022.

McCloskey, M. and Cohen, N. J. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Nguyen, A. T., Torr, P., and Lim, S.-N. FedSR: A simple and effective domain generalization method for federated learning. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=mrt90D00aQX.

Nguyen, J., Malik, K., Sanjabi, M., and Rabbat, M. Where to begin? exploring the impact of pre-training and initialization in federated learning. *International Conference on Learning Representations*, 2023.

Oh, J., Kim, S., and Yun, S.-Y. Fedbabu: Towards enhanced representation for federated image classification. *International Conference on Learning Representations*, 2021.

Ozfatura, E., Ozfatura, K., and Gündüz, D. Fedadc: Accelerated federated learning with drift control. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pp. 467–472. IEEE, 2021.

Papyan, V., Han, X., and Donoho, D. L. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

Ramasesh, V. V., Dyer, E., and Raghu, M. Anatomy of catastrophic forgetting: Hidden representations and task semantics. In *International Conference on Learning Representations*, 2020.

Ratcliff, R. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.

Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H. B. Adaptive federated optimization. *International Conference on Learning Representations*, 2021.

Reddi, S. J., Hefny, A., Sra, S., Poczos, B., and Smola, A. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pp. 314–323. PMLR, 2016.

Richtárik, P. and Takáč, M. Distributed coordinate descent method for learning with big data. *The Journal of Machine Learning Research*, 17(1):2657–2681, 2016.

Rifkin, R., Yeo, G., Poggio, T., et al. Regularized least-squares classification. *Nato Science Series Sub Series III Computer and Systems Sciences*, 190:131–154, 2003.

Ruan, Y., Zhang, X., Liang, S.-C., and Joe-Wong, C. Towards flexible device participation in federated learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 3403–3411. PMLR, 2021.

Rudi, A. and Rosasco, L. Generalization properties of learning with random features. *Advances in neural information processing systems*, 30, 2017.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and 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.

Sayed, A. H. *Adaptive Filters*. Wiley-IEEE Press, 2008.

Shang, X., Lu, Y., Huang, G., and Wang, H. Federated learning on heterogeneous and long-tailed data via classifier re-training with federated features. *arXiv preprint arXiv:2204.13399*, 2022.

Shawe-Taylor, J. and Cristianini, N. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

Sherman, J. and Morrison, W. J. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, 21 (1):124–127, 1950.

Stadje, W. The collector's problem with group drawings. *Advances in Applied Probability*, 22(4):866–882, 1990.

Steinwart, I. and Christmann, A. *Support vector machines*. Springer Science & Business Media, 2008.

Stigler, S. M. Gauss and the invention of least squares. *the Annals of Statistics*, pp. 465–474, 1981.

Tan, A. Z., Yu, H., Cui, L., and Yang, Q. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Van Horn, G., Mac Aodha, O., Song, Y., Cui, Y., Sun, C., Shepard, A., Adam, H., Perona, P., and Belongie, S. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8769–8778, 2018.

Varno, F., Saghayi, M., Rafiee Sevyeri, L., Gupta, S., Matwin, S., and Havaei, M. Adabest: Minimizing client drift in federated learning via adaptive bias estimation. In *European Conference on Computer Vision*, pp. 710–726. Springer, 2022.

Wang, J., Tantia, V., Ballas, N., and Rabbat, M. Slowmo: Improving communication-efficient distributed sgd with slow momentum. *International Conference on Learning Representations*, 2019.

Wang, J., Liu, Q., Liang, H., Joshi, G., and Poor, H. V. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.

Wang, R., Ciccone, M., Luise, G., Pontil, M., Yapp, A., and Ciliberto, C. Schedule-robust online continual learning. *arXiv preprint arXiv:2210.05561*, 2022.

Weyand, T., Araujo, A., Cao, B., and 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.

Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., and Fu, Y. Large scale incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 374–382, 2019.

Xu, J., Wang, S., Wang, L., and Yao, A. C.-C. Fedcm: Federated learning with client-level momentum. *arXiv preprint arXiv:2106.10874*, 2021.

Yang, Q., Liu, Y., Chen, T., and Tong, Y. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

Yang, T., Andrew, G., Eichner, H., Sun, H., Li, W., Kong, N., Ramage, D., and Beaufays, F. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018.

Yi, L., Yu, H., Wang, G., and Liu, X. Fedlora: Model-heterogeneous personalized federated learning with lora tuning. *arXiv preprint arXiv:2310.13283*, 2023.

Yu, Y., Wei, A., Karimireddy, S. P., Ma, Y., and Jordan, M. Tct: Convexifying federated learning using bootstrapped neural tangent kernels. *Advances in Neural Information Processing Systems*, 35:30882–30897, 2022.

Zhang, J., Vahidian, S., Kuo, M., Li, C., Zhang, R., Yu, T., Wang, G., and Chen, Y. Towards building the federatedgpt: Federated instruction tuning. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6915–6919. IEEE, 2024.

Zhang, Y., Wainwright, M. J., and Duchi, J. C. Communication-efficient algorithms for statistical optimization. *Advances in neural information processing systems*, 25, 2012.

Zhang, Y., Duchi, J., and Wainwright, M. Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates. *The Journal of Machine Learning Research*, 16(1):3299–3340, 2015.

Zhou, T., Zhang, J., and Tsang, D. Fedfa: federated learning with feature anchors to align feature and classifier for heterogeneous data. *arXiv preprint arXiv:2211.09299*, 2022.

A. Additional Related Works

In this section, we expand on the related works concerning Ridge Regression in Distributed and Federated Learning settings and Transfer Learning methods involving pre-trained models in FL.

Ridge Regression in distributed and federated learning. Regarding prior works on Ridge Regression in FL settings, Afonin & Karimireddy (2021) finds an optimal Tikhonov Regularized Least Squares solution for a federation of only two clients that, in practice, constitutes a cross-knowledge distillation framework. Cai et al. (2022) and Huang et al. (2022) apply Ridge Regression to a *Vertical FL* scenario in which each client possesses different features of all the samples. Conversely, in this work, we focus on the more common *Horizontal* FL (Yang et al., 2019) setting, where the feature space is shared among clients, but local datasets vary. The most critical distinction between Federated RR methods for V-FL and H-FL concerns how best to compute gradients and aggregate statistics in a privacy-preserving manner while having the global dataset partitioned in a fundamentally different way across clients. Such radically different splitting strategies result in distinct algorithm design choices. For example, client drift is a major challenge in H-FL, and algorithms aim to reduce the effect of biased local gradients during aggregation. On the other hand, V-FL methods mostly focus on reducing the communication costs for computing good loss and gradient estimates based on a reduced feature set while preserving privacy. Importantly, we also have to face severe statistical heterogeneity in H-FL, which is typically not an issue in V-FL since it concerns only Cross-Silo scenarios. To the best of our knowledge, we are the first to apply RR within this specific context by leveraging its online formulation as an alternative to gradient-based optimization to speed up training and improve communication efficiency in realistic heterogeneous cross-device scenarios.

Other works tackle large-scale least-squares problems in distributed learning and optimization settings (Zhang et al., 2012; 2015; Richtárik & Takáč, 2016). While similar in spirit, distributed settings differ fundamentally from FL as privacy is not a constraint, and data is usually assumed i.i.d. across clients.

Transfer learning methods with pre-trained models in FL. The work presented in Oh et al. (2021) proposes a two-stage algorithm that uses a fixed, random classifier and trains only the feature extractor. However, while Oh et al. (2021) focuses on the Personalized FL setting by specializing in the head for each client, our work addresses the classifier bias problem in the conventional FL setting, where the goal is to learn a global classifier that represents the overall underlying distribution. Similarly, Kim et al. (2024) outlines a method for semi-supervised learning scenarios and object detection to selectively train the model's backbone while keeping the rest of the model frozen. They claim this approach helps train more consistent representations and establishes a stronger backbone for further fine-tuning with an extra regularization term.

Transfer learning techniques have recently garnered attention in the FL community to make Foundation Models suitable for the cross-device setting (Guo et al., 2023; Chen et al., 2022; Zhang et al., 2024). These techniques exploit methods based on Low-Rank Approximation for parameter-efficient fine-tuning (Babakniya et al., 2023; Cho et al., 2024; Yi et al., 2023).

B. Privacy of FED3R

In the context of FED3R, clients have to transmit only the A_k and b_k statistics. Some may express concerns about the potential information leakage inherent in sharing this data, which extends beyond the disclosure associated with merely sharing model weights or gradients. However, it is crucial to note that any information the clients send to the server only needs to be aggregated. In other words, the server does not necessitate accessing individual values but rather needs solely to use the aggregated results. Therefore, privacy can be easily achieved by employing the Secure Aggregation protocol (Bonawitz et al., 2016).

C. Additional Implementation Details

FED3R requires only one communication round with each client, which can occur as soon as the clients are ready. However, to guarantee privacy, we simulate the server waiting for a group of clients, similar to classical algorithms. In this way, a practical implementation might incorporate a Secure Aggregation (Bonawitz et al., 2016) step, where the information provided by individual clients is concealed within the aggregation of statistics shared by all sampled clients.

We run all the experiments using an NVIDIA A100-SXM4-40GB using the FL clients partitions provided by (Hsu et al., 2020) for Landmarks (Weyand et al., 2020) and iNaturalist (Van Horn et al., 2018). For the cifar100 (Krizhevsky et al., 2009) experiments, we focus on the most heterogeneous case ($\alpha = 0$, (Hsu et al., 2019)), where each client has access to images belonging to the same single class.

Details on the datasets are provided in Table 4. We used a MobileNetV2 (Sandler et al., 2018) network pre-trained on ImageNet for all our experiments. We replicate the same augmentation used to pre-train the model to best exploit the

Table 4: Datasets additional information.

Dataset	**Avg. samples per client**	K	C
Landmark-Users-160K	119.9	1262	2028
iNaturalist-Users-120K	13.0	9275	1203
iNaturalist-Geo-100	33.4	3606	1203
iNaturalist-Geo-300	99.6	1208	1203
iNaturalist-Geo-1K	326.9	368	1203
Cifar100	500	100	100

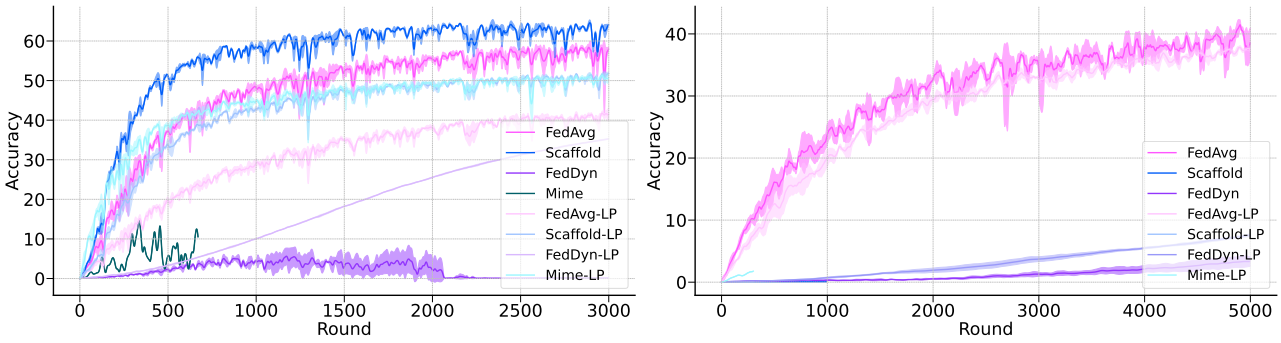


Figure 6: Comparison between baseline algorithms and their LP versions. Left: Landmark-Users-160K; right: iNaturalist-Users-120K.

pre-trained features. Therefore, we scaled all the images to 224×224, even for the 32×32 images of Cifar100.

We conducted the Landmarks experiments for 3000 rounds, the iNaturalist experiments for 5000 rounds, and the Cifar100 experiments for 1500 rounds. We sampled 10 clients per round in all three cases unless stated otherwise. We utilized SGD as the client optimizer with a learning rate (lr) of 0.1 and a weight decay (wd) of 4×10^{-5}, a batch size of 50, and 5 local epochs for both Landmarks and iNaturalist, and 1 local epoch for Cifar100. Additionally, we employed SGD as the server optimizer (Reddi et al., 2021) with a learning rate (slr) set to 1.0 and no momentum (smom). The best hyper-parameters were the same across all methods and datasets, selected based on a grid search: $lr = \{0.1, 0.01\} \times slr = \{0.1, 1.0\} \times wd = \{0.0, 4 \cdot 10^{-5}\} \times smom = \{0.0, 0.9\}$. For the FED3R λ hyper-parameter, we set $\lambda = 0.01$ as it consistently yielded the best RR results.

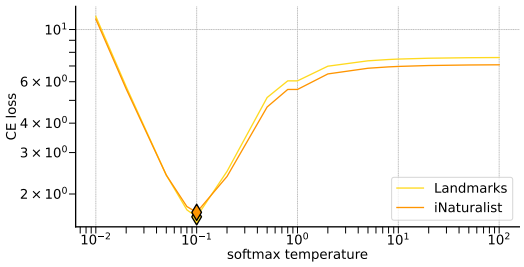
Despite our best efforts and hyper-parameters tuning, Scaffold failed to converge on the iNaturalist dataset, although it converges on the Landmarks experiments. We attribute this to its initial design for cross-silo settings, as control variates become stale in realistic cross-device scenarios (Karimireddy et al., 2020a).

Figure 6 shows the performance of the baseline algorithms with the best hyper-parameters from the grid search. Only FedAvg (and FedAvgM, which had similar results to FedAvg and was not included in the two plots for clarity) consistently performs well on both datasets. At the same time, the other algorithms struggle, especially in the realistic cross-device scenario of iNaturalist, where FedAvg is the only baseline algorithm that works.

In all our FED3R-RF experiments, we considered a random features approximation of an RBF kernel $k(z, \zeta) = e^{-\|z-\zeta\|^2/2\sigma^2}$, $z, \zeta \in \mathbb{R}^d$. The hyper-parameter σ has been tuned once in the centralized Landmarks setting, and the best value $\sigma = 1000$ has been selected for all the experiments. For the FED3R+FT, FED3R+FTLP, FED3R+FTFEAT experiments, we found that the softmax temperature value of 0.1 yields the best results on both the Landmarks and iNaturalist datasets, as Figure 7 shows.

We assume all the values are stored as FP32 numbers to estimate the communication and computation costs. See Appendix D and Appendix E for more details.

Figure 7: Cross-entropy loss values evaluated on the training set using different softmax temperatures. The model is initialized with the FED3R classifier and the pre-trained feature extractor. The best temperature is 0.1 for both the Landmarks and iNaturalist datasets.



D. Communication Costs Computation

We initially estimate the costs per round for each client to evaluate the communication costs. The overall communication cost per round and client comprises two components: the downstream and upstream costs, representing the communication from and to the server, respectively, which may vary across different methods. With this figure in hand, we then determine the total costs per round by multiplying the cost per round per client by the number of sampled clients per round. In all the communication costs plots, we multiplied the final values by 4 to measure the final cost in bytes, as we assume all the parameters are stored as FP32 values, *i.e.*, 4 bytes.

Below, we briefly summarize how the downstream and upstream costs have been calculated per each algorithm and eventual additional costs. Let m, b, and c be the sizes of the whole model, the feature extractor, and the classifier, respectively. As the classifier is a linear layer, its size is equivalent to the product of the latent feature dimensionality and the number of classes of the dataset: $c = dC$. Therefore, $m = b + dC$. Then:

- **FedAvg (McMahan et al., 2017), FedAvgM (Hsu et al., 2019).** Each sampled client downloads and uploads the model only once: *Downstream/k* = $b + dC$, *Upstream/k* = $b + dC$.
- **Scaffold (Karimireddy et al., 2020b).** Each sampled client downloads and uploads both the model and its control variate: *Downstream/k* = $2(b + dC)$, *Upstream/k* = $2(b + dC)$.
- **FedAvg-LP, FedAvgM-LP.** Each sampled client downloads and uploads the classifier only once: *Downstream/k* = dC, *Upstream/k* = dC.
- **Scaffold-LP.** Each sampled client downloads and uploads both the classifier and its control variate: *Downstream/k* = $2dC$, *Upstream/k* = $2dC$.
- **FED3R, FED3R-RF.** Each client needs to receive the feature extractor parameters only once. If we do not assume clients already have the feature extractor parameters before the training begins (though this assumption is reasonable in scenarios where the server, as a business, deploys its application and may have already incorporated these parameters in the clients' software), there is an additional communication cost of bK. Except that for these costs, each sampled client does not need to download any information from the server, but it needs to upload the local statistics A_k, b_k to the server: *Downstream/k* = 0, *Upstream/k* = $d^2 + dC$. If we use FED3R-RF the upstream costs per client are *Upstream/k* = $D^2 + DC$ instead.
- **FED3R+FT**. In this scenario, the *Downstream/k* and *Upstream/k* costs correspond to those of FED3R during the initial phase of the experiments, when FED3R+FT generates the FED3R classifier as initialization for the softmax classifier. Subsequently, the costs reflect those of the FT algorithm, with the sole exception for FED3R+FTFEAT, where the FT phase costs are *Downstream/k* = *Upstream/k* = b for FedAvg and FedAvgM, and *Downstream/k* = *Upstream/k* = $2b$ for Scaffold.

E. Average Computation Costs per Client

We prioritize the average computation costs per client over the total computation cost among all clients. This decision stems from our belief that this statistic provides more insightful information regarding the budget required by an FL system developer for their clients.

To estimate this value, let $\mathcal{T}$ be the total average cost *per round* per single client for a given algorithm. Let $\boldsymbol{x}$ be the number of times a specific client is sampled. Then, the cumulative average cost $\mathcal{T}_t$ from round 1 to round t is proportional to the expected number of times $\mathbb{E}[\boldsymbol{x}]$ a specific client is sampled over t rounds, that is $\mathbb{E}[\boldsymbol{x}] = t\frac{\kappa}{K}$, where κ is the number of clients sampled per each round and K is the total number of clients. Therefore, $\mathcal{T}_t = \mathcal{T}\mathbb{E}[\boldsymbol{x}] = \mathcal{T}t\frac{\kappa}{K}$.

Table 5: MobileNetV2 (Sandler et al., 2018) forward MFLOPs.

Dataset	F_φ	F_ϕ	$F_\mathcal{M}$
Landmarks	332.9	2.6	335.5
iNaturalist	332.9	1.5	334.4
Cifar100	332.9	0.1	333.0

The specific $\mathcal{T}$ value depends on the algorithm. Let F_* and B_* be the costs of one forward pass of a single image and one backward pass of a single image through the model $*$. As the authors of (Legate et al., 2023a), we approximate $B_* \simeq 2F_*$, and consider both the forward F_*^N and backward B_*^N of a batch of N images as directly proportional to B and F, *i.e.*, $F_*^N = NF_*$ and $B_*^N = NB_*$. Therefore, one epoch's total forward and backward costs for a single client are simply $F_*^{n_k}$ and $B_*^{n_k}$.

We measure these costs in FLOPs. Therefore, we divide by half the count of the matrix operations since one FLOP is defined as one addition and one multiplication of floating point numbers. Let E be the number of local epochs. Then:

- **FedAvg (McMahan et al., 2017), FedAvgM (Hsu et al., 2019), Scaffold (Karimireddy et al., 2020b).** All these methods have one forward pass and one backward pass through the whole model and other negligible operations, such as SGD updates and computations of client control variates for Scaffold. Therefore, we consider the same total cost per round $\mathcal{T} = En_k(F_\mathcal{M} + B_\mathcal{M}) = 3En_kF_\mathcal{M}$.
- **FedAvg-LP, FedAvgM-LP, Scaffold-LP.** In this case, the forward is through the whole model, but the backward is only up to the classifier: $\mathcal{T} = En_k(F_\mathcal{M} + B_\phi) = En_k(F_\varphi + 3F_\phi)$.
- **FED3R**. The clients need to forward the input images through the feature extractor once. Then, they must compute the matrices $A_k = Z_k^T Z_k$ and $b_k = Z_k^T Y_k$. Since A_k is symmetric, computing A_k costs $\frac{1}{2}n_k d(d+1)$ FLOPs. Instead, computing b_k costs $n_k dC$ FLOPs. Therefore, $\mathcal{T} = n_k(F_\varphi + \frac{1}{2}d(d+1) + dC)$.
- **FED3R-RF**. The costs are the same of FED3R, with the sole exception that the latent feature space is D-dimensional here.
- **FED3R+FT**. The costs correspond to those of FED3R during the initial phase of the experiments, when FED3R+FT generates the FED3R classifier as initialization for the softmax classifier. Subsequently, the costs reflect those of the FT algorithm. For FED3R+FTFEAT, the FT phase costs are $\mathcal{T} = 3En_kF_\mathcal{M}$.

The specific forward FLOPs of the MobileNetV2 model are summarized in Table 5.

F. Centralized RR Results Using the Random Features

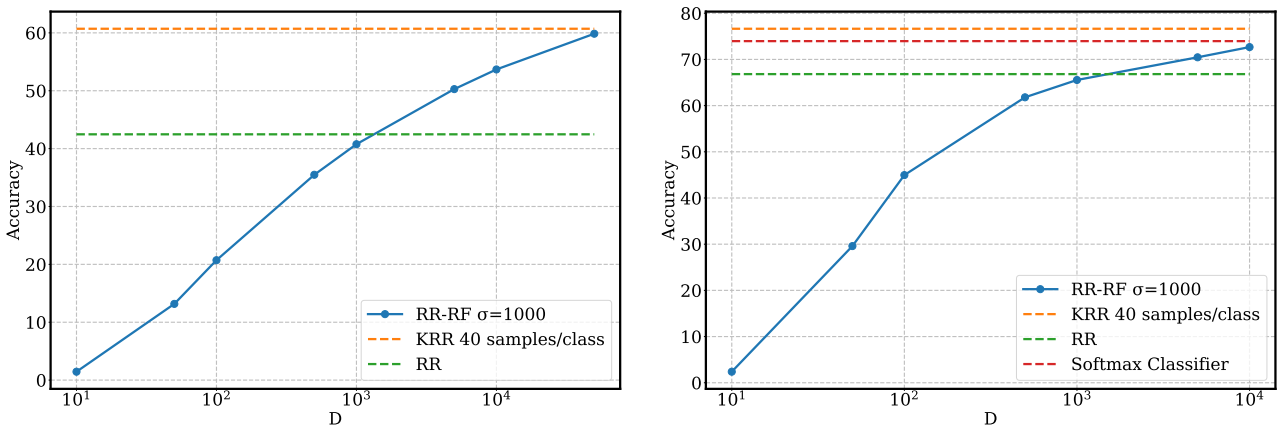
The outcomes of centralized experiments employing random features to approximate the RBF kernel empirically show that augmenting the number of random features significantly enhances performance. Specifically, Figure 8 illustrates how, with the random features approximation, the performance of RR calculated over the feature maps provided by the feature extractor across the entire Landmarks dataset eventually approaches the upper bound established by the exact KRR solution on a subset of the dataset, where a maximum of 40 images per class is considered.

It is noteworthy that the exact KRR solution was not computed over the entire dataset due to computational constraints. Indeed, the exact solution would require storing a kernel matrix of dimensionality $n \times n$, where $n = 164172$ for Landmarks. Nevertheless, utilizing the whole dataset or increasing the number of random features should theoretically improve results further.

In addition, Figure 8b empirically shows that KRR can even yield superior performance compared to a softmax classifier at convergence.

G. Cifar100 Experiments

Figure 9 shows the immunity to statistical heterogeneity property of FED3R and FED3R-RF also for several Cifar100 FL splits. The number of rounds necessary to converge is only 10 since there are only 100 clients, and we simulate a participation rate of 0.1 *i.e.*, we sample 10 clients per round. Moreover, Table 6 shows a comparison between the FED3R and FED3R-RF classifiers with the FEDNCM classifier (Legate et al., 2023a).



(a) Pre-trained on ImageNet

(b) Fine-tuned on Landmarks-Users-160K

Figure 8: (Centralized) RR using D random features to approximate the RBF kernel compared to the exact KRR solution with RBF kernel computed over a subset of the whole Landmarks-Users-160K dataset where there are at most 40 images per class, using the MobileNetV2 (Sandler et al., 2018) architecture. We keep $\sigma = 1000$ for both KRR and RR-RF.

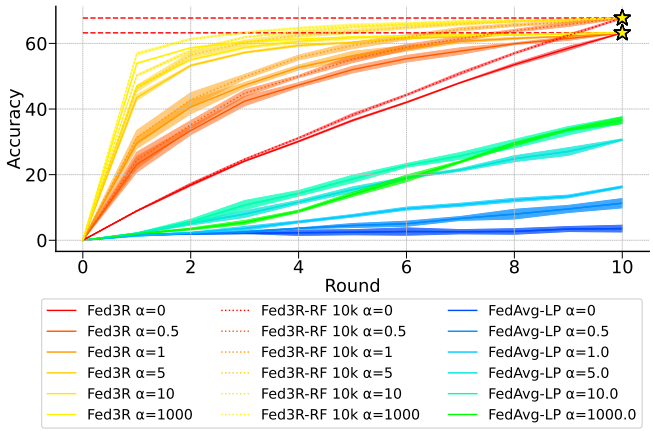


Figure 9: FED3R and FED3R-RF immunity to statistical heterogeneity showed with several Cifar100 FL splits. A lower value of α is associated with a higher level of statistical heterogeneity.

Table 6: Final accuracy (%) of the FED3R family of classifiers and FedNCM on the Cifar100 dataset.

Algorithm	**Accuracy** (%)
FED3R	63.2
FED3R-RF 5k	66.2
FED3R-RF 10k	**67.5**
FEDNCM	51.0

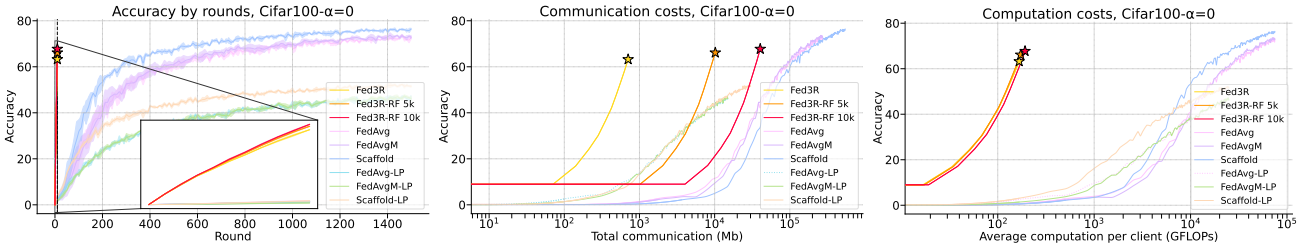


Figure 10: Comparison between FED3R and the baselines for the Cifar100-α=0 dataset.

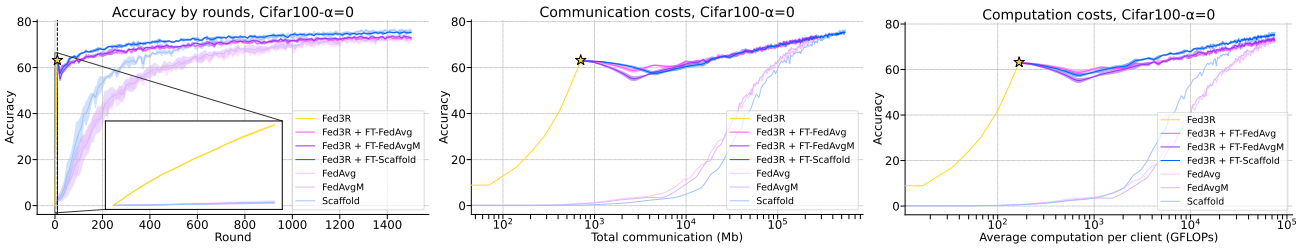


Figure 11: Comparison between FED3R+FT and the baselines for the Cifar100-α=0 dataset.

Furthermore, Figure 10 illustrates the performance and costs of both FED3R and FED3R-RF, whereas Figure 11 showcases the performance and costs of FED3R+FT. While FED3R alone may not achieve satisfactory performance compared to the baselines in this scenario, incorporating the fine-tuning stage remarkably enhances its performance to a comparable or superior level with respect to the baselines while retaining the same communication and computation advantages.

H. Best Methods Comparison

Figures 12 and 13 show a comparison among the best FED3R algorithms and the best baseline for the Landmarks and iNaturalist datasets, respectively.

FED3R+FT provides the best performance for Landmarks, and has better communication costs than FED3R-RF at the maximum FED3R-RF accuracy (56.6%). However, if the computation budget is the main constraint, FED3R-RF is the best method for computation budget $\leq 10^3$ GFLOPs. Similarly, FED3R-RF is the best method up to 10^2 GFLOPs per client in the iNaturalist experiments, although suffering high communication costs.

In all the cases, all our best-performing methods are much better than the best of the baselines, with the sole exception of the communication costs required to reach a target accuracy of FED3R-RF, which are comparable with the best baseline in the final stages of the training.

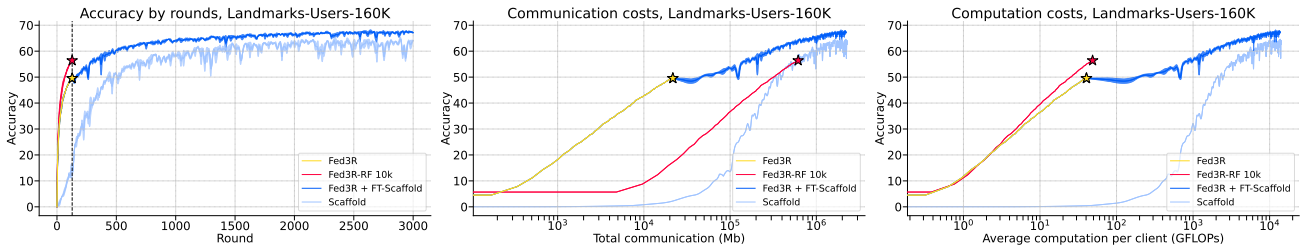


Figure 12: Comparison between the best FED3R methods and the best baseline method for the Landmarks dataset.

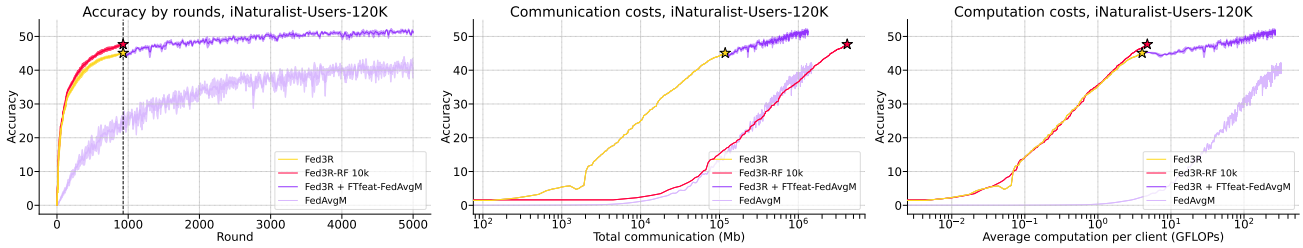


Figure 13: Comparison between the best FED3R methods and the best baseline method for the iNaturalist dataset.

I. Expected Number of Rounds to Sample Each Client at Least Once With Replacement

Given the total number of clients K in the federation and the number of clients κ sampled per round, it is theoretically possible to estimate the expected number of rounds that are necessary to sample each client at least once. This problem is known in the literature as the Batch Coupon Collector's Problem (Stadje, 1990; Ferrante & Frigo, 2012; Ferrante & Saltalamacchia, 2014).

Table 7 shows the average number of rounds necessary to sample a given percentage of clients in the corresponding settings after simulating the sampling with replacement one thousand times for each case. It is possible to observe how, to sample all the clients, many more rounds are needed on average than the rounds needed to sample 50% or 75% distinct clients. Interestingly, regarding the scenarios involving sampling with replacement as the one in Figure 3, this table provides insight into why FED3R achieves good performance with few rounds compared to the total needed to achieve full convergence.

Table 7: Average number of rounds necessary to sample the given percentage of clients in the corresponding settings when the clients are sampled with replacement.

Dataset	K	κ	Participation Rate (%)	25%	50%	75%	100%
Landmarks	1262	10	0.8	37 $\pm$ 1	88 $\pm$ 2	175 $\pm$ 4	970 $\pm$ 155
		20	1.6	19 $\pm$ 1	44 $\pm$ 1	87 $\pm$ 2	483 $\pm$ 79
		50	4.0	8 $\pm$ 0	18 $\pm$ 1	35 $\pm$ 1	191 $\pm$ 32
iNaturalist	9275	10	0.1	267 $\pm$ 2	643 $\pm$ 5	1286 $\pm$ 12	9020 $\pm$ 1189
		20	0.2	134 $\pm$ 1	322 $\pm$ 3	643 $\pm$ 6	4494 $\pm$ 596
		50	0.5	54 $\pm$ 1	129 $\pm$ 1	257 $\pm$ 2	1809 $\pm$ 247
Cifar100	100	10	10	3 $\pm$ 0	7 $\pm$ 1	14 $\pm$ 1	50 $\pm$ 12
		20	20	2 $\pm$ 0	4 $\pm$ 0	7 $\pm$ 1	24 $\pm$ 5
		50	50	1 $\pm$ 0	1 $\pm$ 0	3 $\pm$ 0	8 $\pm$ 2