ON BRIDGING GENERIC AND PERSONALIZED FEDERATED LEARNING FOR IMAGE CLASSIFICATION

Hong-You Chen
The Ohio State University, USA

Wei-Lun Chao
The Ohio State University, USA

ABSTRACT

Federated learning is promising for its capability to collaboratively train models with multiple clients without accessing their data, but vulnerable when clients' data distributions diverge from each other. This divergence further leads to a dilemma: *"Should we prioritize the learned model's generic performance (for future use at the server) or its personalized performance (for each client)?"* These two, seemingly competing goals have divided the community to focus on one or the other, yet in this paper we show that it is possible to approach both at the same time. Concretely, we propose a novel federated learning framework that explicitly *decouples a model's dual duties with two prediction tasks*. On the one hand, we introduce a family of losses that are robust to non-identical class distributions, enabling clients to train a *generic* predictor with a consistent objective across them. On the other hand, we formulate the *personalized* predictor as a lightweight adaptive module that is learned to minimize each client's empirical risk on top of the generic predictor. With this *two-loss, two-predictor* framework which we name **Federated Robust Decoupling (FED-ROD)**, the learned model can simultaneously achieve state-of-the-art generic and personalized performance, essentially bridging the two tasks.

1 INTRODUCTION

Large-scale data are the driving forces for modern machine learning but come with the risk of data privacy. In applications like health care, data are required to be kept separate to enforce ownership and protection, hindering the collective wisdom (of data) for training strong models. Federated learning (FL), which aims to train a model with multiple data sources (*i.e.*, clients) while keeping their data decentralized, has emerged as a popular paradigm to resolve these concerns (Kairouz et al., 2019).

The standard setup of FL seeks to train a single "global" model that can perform well on *generic* data distributions (Kairouz et al., 2019), *e.g.*, the union of clients' data. As clients' data are kept separate, mainstream algorithms like FEDAVG (McMahan et al., 2017) take a multi-round approach shown in Figure 1. Within each round, the server first broadcasts the "global" model to the clients, who then independently update the model locally using their own (often limited) data. The server then aggregates the "local" models back into the "global" model and proceeds to the next round. This pipeline is shown promising if clients' data are IID (*i.e.*, with similar data and label distributions) (Stich, 2019; Zhou & Cong, 2017), which is, however, hard to meet in reality and thus results in a drastic performance drop (Li et al., 2020b; Zhao et al., 2018). Instead of sticking to a single "global" model that features the generic performance, another setup of FL seeks to construct a "personalized" model for each client to acknowledge the heterogeneity among clients (Dinh et al., 2020; Hanzely et al., 2020; Smith et al., 2017). This latter setup (usually called **personalized FL**) is shown to outperform the former (which we name **generic FL**) regarding the test accuracy of each client alone.

So far, these two seemingly contrasting FL setups are developed independently. In this paper, we however found that they can be approached simultaneously by generic FL algorithms like FEDAVG.

Concretely, algorithms designed for generic FL (G-FL) often discard the local models $\{\boldsymbol{w}_m\}$ after training (see Figure 1). As a result, when they are evaluated in a personalized setting (P-FL), it is the global model $\bar{\boldsymbol{w}}$ being tested (Arivazhagan et al., 2019; Dinh et al., 2020; Fallah et al., 2020; Li et al., 2021a; Liang et al., 2020; Smith et al., 2017; Zhang et al., 2021). Here, we found that if we instead keep $\{\boldsymbol{w}_m\}$ and evaluate them in P-FL, they outperform nearly all the existing P-FL algorithms. In other words, *personalized models seem to come for free from the local training step of generic FL.*

At first glance, this may not be totally surprising: local training in G-FL algorithms is driven by the client's empirical risk, which is what a personalized model strives to optimize[1]. What really surprises us is that even without an explicit regularization term imposed by most P-FL algorithms (Dinh et al., 2020; Smith et al., 2017), the local models of G-FL algorithms can achieve better generalization performance. We conduct a detailed analysis and argue that global aggregation — taking average over model weights — indeed acts like a regularizer for local models. Moreover, applying advanced G-FL algorithms (Acar et al., 2021) to improve the G-FL accuracy seems to not hurt the "local" models' P-FL accuracy.

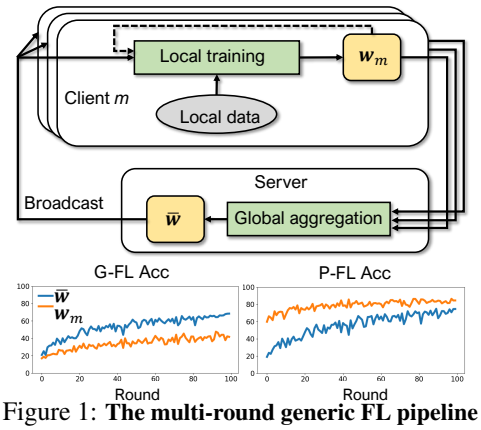


Figure 1: **The multi-round generic FL pipeline (top).** The dashed arrow indicates that local models or statistics may be carried to the next round. Here we apply FEDAVG (McMahan et al., 2017) on CIFAR-10 with 20 non-IID clients (see section 5), and show that *personalized models come for free from generic FL* **(bottom)**. The global model $\bar{w}$ outperforms local models w_m on the bottom-left generic accuracy (G-FL), yet w_m outperforms $\bar{w}$ on the bottom-right personalized accuracy (P-FL). The accuracy is computed at the end of each round.

Building upon these observations, we dig deeper into generic FL. Specifically for classification, the non-IID clients can result from non-identical **class distributions** or non-identical **class-conditional data distributions**. One way to mitigate their influences is to make the local training objectives more aligned among clients. While this can hardly be achieved for the latter case without knowing clients' data, we can do so for the former case by setting a consistent goal among clients — *the learned local models should classify every class well, even if clients' data have different class distributions.* We realize this by viewing each client's local training as an independent class-imbalanced problem (Cui et al., 2019; He & Garcia, 2009) and applying objective functions dedicated to it (Cao et al., 2019; Ren et al., 2020). As will be shown in section 5, these class-balanced objectives lead to much consistent local training among clients, making the resulting "global" model more robust to non-IID conditions.

The use of class-balanced objectives, nevertheless, degrades the local models' P-FL performance. This is because the local models are no longer learned to optimize clients' empirical risks.

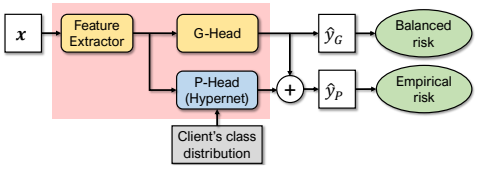


Figure 2: **Local training of FED-ROD.** Yellow/blue boxes are the models for G-FL/P-FL. Green ellipsoids are the learning objectives. The red area means what to be aggregated at the server.

To address this issue, we propose a unifying framework for G-FL and P-FL which explicitly *decouples a local model's dual duties*: serving as the personalized model and the ingredient of the global model. Concretely, we follow the FEDAVG pipeline and train the local model with the *class-balanced loss*, but on top of the feature extractor, we introduce a lightweight personalized predictor and train it with client's *empirical risk* (see Figure 2). With this *two-loss, two-predictor* framework which we name **Federated Robust Decoupling (FED-ROD)**, the resulting global model can be more robust to non-identical class distributions; the personalized predictor can lead to decent P-FL accuracy due to the implicit regularization and the empirical loss. Specifically for the personalized predictor, we propose to explicitly parameterize it with clients' class distributions via a hypernetwork (Ha et al., 2017). That is, we learn a shared meta-model that outputs personalized predictors for clients given their class distributions. This not only enables *zero-shot model adaptation* to new clients (without their data but class distributions), but also provides a better initialization to fine-tune the models given new clients' data.

We validate FED-ROD on multiple datasets under various non-IID settings. FED-ROD consistently outperforms existing generic and personalized FL algorithms in both setups. Moreover, FED-ROD is compatible with and can further improve advanced generic FL algorithms like FEDDYN (Acar et al., 2021) whenever non-identical class distributions occur. Our contributions are three-folded:

- Unlike most of the previous works that focus on either generic FL or personalized FL, we propose FED-ROD to excel on both at the same time. We validate FED-ROD with extensive experiments.

[1]However, when G-FL algorithms are tested on the P-FL setup, the literature does not use their local models.

- We show that strong personalized models emerge from the local training step of generic FL algorithms, due to implicit regularization. We further show that class-balanced objectives are effective for improving the generic FL performance when clients have different class distributions.
- FED-ROD enables zero-shot adaptation and much effective fine-tuning for new clients.

2 RELATED WORK (A DETAILED VERSION IS IN APPENDIX A)

Generic federated learning. FEDAVG (McMahan et al., 2017) is the standard algorithm, and many works are proposed to improve it, either in the **global aggregation** step (Chen & Chao, 2021; Hsu et al., 2019; Lin et al., 2020; Reddi et al., 2021; Wang et al., 2020a; Yurochkin et al., 2019) or **local training** step (Malinovskiy et al., 2020; Wang et al., 2020b; Yuan & Ma, 2020; Zhao et al., 2018). For example, to reduce local models' drifts from the global model, FEDPROX (Li et al., 2020a) and FEDDYN (Acar et al., 2021) employed regularization toward the global model; SCAFFOLD (Karimireddy et al., 2020a) leveraged control variates to correct local gradients. We also aim to reduce local models' drifts but via a different way. We apply objective functions in class-imbalanced learning (He & Garcia, 2009), which are designed to be robust to class distribution changes. The closest to ours is (Hsu et al., 2020), which used a traditional class-imbalanced treatment named re-weighting. We show that more advanced techniques can be applied to further improve the performance, especially under extreme non-IID conditions where re-weighting is ineffective.

Personalized federated learning. Many approaches for personalized FL (Kulkarni et al., 2020) are based on multi-task learning (MTL) (Ruder, 2017; Zhang & Yang, 2017). For instance, Smith et al. (2017) encouraged related clients to learn similar models; Dinh et al. (2020); Hanzely et al. (2020); Li et al. (2021a) regularized local models with a learnable global model. Our approach is inspired by MTL as well but has notable differences. First, we found that global aggregation in generic FL already serves as a strong regularizer. Second, instead of learning for each client a feature extractor (Bui et al., 2019; Liang et al., 2020) or an entire model, FED-ROD shares a single feature extractor among clients, inspired by Caruana (1997); Zhang et al. (2014). This reduces the total parameters to be learned and improves generalization. Compared to (Arivazhagan et al., 2019; Collins et al., 2021) which also learned a shared feature extractor, FED-ROD simultaneously excels in both FL setups.

Instead of designing specific algorithms for personalized FL, Cheng et al. (2021); Wang et al. (2019); Yu et al. (2020) showed that performing post-processing (*e.g.*, fine-tuning) to a generic FL model (*e.g.*, $\bar{\boldsymbol{w}}$ in FEDAVG) leads to promising personalized accuracy. We further showed that, the *local models* $\{\boldsymbol{w}_m\}$ learned in FEDAVG and other generic FL algorithms are strong personalized models.

We note that, while many personalized FL algorithms also produce a global model, it is mainly used to regularize or construct personalized models but not for evaluation in the generic setup. In contrast, we learn models to excel in both setups via a single framework without sacrificing either of them.

A recent work PFEDHN (Shamsian et al., 2021) also applies hypernetworks (Ha et al., 2017) but in a very different way from FED-ROD. PFEDHN learns a hypernetwork at the server to aggregate clients' updates and produce entire models for them for the next round. In contrast, we learn the hypernetwork locally to construct the personalized predictors, not the entire models, for fast adaptation to clients.

3 PERSONALIZED MODELS EMERGE FROM GENERIC FEDERATED LEARNING

In this section, we show that personalized FL (P-FL) models emerge from the training process of generic FL (G-FL) algorithms. To begin with, we review representative G-FL and P-FL algorithms.

3.1 BACKGROUND

Generic federated learning. In a generic FL setting with M clients, where each client has a data set $\mathcal{D}_m = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{|\mathcal{D}_m|}$, the optimization problem to solve can be formulated as

$$\min_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}) = \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \mathcal{L}_m(\boldsymbol{w}), \quad \text{where} \quad \mathcal{L}_m(\boldsymbol{w}) = \frac{1}{|\mathcal{D}_m|} \sum_i \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w}). \tag{1}$$

Here, $\boldsymbol{w}$ is the model parameter; $\mathcal{D} = \cup_m \mathcal{D}_m$ is the aggregated data set from all clients; $\mathcal{L}_m(\boldsymbol{w})$ is the empirical risk computed from client m's data; ℓ is a loss function applied to each data instance.

Federated averaging (FEDAVG). As clients' data are separate, Equation 1 cannot be solved directly. A standard way to *relax* it is FEDAVG (McMahan et al., 2017), which iterates between two steps, local training and global aggregation, for *multiple rounds of communication*

$$\textbf{Local:}\quad \boldsymbol{w}_m = \arg\min_{\boldsymbol{w}} \mathcal{L}_m(\boldsymbol{w}),\ \text{initialized with } \bar{\boldsymbol{w}}; \qquad \textbf{Global:}\quad \bar{\boldsymbol{w}} \leftarrow \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \boldsymbol{w}_m. \tag{2}$$

The local training is performed at all (or part of) the clients in parallel, usually with multiple epochs of SGD to produce the local model $\boldsymbol{w}_m$. The global aggregation is by taking element-wise average over model weights. Since local training is driven by clients' empirical risks, when clients' data are non-IID, $\boldsymbol{w}_m$ would drift away from each other, making $\bar{\boldsymbol{w}}$ deviate from the solution of Equation 1.

Personalized federated learning. Personalized FL learns for each client m a model $\boldsymbol{w}_m$, whose goal is to perform well on client m's data. While there is no agreed objective function so far, many existing works (Dinh et al., 2020; Hanzely & Richtárik, 2020; Hanzely et al., 2020; Li & Wang, 2019; Li et al., 2021a; Smith et al., 2017) define the optimization problems similar to the following

$$\min_{\{\boldsymbol{\Omega}, \boldsymbol{w}_1, \cdots, \boldsymbol{w}_M\}} \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \mathcal{L}_m(\boldsymbol{w}_m) + \mathcal{R}(\boldsymbol{\Omega}, \boldsymbol{w}_1, \cdots, \boldsymbol{w}_M), \tag{3}$$

where $\mathcal{R}$ is a regularizer; $\boldsymbol{\Omega}$ is introduced to relate clients. The regularizer is imposed to prevent $\boldsymbol{w}_m$ from over-fitting client m's limited data. Unlike Equation 1, Equation 3 directly seeks to minimize each client's empirical risk (plus a regularization term) by the corresponding personalized model $\boldsymbol{w}_m$.

In practice, personalized FL algorithms often run iteratively between the local and global steps as well, so as to update $\boldsymbol{\Omega}$ according to clients' models. One example is to define $\boldsymbol{\Omega}$ as a global model (Dinh et al., 2020; Hanzely & Richtárik, 2020; Hanzely et al., 2020; Li et al., 2021a), *e.g.*, by taking average over clients' models, and apply an L_2 regularizer between $\boldsymbol{\Omega}$ and each $\boldsymbol{w}_m$. The corresponding local training step thus could generally be formulated as

$$\textbf{Local:}\quad \boldsymbol{w}_m^{(t+1)} = \arg\min_{\boldsymbol{w}} \mathcal{L}_m(\boldsymbol{w}) + \frac{\lambda}{2}\|\boldsymbol{w} - \boldsymbol{\Omega}\|_2^2,\ \text{initialized with } \boldsymbol{w}_m^{(t)}, \tag{4}$$

where $\boldsymbol{w}_m^{(t)}$ denotes the local model after the t-th round; λ is the regularization coefficient. *It is worth noting that unlike Equation 2, $\boldsymbol{w}$ in Equation 4 is initialized by $\boldsymbol{w}_m^{(t)}$, not by $\boldsymbol{\Omega}$ (or $\bar{\boldsymbol{w}}$).*

Terminology. Let us clarify the concepts of "global" vs. "local" models, and "generic" vs. "personalized" models. The former corresponds to the **training** phase: local models are the ones after every round of local training, which are then aggregated into the global model at the server (Equation 2). The latter corresponds to the **testing** phase: the generic model is used at the server for generic future test data, while personalized models are specifically used for each client's test data.

3.2 LOCAL MODELS OF GENERIC FL ALGORITHMS ARE STRONG PERSONALIZED MODELS

Building upon the aforementioned concepts, we investigate the literature and found that when generic FL algorithms are evaluated in the P-FL setup, it is their global models being tested. In contrast, when personalized FL algorithms are applied, it is their local models (*e.g.*, Equation 4) being tested. This discrepancy motivates us to instead evaluate generic FL algorithms using their local models.

Figure 1 summarizes the results (see section 5 for details). Using local models of FEDAVG (*i.e.*, Equation 2) notably outperforms using its global model in the P-FL setup. At first glance, this may not be surprising, as local training in FEDAVG is driven by clients' empirical risks. What really surprises us, as will be seen in section 5, is that FEDAVG*'s local models outperform most of the existing personalized FL algorithms, even if no explicit regularization is imposed in Equation 2.*

3.3 INITIALIZATION WITH WEIGHT AVERAGE IS A STRONG REGULARIZER

To gain a further understanding, we plot FEDAVG local models' accuracy on clients' training and test data. We do so also for a state-of-the-art personalized FL algorithm DITTO (Li et al., 2021a), whose local training step for producing personalized models is similar to Equation 4. As shown in Figure 3, FEDAVG has a lower training but higher test accuracy, implying that FEDAVG's local training is more regularized than Equation 4.

We attribute this effect to the initialization in Equation 2. Specifically, by initializing $\boldsymbol{w}$ with $\bar{\boldsymbol{w}}$, we essentially impose an L_2 regularizer $\frac{\lambda}{2}\|\boldsymbol{w}-\bar{\boldsymbol{w}}\|_2^2$ with $\lambda \to \infty$ at the beginning of each round of local training, followed by resetting λ to be 0. We found that this implicit regularization leads to a smaller value of $\|\boldsymbol{w}-\bar{\boldsymbol{w}}\|_2^2$ at the end of each local training round, compared to Equation 4. Due to the page limit, we leave additional analyses in the appendix. We note that, advanced generic FL algorithms like SCAFFOLD (Karimireddy et al., 2020b) and FEDDYN (Acar et al., 2021) still apply this initialization and learn with the empirical risk during local training. Thus, their local models are strong personalized models as well.

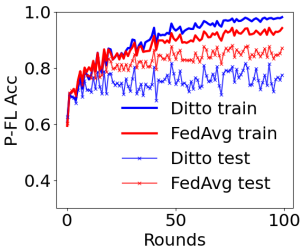


Figure 3: Comparison of the training and test accuracy in the P-FL setup. FEDAVG's local models achieve lower training accuracy but higher test accuracy.

4 FEDERATED ROBUST DECOUPLING (FED-ROD)

The fact that personalized models emerge from generic FL algorithms motivate us to focus more on how to improve the latter, especially when clients have non-IID data distributions.

4.1 IMPROVING GENERIC FL WITH BALANCED RISK MINIMIZATION (BRM)

We first analyze what factors may lead to non-IID conditions. Suppose the data instance $(\boldsymbol{x}, y)$ of client m is sampled from a client-specific joint distribution $\mathcal{P}_m(\boldsymbol{x}, y) = \mathcal{P}_m(\boldsymbol{x}|y)\mathcal{P}_m(y)$, the non-IID distributions among clients can result from non-identical class distributions $\mathcal{P}_m(\boldsymbol{x}|y)$, non-identical class-conditional data distributions $\mathcal{P}_m(y)$, or both. All these cases can make $\mathcal{L}_m(\boldsymbol{w})$ deviate from $\mathcal{L}(\boldsymbol{w})$ in Equation 1, which is the main cause of degradation in generic FL (Li et al., 2020a;b).

One way to mitigate the influence of non-IID data is to make $\mathcal{L}_m(\boldsymbol{w})$ align with each other. This can be challenging to achieve if clients have different $\mathcal{P}_m(\boldsymbol{x}|y)$: without knowing clients' data[2], it is hard to design such an aligned $\mathcal{L}_m(\boldsymbol{w})$. However, when clients have different $\mathcal{P}_m(y)$[3], *i.e.*, different and hence imbalanced class distributions, we can indeed design a consistent local training objective by setting a shared goal for the clients — *the learned local models should classify all the classes well.* It is worth noting that setting such a goal does not require every client to know others' data.

Learning a classifier to perform well on all classes irrespective of the training class distribution is the main focus of class-imbalanced learning (He & Garcia, 2009; Japkowicz, 2000; Johnson & Khoshgoftaar, 2019). We therefore propose to treat each client's local training as a class-imbalanced learning problem and leverage techniques developed in this sub-field. Re-weighting and re-sampling (Buda et al., 2018) are the most fundamental techniques. Denote by $N_{m,c}$ the number of training instances of class c for client m, these techniques adjust $\mathcal{L}_m(\boldsymbol{w})$ in Equation 1 into

$$\mathcal{L}_m^{BR}(\boldsymbol{w}) \propto \sum_i q_{y_i} \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w}), \quad \text{where } q_{y_i} \text{ is usually set as } \frac{1}{N_{m,y_i}} \text{ or } \frac{1}{\sqrt{N_{m,y_i}}}. \tag{5}$$

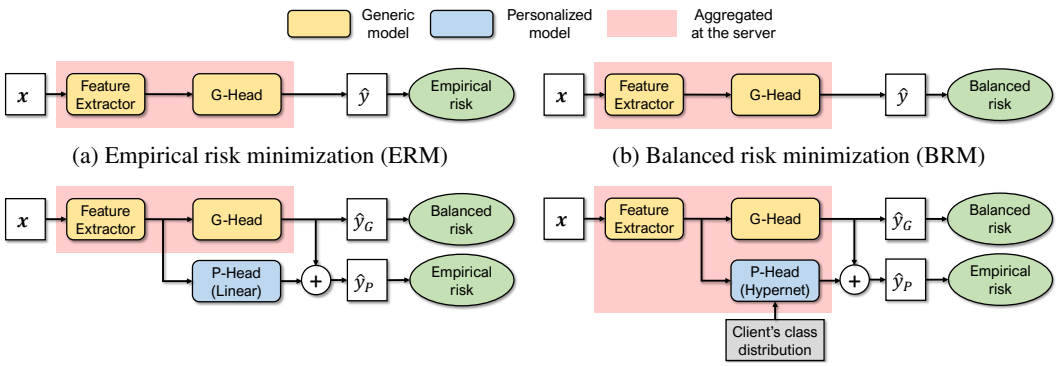
Namely, they mitigate the influence of $\mathcal{P}_m(y)$ by turning the empirical risk $\mathcal{L}_m$ into a *balanced risk* $\mathcal{L}_m^{BR}$, such that every client solves a more consistent objective that is robust to the class distributions. Recently, many class-imbalanced works proposed to replace the instance loss ℓ (*e.g.*, cross entropy) with a class-balanced loss (Cao et al., 2019; Kang et al., 2020; Khan et al., 2017; Ren et al., 2020; Ye et al., 2020), showing more promising results than re-weighting or re-sampling. We can also define $\mathcal{L}_m^{BR}$ using these losses, *e.g.*, the balanced softmax (BSM) loss (Ren et al., 2020)

$$\mathcal{L}_m^{BR}(\boldsymbol{w}) \propto \sum_i \ell^{\text{BSM}}(\boldsymbol{x}_i, y_i; \boldsymbol{w}), \text{ where } \ell^{\text{BSM}}(\boldsymbol{x}, y; \boldsymbol{w}) = -\log \frac{N_{m,y}^{\gamma} \exp(g_y(\boldsymbol{x}; \boldsymbol{w}))}{\sum_{c \in \mathbb{C}} N_{m,c}^{\gamma} \exp(g_c(\boldsymbol{x}; \boldsymbol{w}))}. \tag{6}$$

Here, $g_c(\boldsymbol{x}; \boldsymbol{w})$ is the logit for class c, $\mathbb{C}$ is the label space, and γ is a hyper-parameter. The BSM loss is an unbiased extension of softmax to accommodate the class distribution shift between training and testing. It encourages a minor-class instance to claim a larger logit $g_y(\boldsymbol{x}; \boldsymbol{w})$ in training to overcome feature deviation (Ye et al., 2020) in testing. We list other class-balanced losses in the appendix.

[2]Clients having different $\mathcal{P}_m(\boldsymbol{x}|y)$ is related to domain adaptation (Gong et al., 2012) and generalization (Muandet et al., 2013), which require knowing the distributions of all/some clients for algorithm design.

[3]This is indeed the main cause of non-IID data distributions in the literature of FL (Hsu et al., 2019; 2020).



(a) Empirical risk minimization (ERM)

(b) Balanced risk minimization (BRM)

(c) FED-ROD (linear)

(d) FED-ROD (hyper)

Figure 4: **Comparison of local training strategies and model architectures.** G: generic; P: personalized. Yellow boxes correspond to the models for G-FL; green boxes, for P-FL. Boxes that are covered by the red background are sent back to the server for aggregation (*e.g.*, weight average), and re-initialized at the next round. Green ellipsoids correspond to the learning objectives. $\hat{y}$ stands for the predicted logits (of all classes); $\hat{y}_G$ and $\hat{y}_P$ come from the G-head and P-head, respectively. **(a)** local training with ERM; **(b)** local training with BRM; **(c) FED-ROD (linear):** learning with both BRM (for G-FL) and ERM (for P-FL) using the two-predictor (head) architecture; **(d) FED-ROD (hyper):** same as (c), but the P-head is constructed by a shared hypernetwork.

We take advantage of these existing efforts by replacing the empirical risk $\mathcal{L}_m$ in Equation 2 with a balanced risk $\mathcal{L}_m^{BR}$, which either takes the form of Equation 5 or applies a class-balanced loss (*e.g.*, Equation 6), or both. We note that, a variant of Equation 5 has been used in (Hsu et al., 2020). However, our experiments show that it is less effective than class-balanced losses in extreme non-IID cases. Interestingly, we found that $\mathcal{L}_m^{BR}$ can easily be incorporated into advanced FL algorithms like FEDDYN (Acar et al., 2021), because these algorithms are agnostic to the local objectives being used.

4.2 LOCAL TRAINING AND LOCAL MODEL DECOUPLING WITH ERM AND BRM

The use of balanced risk $\mathcal{L}_m^{BR}$ in local training notably improves the resulting global model $\bar{\boldsymbol{w}}$'s generic performance, as will be seen in section 5. Nevertheless, it inevitably hurts the local model $\boldsymbol{w}_m$'s personalized performance, since it is no longer optimized towards client's empirical risk $\mathcal{L}_m$.

To address these contrasting pursuits of generic and personalized FL, we propose a unifying FL framework named **Federated Robust Decoupling (FED-ROD)**, which *decouples the dual duties of local models* by learning two predictors on top of a shared feature extractor: one trained with empirical risk minimization (ERM) for personalized FL (P-FL) and the other with balanced risk minimization (BRM) for generic FL (G-FL). Figure 4 (c-d) illustrates the model and local training objective of FED-ROD. The overall training process of FED-ROD follows FEDAVG, iterating between local training and global aggregation. As mentioned in subsection 4.1, other generic FL algorithms (Acar et al., 2021; Karimireddy et al., 2020b; Li et al., 2020a) can easily be applied to the BRM branch to further improve the generic performance. Without loss of generality, we focus on the basic version built upon FEDAVG. We start with the model in Figure 4 (c).

Notations. We denote by $f(\boldsymbol{x};\boldsymbol{\theta})$ the shared feature extractor parameterized by $\boldsymbol{\theta}$, whose output is $\boldsymbol{z}$. We denote by $h^G(\boldsymbol{z};\boldsymbol{\psi})$ and $h^P(\boldsymbol{z};\boldsymbol{\phi}_m)$ the generic and personalized prediction heads parameterized by $\boldsymbol{\psi}$ and $\boldsymbol{\phi}_m$, respectively; both are fully-connected (FC) layers. In short, our generic model is parameterized by $\{\boldsymbol{\theta},\boldsymbol{\psi}\}$; our personalized model for client m is parameterized by $\{\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\phi}_m\}$.

Predictions. For generic prediction, we perform $\boldsymbol{z} = f(\boldsymbol{x};\boldsymbol{\theta})$, followed by $\hat{y}_G = h^G(\boldsymbol{z};\boldsymbol{\psi})$. For personalized prediction, we perform $f(\boldsymbol{x};\boldsymbol{\theta})$, followed by $\hat{y}_P = h^G(\boldsymbol{z};\boldsymbol{\psi}) + h^P(\boldsymbol{z};\boldsymbol{\phi}_m)$. That is, h^P is an add-on to h^G, providing personalized information that is not captured by the generic head.

The overall objective. FED-ROD learns the generic model with the balanced risk $\mathcal{L}_m^{BR}$ and the personalized predictor with the empirical risk $\mathcal{L}_m$. That is, different from Equation 1, FED-ROD aims to solve the following two optimization problems *simultaneously*

$$\min_{\boldsymbol{\theta},\boldsymbol{\psi}} \mathcal{L}(\{\boldsymbol{\theta},\boldsymbol{\psi}\}) = \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \mathcal{L}_m^{\text{BR}}(\{\boldsymbol{\theta},\boldsymbol{\psi}\}) \quad \text{and} \quad \min_{\boldsymbol{\phi}_m} \mathcal{L}_m(\{\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\phi}_m\}), \forall m \in [M]. \tag{7}$$

We note that, $\mathcal{L}_m$ is only used to learn the personalized head parameterized by $\boldsymbol{\phi}_m$.

Learning. Equation 7 cannot be solved directly in federated learning, so FED-ROD follows FEDAVG to learn iteratively between the local training and global aggregation steps

$$\textbf{Local:}\quad \boldsymbol{\theta}_m^\star, \boldsymbol{\psi}_m^\star = \arg\min_{\boldsymbol{\theta},\boldsymbol{\psi}} \mathcal{L}_m^{\text{BR}}(\{\boldsymbol{\theta},\boldsymbol{\psi}\}), \quad \text{initialized with } \bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\psi}}, \tag{8}$$

$$\boldsymbol{\phi}_m^\star = \arg\min_{\boldsymbol{\phi}_m} \mathcal{L}_m(\{\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\phi}_m\}), \quad \text{initialized with } \boldsymbol{\phi}_m', \tag{9}$$

$$\textbf{Global:}\quad \bar{\boldsymbol{\theta}} \leftarrow \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|}\boldsymbol{\theta}_m^\star, \quad \bar{\boldsymbol{\psi}} \leftarrow \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|}\boldsymbol{\psi}_m^\star, \tag{10}$$

where $\boldsymbol{\phi}_m'$ is learned from the previous round, similar to $\boldsymbol{w}_m^{(t)}$ in Equation 4. That is, the personalized head will not be averaged globally but kept locally. In our implementation, Equation 8 and Equation 9 are solved simultaneously via SGD, and we do not derive gradients w.r.t. $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ from $\mathcal{L}_m(\{\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\phi}_m\})$. The $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ in Equation 9 thus come dynamically from the SGD updates of Equation 8. In other words, Equation 9 is not merely fine-tuning on top of the generic model. In the end of federated learning, we will obtain $\bar{\boldsymbol{\theta}}$ and $\bar{\boldsymbol{\psi}}$ (Equation 10) for generic predictions and $\{\boldsymbol{\theta}_m^\star, \boldsymbol{\psi}_m^\star, \boldsymbol{\phi}_m^\star\}_{m=1}^M$ (Equation 8 and Equation 9) for personalized predictions, respectively. Please be referred to the appendix for the pseudocode.

4.3 ADAPTIVE PERSONALIZED PREDICTORS VIA HYPERNETWORKS

In subsection 4.2, the parameter $\boldsymbol{\phi}_m$ of the personalized predictor is learned independently for each client and never shared across clients. In other words, for a new client not involved in the training phase, FED-ROD can only offer the global model for generic prediction. In this subsection, we investigate learning a shared personalized predictor that can adapt to new clients. Concretely, we propose to learn a *meta-model* which can generate $\boldsymbol{\phi}_m$ for a client given the client's class distribution. We denote by $H^P(\boldsymbol{a}_m; \boldsymbol{\nu})$ the meta-model parameterized by $\boldsymbol{\nu}$, whose output is $\boldsymbol{\phi}_m$. Here, $\boldsymbol{a}_m \in \mathbb{R}^{|\mathbb{C}|}$ is the $|\mathbb{C}|$-dimensional vector that records the class distribution of client m; *i.e.*, the c-th dimension $a_m[c] = \frac{N_{m,c}}{\sum_{c'} N_{m,c'}}$. Accordingly, the local training step of $\boldsymbol{\phi}_m$ in Equation 9 is replaced by

$$\textbf{Local:}\quad \boldsymbol{\nu}_m^\star = \arg\min_{\boldsymbol{\nu}} \mathcal{L}_m(\{\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\nu}\}), \text{ initialized with } \bar{\boldsymbol{\nu}}; \quad \textbf{Global:}\quad \bar{\boldsymbol{\nu}} \leftarrow \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|}\boldsymbol{\nu}_m^\star. \tag{11}$$

We implement H^P by a lightweight hypernetwork (Ha et al., 2017) with two fully-connected layers. With the learned $\bar{\boldsymbol{\nu}}$, the meta-model H^P can locally generate $\boldsymbol{\phi}_m$ based on $\boldsymbol{a}_m$, making it adaptive to new clients simply by class distributions. The parameter $\boldsymbol{\phi}_m$ can be further updated using clients' data. **We name this version FED-ROD (hyper); the previous one, FED-ROD (linear).** Please see Figure 4 (c-d) for an illustration. We include more details in the appendix.

5 EXPERIMENT (MORE DETAILS AND RESULTS IN THE APPENDIX)

Datasets, models, and settings. We use CIFAR-10/100 (Krizhevsky et al., 2009) and Fashion-MNIST (FMNIST) (Xiao et al., 2017). We also include a realistic EMNIST (Cohen et al., 2017) dataset, which collects hand-written letters of thousands of writers. To simulate the non-IID data distributions on CIFAR and FMNIST, we follow Hsu et al. (2019) to create a heterogeneous partition for M clients: an M-dimensional vector $\boldsymbol{q}_c$ is drawn from $\text{Dir}(\alpha)$ for class c, and we assign data of class c to client m proportionally to $\boldsymbol{q}_c[m]$. The resulting clients have different numbers of total images and different class distributions. With $\alpha < 1$, most of the training examples of one class are likely assigned to a small portion of clients. Similar to Lin et al. (2020), we use $M = 100$ clients for FMNIST and $M = 20$ for CIFAR-10/100, and sample $20\%/40\%$ clients at every round, respectively. For EMNIST, we use the digit images, follow Caldas et al. (2018) to construct $2,185$ clients (each is a writer), and sample 5% clients at every round. We use a ConvNet (LeCun et al., 1998) similar to (Acar et al., 2021; McMahan et al., 2017). It contains 3 convolutional layers and 2 fully-connected layers. We train every FL algorithm for 100 rounds, with 5 local epochs in each round.

We report the mean accuracy of **five times** of experiments with different random seeds. We evaluate the generic performance (G-FL) using the **generic model (GM)** on the standard generic test set. For FMNIST and CIFAR-10/100, we evaluate the personalized performance (P-FL) using **personalized models (PM)** on the same set, but re-weight the accuracy according to clients' class distributions

Table 1: Results in G-FL accuracy and P-FL accuracy (%). $\star$: methods with no G-FL models and we combine their P-FL models. §: official implementation. Blue/**bold** fonts highlight the best baseline/our approach.

Dataset	EMNIST			FMNIST						CIFAR-10						CIFAR-100					
Non-IID	Writers			Dir(0.1)			Dir(0.3)			Dir(0.1)			Dir(0.3)			Dir(0.1)			Dir(0.3)		
Test Set	G-FL	P-FL		G-FL	P-FL		G-FL	P-FL		G-FL	P-FL		G-FL	P-FL		G-FL	P-FL		G-FL	P-FL	
Method / Model	GM	GM	PM	GM	GM	PM	GM	GM	PM	GM	GM	PM	GM	GM	PM	GM	GM	PM	GM	GM	PM
FedAvg	97.0	96.9	97.2	81.1	81.0	91.5	83.4	83.2	90.5	57.6	57.1	90.5	68.6	69.4	85.1	41.8	41.6	70.2	46.4	46.2	61.7
FedProx	97.0	97.0	97.0	82.2	82.3	91.4	84.5	84.5	89.7	58.7	58.9	89.7	69.9	69.8	84.7	41.7	41.6	70.4	46.5	46.4	61.5
SCAFFOLD	97.1	97.0	97.1	83.1	83.0	89.0	85.1	85.0	90.4	61.2	60.8	90.1	71.1	71.5	84.8	42.3	42.1	70.4	46.5	46.5	61.7
FedDyn §	97.3	97.3	97.3	83.2	83.2	90.7	86.1	86.1	91.5	63.4	63.9	92.4	72.5	73.2	85.4	43.0	43.0	72.0	47.5	47.4	62.5
MTL $\star$	75.4	75.0	85.6	36.1	36.0	87.3	53.1	53.4	78.3	12.1	12.7	90.6	13.5	13.7	80.2	9.5	9.3	60.7	10.8	10.7	49.9
LG-FedAvg $\star$§	80.1	80.0	95.6	54.8	54.5	89.5	66.8	66.8	84.4	29.5	28.8	90.8	46.7	46.2	82.4	23.5	23.4	66.7	34.5	33.9	55.4
FedPer $\star$	93.3	93.1	97.2	74.5	74.4	91.3	79.9	79.9	90.4	50.4	50.2	89.9	64.4	64.5	84.9	37.6	37.6	71.0	40.3	40.1	62.5
Per-FedAvg	95.1	-	97.0	80.5	-	82.8	84.1	-	86.7	60.7	-	82.7	70.5	-	80.7	39.0	-	66.6	44.5	-	58.9
pFedMe §	96.3	96.0	97.1	76.7	76.7	83.4	79.0	79.0	83.4	50.6	50.7	76.6	62.1	61.7	70.5	38.6	38.5	63.0	41.4	41.1	53.4
Ditto	97.0	97.0	97.4	81.5	81.5	89.4	83.3	83.2	90.1	58.1	58.3	86.8	69.7	69.8	81.5	41.7	41.8	68.5	46.4	46.4	58.8
FedFOMO $\star$	80.5	80.4	95.9	34.5	34.3	90.0	70.1	69.9	89.6	30.5	31.2	90.5	45.3	45.1	83.4	35.4	35.3	68.9	39.6	39.3	58.4
FedRep $\star$§	95.0	95.1	97.5	79.5	80.1	91.8	80.6	80.5	90.5	56.6	56.2	91.0	67.7	67.5	85.2	40.7	40.7	71.5	46.0	46.0	62.1
Local only	-	-	64.2	-	-	85.9	-	-	85.0	-	-	87.4	-	-	75.7	-	-	40.0	-	-	32.5
Fed-RoD (linear)	**97.3**	**97.3**	**97.5**	**83.9**	**83.9**	**92.7**	**86.3**	**86.3**	**94.5**	**68.5**	**68.5**	**92.7**	**76.9**	**76.8**	**86.4**	**45.9**	**45.8**	**72.2**	**48.5**	**48.5**	**62.3**
Fed-RoD (hyper)	**97.3**	**97.3**	**97.5**	**83.9**	**83.9**	**92.9**	**86.3**	**86.3**	**94.8**	**68.5**	**68.5**	**92.5**	**76.9**	**76.8**	**86.8**	**45.9**	**45.8**	**72.3**	**48.5**	**48.5**	**62.5**
+ FedDyn	**97.4**	**97.4**	**97.5**	**85.9**	**85.7**	**95.3**	**87.5**	**87.5**	**94.6**	**68.2**	**68.2**	**92.7**	**74.6**	**74.6**	**85.6**	**46.2**	**46.2**	**72.5**	**48.4**	**48.4**	**62.5**

$P_m(y)$ and average the weighted accuracy across M clients as $\frac{1}{M}\sum_m \frac{\sum_i \mathcal{P}_m(y_i)\mathbf{1}(y_i=\hat{y}_i)}{\sum_i \mathcal{P}_m(y_i)}$. Here, i is the instance index. This evaluation is more robust (essentially as the expectation) than assigning each client a specific test set. For EMNIST, each client has its own test set with the same writing style.

Our variants. We mainly use Equation 6 with $\gamma = 1$ as the $\mathcal{L}_m^{BR}$ and report the Fed-RoD (hyper) version (cf. subsection 4.3). Table 2 provides the ablation study.

Baselines. For G-FL methods including FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020a), SCAFFOLD (Karimireddy et al., 2020b), and FedDyn (Acar et al., 2021), we use their global models $\bar{\boldsymbol{w}}$ for G-FL evaluation; their local models (*i.e.*, $\boldsymbol{w}_m$ in Figure 1) for P-FL evaluation. For P-FL methods, to evaluate their G-FL performance, we use the available global models in pFedMe (Dinh et al., 2020) and Ditto (Li et al., 2021a) or average the final personalized models for MTL (Smith et al., 2017), FedPer (Arivazhagan et al., 2019), LG-FedAvg (Liang et al., 2020), FedFOMO (Zhang et al., 2021), and FedRep (Collins et al., 2021).

To illustrate the difference between applying GMs and PMs in a P-FL setting, we also evaluate the P-FL performance using GMs, which is how FedAvg *has been applied to P-FL in literature.*

5.1 Results

Fed-RoD bridges G-FL and P-FL and consistently outperforms all generic and personalized FL methods. Table 1 summarizes the results. In terms of **G-FL accuracy**, advanced local training (*i.e.*, SCAFFOLD, FedProx, and FedDyn) outperforms FedAvg and personalized methods, and our Fed-RoD can have further gains by using balanced risk minimization (BRM). We also investigate combining Fed-RoD and FedDyn (Acar et al., 2021), using the latter to optimize the generic model with BRM, which outperforms either ingredient in many cases. We report the G-FL accuracy of personalized FL algorithms mainly to investigate if they have similar properties like FedAvg: an algorithm designed for one setup can also construct models for the other setup.

In terms of **P-FL accuracy**, by using PMs most methods outperform the baseline of local training with individual client's data without communication (*i.e.*, local only), justifying the benefits of federated collaboration[4]. For generic FL methods, using PMs (*i.e.*, local models $\{\boldsymbol{w}_m\}$) clearly outperforms using GMs (*i.e.*, $\bar{\boldsymbol{w}}$), which supports our claims and observations in Figure 1 and subsection 3.2. *It is worth noting that the local models from generic FL methods are highly competitive to or even outperform personalized models produced by personalized FL methods.* This provides generic FL methods with an add-on functionality to output personalized models by keeping the checkpoints on clients after local training. **Our Fed-RoD achieves the highest P-FL accuracy** and we attribute this to (a) the shared feature extractor learned with the balanced risk and re-initialized every round to benefit from implicit regularization; (b) the personalized head learned with clients' empirical risks.

[4]PMs of "local only" could outperform the GM of FedAvg on the P-FL accuracy, especially when the non-IID condition becomes severe (*e.g.*, Dir (0.1)): it is hard to train a single GM to perform well in P-FL.

Table 2: Ablation study on variants of FED-ROD. FT: fine-tuning

Test Set	|G-FL|	P-FL	
Method / Model	| GM	GM	PM
Centralized	| 85.4	85.4	-
FEDAVG	68.6	69.4	85.1
FEDAVG (BRM)	76.8	76.7	76.1
FEDAVG (BRM, FT)	76.8	76.7	84.5
FED-ROD (linear, BRM)	76.9	76.8	86.4
FED-ROD (hyper, BRM)	76.9	76.8	86.8

Table 3: FED-ROD with different balanced losses. ⋆: BSM

Test Set	|G-FL|	P-FL	
Loss / Model	| GM	GM	PM
Cross entropy	68.6	69.4	85.1
(Hsu et al., 2020)	65.8	65.8	80.1
(Cao et al., 2019)	75.7	75.9	83.3
(Ye et al., 2020)	75.2	75.0	85.1
(Ren et al., 2020)⋆	76.9	76.8	86.8

Table 4: P-FL accuracy on future non-IID clients (Dir(0.3) for all datasets). Each cell is before/after local fine-tuning.

Method	|FMNIST	CIFAR-10	CIFAR-100
FEDAVG	80.3/87.2	56.2/76.0	38.9/54.3
FEDDYN	82.3/87.6	61.7/76.2	40.1/57.2
PER-FEDAVG	82.1/89.6	60.0/79.8	37.6/55.6
FED-ROD (linear)	83.5/91.3	62.4/80.2	40.0/58.2
FED-ROD (hyper)	88.9/91.4	75.7/81.5	40.7/59.0
+FEDDYN	89.2/91.3	77.1/83.5	41.4/59.5

BRM effectively reduces the variance of G-FL accuracy and local gradients. To understand why FED-ROD improves G-FL, we visualize the global model $\bar{\boldsymbol{w}}$'s and each local model $\boldsymbol{w}_m$'s G-FL accuracy on CIFAR-10 (Dir(0.3)), in Figure 5 (upper). FED-ROD not only learns a better global model for G-FL, but also has a smaller variance of accuracy across the local models' generic heads (as their objectives are more aligned). We also show how $\boldsymbol{w}_m$ deviates from $\bar{\boldsymbol{w}}$ after local training in Figure 5 (lower). FED-ROD has a smaller variance. This coincides with the study in (Kong et al., 2021): lower variances of the local gradients could imply better generic performance.

FED-ROD benefits from decoupling. We compare several variants of FED-ROD (cf. Figure 4), with one head (reduced to FEDAVG) or different networks (linear/hyper). We evaluate on CIFAR-10 (Dir(0.3)). As shown in Table 2, FEDAVG *with BRM significantly improves G-FL but degrades in P-FL.* FED-ROD remedies this by training a decoupled personalized head. We note that, FED-ROD does not merely fine-tune the global model with clients' data (cf. subsection 4.2). We also compare different balanced losses in Table 3: advanced losses outperforms importance re-weighting (Hsu et al., 2020).

FED-ROD (hyper) benefits future clients. To validate the generalizability to new clients, we build on the Dir(0.3) non-IID setting for FMNIST and CIFAR-10/100, but split the training data into 100 clients (50 are in training; 50 are new). We train on the 50 training clients for 100 rounds (sampling 20 of them every round). We then evaluate on the 50 new clients individually, either using the global model directly or fine-tuning it with clients' data for several steps. Table 4 and Figure 6 shows the averaged accuracy on new clients. Without fine-tuning, FED-ROD (hyper) can already generate personalized models, and outperforms others methods stably with fine-tuning.

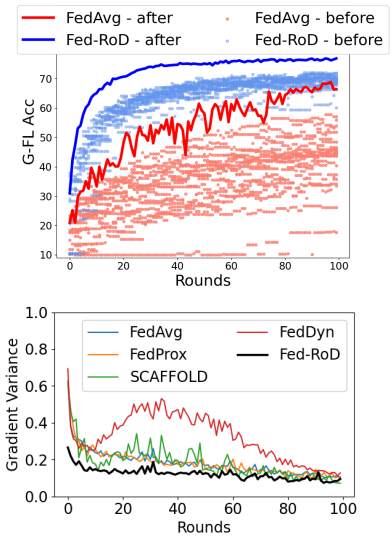


Figure 5: **Upper:** G-FL test accuracy along the training rounds before/after averaging the local models. **Lower:** variances of $\boldsymbol{w}_m - \bar{\boldsymbol{w}}$ across clients.

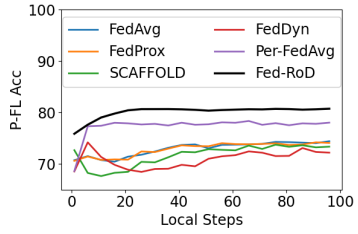


Figure 6: The average P-FL accuracy on future clients, with local training.

More results and analyses in the Appendix. The Appendix includes studies with class-imbalanced global distributions and we show that FED-ROD still performs well. BRM can be further improved with meta-learned hyper-parameters. We validate that re-initializing the local models by the global model at every round (*i.e.*, Equation 2) does lead to a much smaller regularization loss than Equation 4 to support our claim in subsection 3.3. More comprehensive results regarding more clients, deeper backbones, compatibility with other methods, robustness against adversaries, etc, are also provided.

6 CONCLUSION

Most of the existing work in federated learning (FL) has been dedicated to either learning a better generic model or personalized models. We show that these two contrasting goals can be achieved simultaneously via a novel two-loss, two-predictor FL framework FED-ROD. Concretely, we show that strong personalized models emerge from the local training of generic FL algorithms, due to implicit regularization; imposing class-balanced objectives further improves the generic FL accuracy when clients have non-IID distributions. FED-ROD seamlessly incorporates these two observations to excel in both FL settings, and further enables fast adaptation to new clients via an adaptive module.

ACKNOWLEDGMENTS

This research is partially supported by NSF IIS-2107077, NSF OAC-2118240, NSF OAC-2112606, and the OSU GI Development funds. We are thankful for the generous support of the computational resources by the Ohio Supercomputer Center and AWS Cloud Credits for Research.

REPRODUCIBILITY STATEMENT

We report the results with the average over 5 runs of different random seeds. We exhaustively provide the information about the hyperparameters, datasets, evaluation, and other details in section 5 and Appendix C, which should be comprehensive for reproducibility. We also provide our code in https://github.com/hongyouc/Fed-RoD.

REFERENCES

Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *ICLR*, 2021. 2, 3, 5, 6, 7, 8, 16, 17, 18, 23, 24, 29, 30

Alekh Agarwal, John Langford, and Chen-Yu Wei. Federated residual learning. *arXiv preprint arXiv:2003.12880*, 2020. 17

Kartik Ahuja, Karthikeyan Shanmugam, Kush Varshney, and Amit Dhurandhar. Invariant risk minimization games. In *ICML*, 2020. 17, 22, 23

Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019. 1, 3, 8, 17, 32

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. 17, 22

Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, H Brendan McMahan, et al. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*, 2019. 16

Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018. 5

Duc Bui, Kshitiz Malik, Jack Goetz, Honglei Liu, Seungwhan Moon, Anuj Kumar, and Kang G Shin. Federated user representation learning. *arXiv preprint arXiv:1909.12535*, 2019. 3, 17

Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018. 7

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Conference on Neural Information Processing Systems*, 2019. 2, 5, 9, 18, 20, 21, 28

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997. 3

Soravit Changpinyo, Wei-Lun Chao, Boqing Gong, and Fei Sha. Synthesized classifiers for zero-shot learning. In *CVPR*, 2016. 18

Soravit Changpinyo, Wei-Lun Chao, and Fei Sha. Predicting visual exemplars of unseen classes for zero-shot learning. In *ICCV*, 2017. 18

Soravit Changpinyo, Wei-Lun Chao, Boqing Gong, and Fei Sha. Classifier and exemplar synthesis for zero-shot learning. *IJCV*, 128(1):166–201, 2020. 18

Fei Chen, Mi Luo, Zhenhua Dong, Zhenguo Li, and X. He. Federated meta-learning with fast convergence and efficient communication. *arXiv: Learning*, 2018. 17

Hong-You Chen and Wei-Lun Chao. Fedbe: Making bayesian model ensemble applicable to federated learning. In *ICLR*, 2021. 3, 16, 18

Gary Cheng, Karan Chadha, and John Duchi. Fine-tuning is fine in federated learning. *arXiv preprint arXiv:2108.07313*, 2021. 3

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *IJCNN*, 2017. 7

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *ICML*, 2021. 3, 8, 17

Luca Corinzia and Joachim M Buhmann. Variational federated multi-task learning. *arXiv preprint arXiv:1906.06268*, 2019. 17

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, 2019. 2, 20

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020a. 17

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Distributionally robust federated averaging. *NeurIPS*, 33, 2020b. 17

Canh T Dinh, Nguyen H Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. In *NeurIPS*, 2020. 1, 2, 3, 4, 8, 17, 24, 29

Moming Duan, Duo Liu, Xianzhang Chen, Renping Liu, Yujuan Tan, and Liang Liang. Self-balancing federated learning with global imbalanced data in mobile systems. *IEEE Transactions on Parallel and Distributed Systems*, 32(1):59–71, 2020. 17

An Evgeniou and Massimiliano Pontil. Multi-task feature learning. In *NeurIPS*, 2007. 17

Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *KDD*, 2004. 17

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. In *NeurIPS*, 2020. 1, 17, 23, 29, 31

Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*, 2015. 17

Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 2066–2073. IEEE, 2012. 5

Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 18

David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In *ICLR*, 2017. 2, 3, 7, 17, 18, 22

Farzin Haddadpour and Mehrdad Mahdavi. On the convergence of local descent methods in federated learning. *arXiv preprint arXiv:1910.14425*, 2019. 16

Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*, 2020. 4, 17

Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtárik. Lower bounds and optimal algorithms for personalized federated learning. In *NeurIPS*, 2020. 1, 3, 4, 17

Chaoyang He, Murali Annavaram, and Salman Avestimehr. Group knowledge transfer: Federated learning of large cnns at the edge. In *NeurIPS*, 2020. 16, 18

Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009. 2, 3, 5, 17

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 31

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019. 3, 5, 7, 16, 24

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Federated visual classification with real-world data distribution. In *ECCV*, 2020. 3, 5, 6, 9, 17, 21

Yutao Huang, Lingyang Chu, Z. Zhou, Lanjun Wang, J. Liu, Jian Pei, and Yanxin Zhang. Personalized cross-silo federated learning on non-iid data. In *AAAI*, 2021. 17

Laurent Jacob, Francis Bach, and Jean-Philippe Vert. Clustered multi-task learning: A convex formulation. In *NeurIPS*, 2009. 17

Nathalie Japkowicz. The class imbalance problem: Significance and strategies. In *Proc. of the Int'l Conf. on Artificial Intelligence*, 2000. 5

Yihan Jiang, Jakub Konecný, Keith Rush, and S. Kannan. Improving federated learning personalization via model agnostic meta learning. *ArXiv*, abs/1909.12488, 2019. 17

Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):27, 2019. 5

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019. 1

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *ICLR*, 2020. 5, 20

Sai Praneeth Karimireddy, Martin Jaggi, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2020a. 3, 16, 17

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *ICML*, 2020b. 5, 6, 8, 16, 18, 29, 30

A Khaled, K Mishchenko, and P Richtárik. Tighter theory for local sgd on identical and heterogeneous data. In *AISTATS*, 2020. 16

Salman H Khan, Munawar Hayat, Mohammed Bennamoun, Ferdous A Sohel, and Roberto Togneri. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE transactions on neural networks and learning systems*, 29(8):3573–3587, 2017. 5, 20

M. Khodak, Maria-Florina Balcan, and Ameet Talwalkar. Adaptive gradient-based meta-learning methods. In *NeurIPS*, 2019. 17

Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016. 16

Lingjing Kong, Tao Lin, Anastasia Koloskova, Martin Jaggi, and Sebastian U Stich. Consensus control for decentralized deep learning. In *ICML*, 2021. 9

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 7

V. Kulkarni, Milind Kulkarni, and A. Pant. Survey of personalization techniques for federated learning. *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pp. 794–797, 2020. 3, 17

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *TPAMI*, 36(3):453–465, 2013. 18

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 7, 23, 31

Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019. 4, 17

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smithy. Feddane: A federated newton-type method. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, 2019. 16

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *MLSys*, 2020a. 3, 5, 6, 8, 16, 18, 23, 29, 30

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through. In *ICML*, 2021a. 1, 3, 4, 8, 17, 26, 29, 30

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *ICLR*, 2020b. 1, 5, 16, 23

Xiaoxiao Li, Meirui JIANG, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fed{bn}: Federated learning on non-{iid} features via local batch normalization. In *ICLR*, 2021b. 17

Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *CVPR*, 2020c. 18

Paul Pu Liang, Terrance Liu, Liu Ziyin, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020. 1, 3, 8, 17, 29

Xianfeng Liang, Shuheng Shen, Jingchang Liu, Zhen Pan, Enhong Chen, and Yifei Cheng. Variance reduced local sgd with lower communication complexity. *arXiv preprint arXiv:1912.12844*, 2019. 16

Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. In *NeurIPS*, 2020. 3, 7, 16, 18, 24

Grigory Malinovskiy, Dmitry Kovalev, Elnur Gasanov, Laurent Condat, and Peter Richtarik. From local sgd to local fixed-point methods for federated learning. In *ICML*, 2020. 3, 16

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020. 17

H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017. 1, 2, 3, 4, 7, 8, 16, 19, 20, 23, 29, 30, 32

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In *ICML*, 2019. 17

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *ICML*, 2013. 5, 17

Reese Pathak and Martin J Wainwright. Fedsplit: An algorithmic framework for fast federated optimization. In *NeurIPS*, 2020. 16

Daniel Peterson, Pallika Kanani, and Virendra J Marathe. Private federated learning with domain adaptation. *arXiv preprint arXiv:1912.06733*, 2019. 17

Tomaso Poggio, Ryan Rifkin, Sayan Mukherjee, and Alex Rakhlin. Bagging regularizes. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB, 2002. 18

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. In *ICLR*, 2021. 3, 16

Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. *arXiv preprint arXiv:1909.13014*, 2019. 16

Matthias Reisser, Christos Louizos, Efstratios Gavves, and Max Welling. Federated mixture of experts, 2021. URL https://openreview.net/forum?id=YgrdmztE4OY. 17

Jiawei Ren, Cunjun Yu, Shunan Sheng, Xiao Ma, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Balanced meta-softmax for long-tailed visual recognition. In *NeurIPS*, 2020. 2, 5, 9, 18, 20, 21, 22

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *ICML*, 2018. 22

Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017. 3, 17

Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. In *ICML*, 2021. 3, 17

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *NeurIPS*, 2019. 22

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 31

Marina Skurichina and Robert PW Duin. Bagging for linear classifiers. *Pattern Recognition*, 31(7):909–930, 1998. 18

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *NeurIPS*, 2017. 1, 2, 3, 4, 8, 17, 29

Sebastian U Stich. Local sgd converges fast and communicates little. In *ICLR*, 2019. 1

Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *CVPR*, 2020. 18

TensorFlow team. Tensorflow convolutional neural networks tutorial. `http://www.tensorflow.org/tutorials/deep_cnn`, 2016. 23

Grant Van Horn and Pietro Perona. The devil is in the tails: Fine-grained classification in the wild. *arXiv preprint arXiv:1709.01450*, 2017. 18

Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *CVPR*, 2018. 18

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *ICLR*, 2020a. 3, 16

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. In *NeurIPS*, 2020b. 3, 16

Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Franccoise Beaufays, and D. Ramage. Federated evaluation of on-device personalization. *ArXiv*, abs/1910.10252, 2019. 3, 17

Lixu Wang, Shichao Xu, Xiao Wang, and Qi Zhu. Addressing class imbalance in federated learning. In *AAAI*, 2020c. 17

Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *TPAMI*, 41(9):2251–2265, 2018. 18

H. Xiao, K. Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *ArXiv*, abs/1708.07747, 2017. 7

Miao Yang, Akitanoshou Wong, Hongbin Zhu, Haifeng Wang, and Hua Qian. Federated learning with class imbalance reduction. *arXiv preprint arXiv:2011.11266*, 2020. 17

Xin Yao, Tianchi Huang, Rui-Xiao Zhang, Ruiyu Li, and Lifeng Sun. Federated learning with unbiased gradient aggregation and controllable meta updating. *arXiv preprint arXiv:1910.08234*, 2019. 16

Han-Jia Ye, Hong-You Chen, De-Chuan Zhan, and Wei-Lun Chao. Identifying and compensating for feature deviation in imbalanced deep learning. *arXiv preprint arXiv:2001.01385*, 2020. 5, 9, 18, 20, 21

Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. Salvaging federated learning by local adaptation. *arXiv preprint arXiv:2002.04758*, 2020. 3, 17

Honglin Yuan and Tengyu Ma. Federated accelerated stochastic gradient descent. In *NeurIPS*, 2020. 3, 16

Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Trong Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *ICML*, 2019. 3, 16

Edvin Listo Zec, Olof Mogren, John Martinsson, Leon René Sütfeld, and Daniel Gillblad. Federated learning using a mixture of experts. *arXiv preprint arXiv:2010.02056*, 2020. 17

Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M. Alvarez. Personalized federated learning with first order model optimization. In *ICLR*, 2021. URL `https://openreview.net/forum?id=ehJqJQk9cw`. 1, 8, 17, 29, 30

Yu Zhang and Qiang Yang. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*, 2017. 3, 17

Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationships in multi-task learning. 2010. 17

Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Facial landmark detection by deep multi-task learning. In *European conference on computer vision*, pp. 94–108. Springer, 2014. 3

Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018. 1, 3, 16, 22, 29, 30

Fan Zhou and Guojing Cong. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. *arXiv preprint arXiv:1708.01012*, 2017. 1

Yanlin Zhou, George Pu, Xiyao Ma, Xiaolin Li, and Dapeng Wu. Distilled one-shot federated learning. *arXiv preprint arXiv:2009.07999*, 2020. 16, 18

Appendix

We provide details omitted in the main paper.

- Appendix A: additional comparison to related work (cf. section 2 and section 3 of the main paper).
- Appendix B: additional details of FED-ROD (cf. section 3 and section 4 of the main paper).
- Appendix C: details of experimental setups (cf. section 5 of the main paper).
- Appendix D: additional experimental results and analysis (cf. section 3 and section 5 of the main paper).

A COMPARISON TO RELATED WORK

A.1 FOLLOW-UP WORKS OF FEDAVG

Several recent works (Karimireddy et al., 2020a;b; Zhao et al., 2018) have shown that, with multiple steps of local SGD updates, the local model $\boldsymbol{w}_m$ would drift away from each other, leading to a degenerated global model $\bar{\boldsymbol{w}}$ that deviates from the solution of Equation 1 of the main text.

One way to mitigate this is to modify the local training objective (cf. Equation 2 of the main text). For instance, FEDPROX (Li et al., 2020a) introduced a regularizer with respect to $\bar{\boldsymbol{w}}$,

$$\min_{\boldsymbol{w}} \mathcal{L}_m(\boldsymbol{w}) + \frac{\lambda}{2}\|\boldsymbol{w} - \bar{\boldsymbol{w}}\|^2.$$

FEDDYN (Acar et al., 2021) further added a dynamic term based on the local model of the previous round $\boldsymbol{w}'_m$,

$$\min_{\boldsymbol{w}} \mathcal{L}_m(\boldsymbol{w}) + \langle \nabla \mathcal{L}_m(\boldsymbol{w}'_m), \boldsymbol{w} \rangle + \frac{\lambda}{2}\|\boldsymbol{w} - \bar{\boldsymbol{w}}\|^2.$$

These regularizers aim to stabilize local training and align the objectives among clients. Some other works did not change the objectives but introduced control variates or momentum to correct the local gradient (Karimireddy et al., 2020a;b), or designed a new optimizer more suitable for decentralized learning (Malinovskiy et al., 2020; Pathak & Wainwright, 2020; Yuan & Ma, 2020).

It is worth mentioning, in most of these works, the empirical risk $\mathcal{L}_m(\boldsymbol{w})$ still plays an important role in driving the local model update. Since $\mathcal{L}_m(\boldsymbol{w})$ directly reflects the (non-IID) client data distribution, the learned local models are indeed strong candidates for personalized models.

A.2 GENERIC FEDERATED LEARNING

FEDAVG (McMahan et al., 2017) is the standard algorithm, which involves multiple rounds of local training and global aggregation. Many works have studied its convergence (Haddadpour & Mahdavi, 2019; Khaled et al., 2020), robustness (Bonawitz et al., 2019), communication (Konečnỳ et al., 2016; Reisizadeh et al., 2019), especially for non-IID clients (Li et al., 2020a;b; Zhao et al., 2018). Many other works proposed to improve FEDAVG. In terms of **global aggregation**, (Wang et al., 2020a; Yurochkin et al., 2019) matched local model weights before averaging. (Chen & Chao, 2021; He et al., 2020; Lin et al., 2020; Zhou et al., 2020) replaced weight average by model ensemble and distillation. (Hsu et al., 2019; Reddi et al., 2021) applied server momentum and adaptive optimization to improve the global model update. In terms of **local training**, (Liang et al., 2019; Malinovskiy et al., 2020; Pathak & Wainwright, 2020; Yuan & Ma, 2020) improved the optimizer. To reduce local models' drifts from the global model, (Zhao et al., 2018) mixed client and server data in local training; FEDPROX (Li et al., 2020a), FEDDANE (Li et al., 2019), and FEDDYN (Acar et al., 2021) employed regularization toward the global model; SCAFFOLD (Karimireddy et al., 2020a) MIME (Karimireddy et al., 2020b) leveraged control varieties and/or server statistics to correct local gradients; (Wang et al., 2020b; Yao et al., 2019) modified the local model update rules. For most of them, the empirical risks on clients' data are the major forces to drive local training.

We also aim to reduce local models' drifts but via a different way. We directly bypass the empirical risks that reflect clients' data distributions. Instead, we apply objective functions in class-imbalanced

learning (He & Garcia, 2009), which are designed to be robust to the change of class distributions. Our approach is different from (Duan et al., 2020; Wang et al., 2020c; Yang et al., 2020), which monitored and resolved class imbalance from the server while we tackled it at the clients. Our approach is also different from agnostic FL (Deng et al., 2020b; Mohri et al., 2019), whose local training is still built on empirical risk minimization. The closest to ours is (Hsu et al., 2020), which used a traditional class-imbalanced treatment, re-weighting, to mitigate non-identical class distributions. We show that more advanced techniques can be applied to further improve the performance, especially under extreme non-IID conditions where re-weighting is less effective. Moreover, our method is compatible with existing efforts like FEDDYN (Acar et al., 2021) and SCAFFOLD (Karimireddy et al., 2020a) to boost the generic performance.

A.3 PERSONALIZED FEDERATED LEARNING

Personalized FL (Kulkarni et al., 2020) learns a customized model for each client. Many approaches are based on multi-task learning (MTL) (Evgeniou & Pontil, 2007; 2004; Jacob et al., 2009; Ruder, 2017; Zhang & Yang, 2017; Zhang & Yeung, 2010) — leveraging the clients' task relatedness to improve model generalizability. For instance, (Smith et al., 2017) encouraged related clients to learn similar models; (Corinzia & Buhmann, 2019; Dinh et al., 2020; Hanzely & Richtárik, 2020; Hanzely et al., 2020; Li & Wang, 2019; Li et al., 2021a) regularized local models with a learnable global model, prior, or set of data logits. (Arivazhagan et al., 2019; Bui et al., 2019; Li et al., 2021b; Liang et al., 2020) designed the model architecture to have both personalized (usually the feature extractor) and shareable components. (Huang et al., 2021; Zhang et al., 2021) constructed for each client an initialized model or regularizer based on learnable bases. Our approach is inspired by MTL as well but has several notable differences from existing works. First, we found that the global aggregation step in generic FL already serves as a strong regularizer. Second, instead of learning for each client a personalized feature extractor (Bui et al., 2019; Liang et al., 2020) or an entire independent model that can operate alone (Dinh et al., 2020; Hanzely et al., 2020; Smith et al., 2017), FED-ROD shares a single feature extractor among all clients, inspired by invariant risk minimization (Ahuja et al., 2020; Arjovsky et al., 2019) and domain generalization (Ghifary et al., 2015; Muandet et al., 2013). This reduces the total parameters to be learned and improves model's generalizability. Compared to FEDPER (Arivazhagan et al., 2019) and FEDREP (Collins et al., 2021) which also learned a shared feature extractor, FED-ROD simultaneously outputs a single, strong global model to excel in the generic FL setup.

Some other approaches are based on mixture models. (Agarwal et al., 2020; Deng et al., 2020a; Mansour et al., 2020; Peterson et al., 2019; Zec et al., 2020) (separately) learned global and personalized models and performed a mixture of them for prediction. (Reisser et al., 2021) learned a sets of expert models and used them to construct personalized models. Meta-learning is also applied to learn a good initialized model that can be adapted to each client with a few steps of local training (Chen et al., 2018; Fallah et al., 2020; Jiang et al., 2019; Khodak et al., 2019).

Instead of designing specific algorithms for personalized FL, (Wang et al., 2019; Yu et al., 2020) showed that performing post-processing (*e.g.*, fine-tuning) to a generic FL model (*e.g.*, $\bar{\boldsymbol{w}}$ learned by FEDAVG) already leads to promising personalized accuracy. In this work, we further showed that, the *local models* $\boldsymbol{w}_m$ learned in FEDAVG and other generic FL algorithms are indeed strong personalized models.

We note that, while many personalized FL algorithms also produce a global model, it is mainly used to regularize or construct personalized models but not for evaluation in the generic setup. In contrast, we learn models to excel in both the setups via a single framework without sacrificing either of them.

PFEDHN (Shamsian et al., 2021) also applies hypernetworks (Ha et al., 2017) but for a very different purpose from FED-ROD. Specifically, PFEDHN learns a hypernetwork at the server to aggregate clients' model updates and produce their entire models for the next round. In contrast, we learn the hypernetwork locally to construct the personalized predictors, not the entire models, for fast adaptation to clients.

A.4 AVERAGING MODEL WEIGHTS AS A REGULARIZER

In subsection 3.3, we demonstrate that taking the average over model weights indeed acts as a regularizer for *local models* to improve their individual *personalized* performance.

In more traditional machine learning, the regularization effects of averaging multiple independently-trained models have been observed in some techniques like bagging (Poggio et al., 2002; Skurichina & Duin, 1998). Indeed, in several recent works of FL (Chen & Chao, 2021; He et al., 2020; Lin et al., 2020; Zhou et al., 2020), the authors replaced weight average by bagging/model ensemble to improve the *generic* performance on the global test set. That is, they found that performing the model ensemble over clients' models can yield more robust predictions on the global test set than the global model, which is generated by averaging the client models' weights.

Here, we however study a different regularization effect, in personalized FL on local test sets. As reviewed in subsection 3.1, personalized FL algorithms often impose a regularizer on the local/personalized models to overcome the fact that clients usually have limited data (please see Equation 3 and Equation 4 and the surrounding text). What we claim is that even without such an explicit regularizer, the model weight average before local training (Equation 2) already serves as an implicit regularizer to the local models for their individual personalized performance, as we discussed in subsection 3.3 (Figure 3) and empirically verified in subsection D.2 and Figure 7.

A.5 SYSTEMATIC OVERHEAD

FED-ROD has similar computation cost, communication size, and number of parameters as FEDAVG. We discuss the difference between FED-ROD and existing generic FL methods from a system view. FEDPROX (Li et al., 2020a) proposes a proximal term to prevent client from diverging from the server model, which is more robust to the heterogeneous system. SCAFFOLD (Karimireddy et al., 2020b) imposes a gradient correction during client training. Maintaining such a correction term, however, doubles the size of communication. FEDDYN (Acar et al., 2021) resolves the communication cost issue by introducing a novel dynamic regularization. However, it requires all users to maintain their previous models locally throughout the FL process, which is not desired when users have memory and synchronization constraints.

A.6 CLASS-IMBALANCED LEARNING

Class-imbalanced learning attracts increasing attention for two reasons. First, models trained under this scenario using empirical risk minimization perform poorly on minor classes of scarce training data. Second, many real-world data sets are class-imbalanced by nature (Gupta et al., 2019; Van Horn & Perona, 2017; Van Horn et al., 2018). In this paper, we employ a mainstream approach, cost-sensitive learning (Cao et al., 2019; Li et al., 2020c; Ren et al., 2020; Tan et al., 2020; Ye et al., 2020), which adjusts the training objective to reflect class imbalance so as to train a model that is less biased toward major classes.

A.7 ZERO-SHOT LEARNING

Our design choice of parameterizing the personalized prediction head with clients' class distributions is reminiscent of zero-shot learning (Changpinyo et al., 2016; 2017; 2020; Lampert et al., 2013; Xian et al., 2018), whose goal is to build an object classifier based on its semantic representation. The key difference is that we build an entire fully-connected layer for FL, not just a single class vector. We employ hypernetworks (Ha et al., 2017) for efficient parameterization.

B ADDITIONAL DETAILS OF FED-ROD

B.1 ADDITIONAL BACKGROUND (CF. SUBSECTION 3.1 OF THE MAIN PAPER)

In the *generic* federated learning (FL) setting, the goal is to construct a single "global" model that can perform well for test data from all the clients. Let $\boldsymbol{w}$ denote the parameters of the model, for a

classification problem whose label space is $\mathbb{C}$, a commonly used loss is the cross entropy,

$$\ell(\boldsymbol{x}, y; \boldsymbol{w}) = -\log \frac{\exp(g_y(\boldsymbol{x}; \boldsymbol{w}))}{\sum_{c \in \mathbb{C}} \exp(g_c(\boldsymbol{x}; \boldsymbol{w}))}, \tag{12}$$

where $g_c(\boldsymbol{x}; \boldsymbol{w})$ is the model's output logit for class c.

We note that, the concepts of global vs. local models and generic vs. personalized models should not be confused. No matter which task (generic or personalized) an FL algorithm focuses on, as long as it has the local training step, it generates local models; as long as it has the global aggregation step (of the entire model), it generates a global model. For instance, FEDAVG (McMahan et al., 2017) aims for generic FL but it creates both the global and local models.

B.2 OVERVIEW OF FED-ROD

For generic predictions, FED-ROD performs feature extraction $\boldsymbol{z} = f(\boldsymbol{x}; \boldsymbol{\theta})$, followed by $h^G(\boldsymbol{z}; \boldsymbol{\psi})$. For personalized predictions, FED-ROD performs $\boldsymbol{z} = f(\boldsymbol{x}; \boldsymbol{\theta})$, followed by $h^G(\boldsymbol{z}; \boldsymbol{\psi}) + h^P(\boldsymbol{z}; \boldsymbol{\phi}_m)$. The element-wise addition is performed at the *logit* level. That is, $g_c(\boldsymbol{x}; \boldsymbol{w})$ in Equation 12 can be re-written as

$$g_c(\boldsymbol{x}; \{\boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{\phi}_m\}) = \begin{cases} h_c^G(\boldsymbol{z}; \boldsymbol{\psi}) & \textbf{Generic model}, \\ h_c^G(\boldsymbol{z}; \boldsymbol{\psi}) + h_c^P(\boldsymbol{z}; \boldsymbol{\phi}_m) & \textbf{Personalized model}, \end{cases} \tag{13}$$

where $\boldsymbol{z} = f(\boldsymbol{x}; \boldsymbol{\theta})$ is the extracted feature.

The overall training process of FED-ROD iterates between the local training and global aggregation steps. In local training, FED-ROD aims to minimize the following objective

$$\mathcal{L}_m^{BR}(\{\boldsymbol{\theta}, \boldsymbol{\psi}\}) + \mathcal{L}_m(\{\boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{\phi}_m\}). \tag{14}$$

The empirical risk $\mathcal{L}_m(\boldsymbol{w}_m = \{\boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{\phi}_m\})$ is defined as $\frac{1}{|\mathcal{D}_m|} \sum_i \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w}_m)$, where $\mathcal{D}_m = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{|\mathcal{D}_m|}$ is the training data of client m. We will introduce more options of the balanced risk $\mathcal{L}_m^{BR}$ in subsection B.3. We optimize Equation 14 via stochastic gradient descent (SGD). We updates $\boldsymbol{\theta}$, $\boldsymbol{\psi}$, and $\boldsymbol{\phi}_m$ in a single forward-backward pass, which consumes almost the same computation cost as FEDAVG. For $\mathcal{L}_m(\{\boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{\phi}_m\})$, we do not derive gradients w.r.t. $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$.

We emphasize that, according to subsection 4.2 of the main paper, the finally learned parameters of FED-ROD (linear) are $\bar{\boldsymbol{\theta}}$, $\bar{\boldsymbol{\psi}}$, and $\{\boldsymbol{\phi}_m^\star\}_{m=1}^M$. We then plug them into Equation 13 for predictions.

In algorithm 1 and algorithm 2, we provide pseudocode of our FED-ROD algorithm.

Algorithm 1: FED-ROD (linear) (**Fed**erated **Ro**bust **D**ecoupling)

Server input : initial global model parameter $\bar{\boldsymbol{\theta}}$ and $\bar{\boldsymbol{\psi}}$;
Client m's input : initial local model parameter $\boldsymbol{\phi}_m^\star$, local step size η, local labeled data $\mathcal{D}_m$;
for $r \leftarrow 1$ **to** R **do**
  Sample clients $\mathcal{S} \subseteq \{1, \cdots, N\}$;
  Communicate $\bar{\boldsymbol{\theta}}$ and $\bar{\boldsymbol{\psi}}$ to all clients $m \in \mathcal{S}$;
  for *each client* $m \in \mathcal{S}$ *in parallel* **do**
    Initialize $\boldsymbol{\theta} \leftarrow \bar{\boldsymbol{\theta}}$, $\boldsymbol{\psi} \leftarrow \bar{\boldsymbol{\psi}}$, and $\boldsymbol{\phi}_m \leftarrow \boldsymbol{\phi}_m^\star$;
    $\{\boldsymbol{\theta}_m^\star, \boldsymbol{\psi}_m^\star, \boldsymbol{\phi}_m^\star\} \leftarrow$ **Client local training**$(\{\boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{\phi}_m\}, \mathcal{D}_m, \eta)$; [Equation 8 and Equation 9]
    Communicate $\boldsymbol{\theta}_m^\star$ and $\boldsymbol{\psi}_m^\star$ to the server;
  end
  Construct $\bar{\boldsymbol{\theta}} = \sum_{m \in \mathcal{S}} \frac{|\mathcal{D}_m|}{\sum_{m' \in \mathcal{S}} |\mathcal{D}_{m'}|} \boldsymbol{\theta}_m^\star$;
  Construct $\bar{\boldsymbol{\psi}} = \sum_{m \in \mathcal{S}} \frac{|\mathcal{D}_m|}{\sum_{m' \in \mathcal{S}} |\mathcal{D}_{m'}|} \boldsymbol{\psi}_m^\star$;
end
Server output : $\bar{\boldsymbol{\theta}}$ and $\bar{\boldsymbol{\psi}}$;
Client m's output : $\{\boldsymbol{\theta}_m^\star, \boldsymbol{\psi}_m^\star, \boldsymbol{\phi}_m^\star\}$.

Algorithm 2: FED-ROD (hyper) (**Fed**erated **Ro**bust **D**ecoupling)

Server input : initial global model parameter $\bar{\boldsymbol{\theta}}$, $\bar{\boldsymbol{\psi}}$, and $\bar{\boldsymbol{\nu}}$;
Client m's input : local step size η, local labeled data $\mathcal{D}_m$;
for $r \leftarrow 1$ **to** R **do**
  Sample clients $\mathcal{S} \subseteq \{1, \cdots, N\}$;
  Communicate $\bar{\boldsymbol{\theta}}$, $\bar{\boldsymbol{\psi}}$, and $\bar{\boldsymbol{\nu}}$ to all clients $m \in \mathcal{S}$;
  for *each client* $m \in \mathcal{S}$ *in parallel* **do**
    Initialize $\boldsymbol{\theta} \leftarrow \bar{\boldsymbol{\theta}}$, $\boldsymbol{\psi} \leftarrow \bar{\boldsymbol{\psi}}$, and $\boldsymbol{\nu} \leftarrow \bar{\boldsymbol{\nu}}$;
    $\{\boldsymbol{\theta}_m^\star, \boldsymbol{\psi}_m^\star, \boldsymbol{\nu}_m^\star\} \leftarrow$ **Client local training**$(\{\boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{\nu}\}, \mathcal{D}_m, \eta)$; [Equation 11]
    Communicate $\boldsymbol{\theta}_m^\star$, $\boldsymbol{\psi}_m^\star$, and $\boldsymbol{\nu}_m^\star$ to the server;
  end
  Construct $\bar{\boldsymbol{\theta}} = \sum_{m \in \mathcal{S}} \frac{|\mathcal{D}_m|}{\sum_{m' \in \mathcal{S}} |\mathcal{D}_{m'}|} \boldsymbol{\theta}_m^\star$;
  Construct $\bar{\boldsymbol{\psi}} = \sum_{m \in \mathcal{S}} \frac{|\mathcal{D}_m|}{\sum_{m' \in \mathcal{S}} |\mathcal{D}_{m'}|} \boldsymbol{\psi}_m^\star$;
  Construct $\bar{\boldsymbol{\nu}} = \sum_{m \in \mathcal{S}} \frac{|\mathcal{D}_m|}{\sum_{m' \in \mathcal{S}} |\mathcal{D}_{m'}|} \boldsymbol{\nu}_m^\star$;
end
Server output : $\bar{\boldsymbol{\theta}}$, $\bar{\boldsymbol{\psi}}$, and $\bar{\boldsymbol{\nu}}$ (for personalized model generation).
Client m's output : $\{\boldsymbol{\theta}_m^\star, \boldsymbol{\psi}_m^\star, \boldsymbol{\nu}_m^\star\}$.

B.3 BALANCED RISK MINIMIZATION (BRM)

To learn a generic model, standard federated learning (*e.g.*, FEDAVG (McMahan et al., 2017)) aims to optimize $\mathcal{L}(\boldsymbol{w})$ in Equation 1 of the main paper. In theory, the overall objective $\mathcal{L}(\boldsymbol{w})$ is equal to the expected objective $\mathbb{E}[\mathcal{L}_m(\boldsymbol{w})]$ for client m, if client m's data $\mathcal{D}_m$ are IID partitioned from $\mathcal{D}$. Here, the expectation is over different $\mathcal{D}_m$ partitioned from $\mathcal{D}$. In reality, $\mathcal{L}_m$ could diverge from $\mathcal{L}$ due to non-IID partitions of the aggregated data $\mathcal{D}$ into clients' data. That is, $\mathbb{E}[\mathcal{L}_m(\boldsymbol{w})] \neq \mathbb{E}[\mathcal{L}_{m'}(\boldsymbol{w})] \neq \mathcal{L}(\boldsymbol{w})$. We mitigate the non-IID situation by directly adjusting $\mathcal{L}_m$ such that $\mathbb{E}[\mathcal{L}_m(\boldsymbol{w})] \approx \mathbb{E}[\mathcal{L}_{m'}(\boldsymbol{w})] \approx \mathcal{L}(\boldsymbol{w})$.

Essentially, $\mathcal{L}_m(\boldsymbol{w})$ is the client's empirical risk, which could be different among clients if their class distribution $\mathcal{P}_m(y)$ are different. We, therefore, propose to turn the empirical risk $\mathcal{L}_m$ into a class-balanced risk $\mathcal{L}_m^{BR}$ by replacing ℓ in Equation 12 with a class-balanced loss (Cao et al., 2019; Cui et al., 2019; Kang et al., 2020; Khan et al., 2017; Ren et al., 2020; Ye et al., 2020). The class-balanced loss attempts to make the learned model robust to different training class distributions, such that the learned model can perform well for all the test classes. In other words, the class-balanced loss is designed with an implicit assumption that the test data will be class-balanced, even though the training data may not be. Table 5 summarizes some popular class-balanced losses. We also include some extensions with meta-learning. See subsection B.5.

Table 5: Balanced risk and loss functions. We ignore the normalization in $\mathcal{L}_m(\boldsymbol{w})$. Red highlights the modifications by the balanced losses. Blue highlights the terms learned with meta-learning (see subsection B.5).

Method	$\ell(\boldsymbol{x}, y; \boldsymbol{w})$	$\mathcal{L}_m$ or $\mathcal{L}_m^{BR}$
Cross entropy	$-\log \frac{\exp\left(g_y(\boldsymbol{x};\boldsymbol{w})\right)}{\sum_{c\in\mathbb{C}}\exp(g_c(\boldsymbol{x};\boldsymbol{w}))}$	$\sum_i \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w})$
IR (Hsu et al., 2020)	$-\log \frac{\exp\left(g_y(\boldsymbol{x};\boldsymbol{w})\right)}{\sum_{c\in\mathbb{C}}\exp(g_c(\boldsymbol{x};\boldsymbol{w}))}$	$\sum_i \frac{\sum_{c\in\mathbb{C}} N_{m,c}}{N_{m,y_i}} \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w})$
LDAM (Cao et al., 2019) (γ tuned with validation)	$-\log \frac{\exp\left(g_y(\boldsymbol{x};\boldsymbol{w}) - \gamma N_{m,y}^{-\frac{1}{4}}\right)}{\sum_{c\in\mathbb{C}, c\neq y}\exp(g_c(\boldsymbol{x};\boldsymbol{w})) + \exp\left(g_y(\boldsymbol{x};\boldsymbol{w}) - \gamma N_{m,y}^{-\frac{1}{4}}\right)}$	$\sum_i \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w})$
CDT (Ye et al., 2020) (γ tuned with validation)	$-\log \frac{\exp\left((\frac{N_{m,y}}{N_{m,\max}})^{\gamma} g_y(\boldsymbol{x};\boldsymbol{w})\right)}{\sum_{c\in\mathbb{C}}\exp\left((\frac{N_{m,c}}{N_{m,\max}})^{\gamma} g_c(\boldsymbol{x};\boldsymbol{w})\right)}$	$\sum_i \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w})$
BSM (Ren et al., 2020) ($\gamma = 1$ fixed)	$-\log \frac{N_{m,y}^{\gamma}\exp\left(g_y(\boldsymbol{x};\boldsymbol{w})\right)}{\sum_{c\in\mathbb{C}} N_{m,c}^{\gamma}\exp(g_c(\boldsymbol{x};\boldsymbol{w}))}$	$\sum_i \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w})$
Meta-BSM ($\gamma = 1$ fixed, q_{m,y_i} meta-learned)	$-\log \frac{N_{m,y}^{\gamma}\exp\left(g_y(\boldsymbol{x};\boldsymbol{w})\right)}{\sum_{c\in\mathbb{C}} N_{m,c}^{\gamma}\exp(g_c(\boldsymbol{x};\boldsymbol{w}))}$	$\sum_i q_{m,y_i} \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w})$
Meta-BSM (γ_m, q_{m,y_i} meta-learned)	$-\log \frac{N_{m,y}^{\gamma_m}\exp\left(g_y(\boldsymbol{x};\boldsymbol{w})\right)}{\sum_{c\in\mathbb{C}} N_{m,c}^{\gamma_m}\exp(g_c(\boldsymbol{x};\boldsymbol{w}))}$	$\sum_i q_{m,y_i} \ell(\boldsymbol{x}_i, y_i; \boldsymbol{w})$

Table 6: # of parameters in ConvNets for EMNIST/FMNIST and CIFAR-10/100

Module	EMNIST/FMNIST	CIFAR-10	CIFAR-100
Feature extractor	92,646	1,025,610	1,025,610
Generic head	500	640	6400
Total	93,146	1,026,250	1,032,010
Hypernetworks	8,160 (+8.8%)	20,800 (+2.0%)	104,000 (+10.0%)

One may wonder what if the global distribution is class-imbalanced? Will BRM still be beneficial to FL? In subsection D.5, we perform experiments to show that FED-ROD with BRM can still improve on FEDAVG since BRM seeks to learn every class well. Even though the global distribution might be skewed, BRM provides a novel alternative to mitigate the non-IID problem by making every client optimize a more consistent objective as we discussed above. Designing better losses for BRM in FL will be interesting future work.

B.4 ON FEDERATED LEARNING FOR THE PERSONALIZED HEAD WITH HYPERNETWORKS

One drawback of existing personalized methods is that the personalized models are only available for clients involved in training or with sufficient training data. When new clients arrive in testing, is it possible for the federated system to provide corresponding personalized models?

To this end, instead of learning a specific prediction head $\boldsymbol{\phi}_m$ for each client m, we propose to learn a meta-model $H^P(\boldsymbol{a}_m; \boldsymbol{\nu})$ with a shared meta-parameter $\boldsymbol{\nu}$. The input to H^P is a vector $\boldsymbol{a}_m \in \mathbb{R}^{|\mathbb{C}|}$, which records the proportion of class $c \in \mathbb{C}$ in client m's data. The output of H^P is $\boldsymbol{\phi}_m$ for h^P. In

other words, H^P can adaptively output personalized prediction heads for clients given their local class distributions $\boldsymbol{a}_m$.

We implement the meta-model H^P by a hypernetwork (Ha et al., 2017), which can be seen as a lightweight classifier generator given $\boldsymbol{a}_m$. This lightweight hypernetwork not only enables clients to collaboratively learn a module that can generate customized models, but also allows any (future) clients to immediately generate their own personalized predictors given their local class distribution $\boldsymbol{a}_m$ as input, even without training. We construct the hypernetwork by two fully-connected (FC) layers (with a ReLU nonlinear layer in between). Table 6 summarizes the number of parameters of each part in FED-ROD. Hypernetworks add only a small overhead to the original model.

B.5 EXTENSION WITH META-LEARNING FOR THE IMPROVED BSM LOSS

FED-ROD incorporates a balanced loss to learn the generic model. Here we study a more advanced way to derive such balanced loss with meta-learning. Inspired by (Ren et al., 2018; Shu et al., 2019) and the FL scenario proposed by (Zhao et al., 2018), we seek to combine the BSM loss and re-weighting as $\sum_i q_{m,y_i} \ell^{\text{BSM}}(\boldsymbol{x}_i, y_i; \boldsymbol{w})$, where q_{m,y_i} is meta-learned with a small balanced meta dataset $\mathcal{D}_{\text{meta}}$ provided by the server. (See Table 5 for a comparison.) The $\mathcal{D}_{\text{meta}}$ should have a similar distribution to the future test data. We implement this idea with the Meta-Weight Net (MWNet) (Shu et al., 2019) with learnable parameter $\boldsymbol{\zeta}$.

In addition, we notice that the original BSM loss $\ell^{\text{BSM}}_{\gamma} = -\log \frac{N^{\gamma}_{m,y} \exp(g_y(\boldsymbol{x};\boldsymbol{w}))}{\sum_{c\in\mathcal{C}} N^{\gamma}_{m,c} \exp(g_c(\boldsymbol{x};\boldsymbol{w}))}$ has a hyper-parameter γ which is set to be 1 via validation (Ren et al., 2020). However, in federated learning it can be hard to tune such a hyperparameter due to the large number of non-IID clients. Therefore, we propose to learn a client-specific γ_m with meta-learning for $\ell^{\text{BSM}}_{\gamma_m}$. More specifically, given a meta-learning rate η, the meta-learning process involves the following iterative steps:

1. Compute the Meta-BSM loss with a mini-batch $B \sim \mathcal{D}_m$; *i.e.*, $\forall(\boldsymbol{x}, y) \in B$, compute $\ell^{\text{BSM}}_{\gamma_m}(\boldsymbol{x}, y; \boldsymbol{w}_m)$.
2. Predict the example weights with $q_{m,y} = \text{MWNet}(\ell^{\text{BSM}}_{\gamma_m}(\boldsymbol{x}, y; \boldsymbol{w}_m); \boldsymbol{\zeta}_m), \forall(\boldsymbol{x}, y) \in B$.
3. Re-weight the Meta-BSM loss: $\mathcal{L}^{\text{BR}}_{m,B}(\boldsymbol{w}_m) = \sum_{(\boldsymbol{x},y)\in B} q_{m,y_i} \ell^{\text{BSM}}_{\gamma_m}(\boldsymbol{x}, y; \boldsymbol{w}_m)$, and perform one step of gradient descent to create a duplicated model $\tilde{\boldsymbol{w}}_m = \boldsymbol{w}_m - \eta \nabla_{\boldsymbol{w}_m} \mathcal{L}^{\text{BR}}_{m,B}$.
4. Computes the loss on the meta dataset $\mathcal{D}_{\text{meta}}$ using the duplicated model: $\mathcal{L}^{\text{BR}}_{m,\mathcal{D}_{\text{meta}}}(\tilde{\boldsymbol{w}}_m) = \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{\text{meta}}} q_{m,y} \ell^{\text{BSM}}_{\gamma_m}(\boldsymbol{x}, y, \tilde{\boldsymbol{w}}_m)$, followed by updating $\gamma_m \leftarrow \gamma_m - \eta \nabla_{\gamma_m} \mathcal{L}^{\text{BR}}_{m,\mathcal{D}_{\text{meta}}}$ and $\boldsymbol{\zeta}_m \leftarrow \boldsymbol{\zeta}_m - \eta \nabla_{\boldsymbol{\zeta}_m} \mathcal{L}^{\text{BR}}_{m,\mathcal{D}_{\text{meta}}}$.
5. Update the model: $\boldsymbol{w}_m \leftarrow \boldsymbol{w}_m - \eta \nabla_{\boldsymbol{w}_m} \mathcal{L}^{\text{BR}}_{m,B}(\boldsymbol{w}_m)$.

Throughout the federated learning process, γ_m and $q_{m,y}$ are dynamically learned with meta-learning for different clients and rounds.

Results of FED-ROD with Meta-BSM We sample **10** images for each class (only 0.2% of the overall training set) from the training set as the meta set. We compare to (Zhao et al., 2018) that concatenates the meta set to clients' local data. The results in Table 10 and Table 12 are encouraging. With a very small meta set, FED-ROD outperforms (Zhao et al., 2018) by 1% to 14% on accuracy across different settings, validating the importance of balanced losses and how to set them up dynamically via meta-learning.

B.6 CONNECTION TO INVARIANT RISK MINIMIZATION GAMES (IRMG)

FED-ROD is inspired by a recently proposed machine learning framework Invariant Risk Minimization (IRM) (Arjovsky et al., 2019) and its extension Invariant Risk Minimization Games (IRMG) (Ahuja et al., 2020).

Suppose that the whole dataset is collected from many environments, where data from each environment is associated with its characteristic, IRM introduces the concept of learning an invariant predictor. (Note that, in IRM the learner can access data from all the environments; thus, it is not for an FL setting.) Given the training data partition, IRM aims to learn an invariant feature extractor $\boldsymbol{z} = f(\boldsymbol{x}; \boldsymbol{\theta})$ and a classifier $h(\boldsymbol{z}; \boldsymbol{\psi})$ that achieves the minimum risk for all the environments.

The concept of *environments* can be connected to clients' private local data in FL which are often non-IID. That is, given M environments, we can re-write IRM in a similar expression to Equation 7 in the main paper

$$\min_{\boldsymbol{\theta},\boldsymbol{\psi}} \mathcal{L}^{\text{IRM}}(\boldsymbol{\theta},\boldsymbol{\psi}) = \sum_{m=1}^{M} \mathcal{L}_m(\boldsymbol{\theta},\boldsymbol{\psi}), \tag{15}$$

$$\text{s.t} \quad \boldsymbol{\psi} \in \arg\min_{\boldsymbol{\psi}'} \mathcal{L}_m(\boldsymbol{\theta},\boldsymbol{\psi}'), \forall m \in [M]. \tag{16}$$

Unfortunately, IRM is intractable to solve in practice given the constraint that every environment relies on the same parameters (Ahuja et al., 2020). IRMG relaxes it by reformulating the classifier $\boldsymbol{\psi}$ as an ensemble of environment-specific classifiers (by averaging over model weights) $\bar{\boldsymbol{\phi}} = \frac{1}{M}\sum_m \boldsymbol{\phi}_m$:

$$\min_{\boldsymbol{\theta},\bar{\boldsymbol{\phi}}} \mathcal{L}^{\text{IRMG}}(\boldsymbol{\theta},\bar{\boldsymbol{\phi}}) = \sum_{m=1}^{M} \mathcal{L}_m(\boldsymbol{\theta},\bar{\boldsymbol{\phi}}), \tag{17}$$

$$\text{s.t} \quad \boldsymbol{\phi}_m \in \arg\min_{\boldsymbol{\phi}_{m'=m}} \mathcal{L}_m(\boldsymbol{\theta}, \{\boldsymbol{\phi}_{m'}\}_{m'=1}^{M}), \forall m \in [M]. \tag{18}$$

IRMG is proved to optimize the same invariant predictor of IRM when it converges to the equilibrium in game theory, and it holds for a large class of non-linear classifiers. IRMG is solved through iterative optimization: (1) training the feature extractor $\boldsymbol{\theta}$ with centralized data (*i.e.*, aggregated data from all environments), (2) training the environment-specific classifiers $\boldsymbol{\phi}_m$ on the data of each environment $\mathcal{D}_m$, and (3) updating the main classifier through weight averaging $\bar{\boldsymbol{\phi}} = \frac{1}{M}\sum_m \boldsymbol{\phi}_m$.

We highlight the similarity between IRMG and FED-ROD: both are training a strong generic feature extractor and a set of personalized classifiers. For predictions on data of client (environment) m in Equation 18, IRMG uses $\hat{y} = \frac{1}{M}(\phi_m^\top \boldsymbol{z} + \sum_{m' \neq m} \boldsymbol{\phi}_{m'}^\top \boldsymbol{z})$; FED-ROD's personalized model is $\hat{y} = h^G(\boldsymbol{z};\boldsymbol{\psi}) + h^P(\boldsymbol{z};\boldsymbol{\phi}_m)$. We can connect IRMG to FED-ROD by re-writing its prediction as $h^G(\boldsymbol{z};\bar{\boldsymbol{\phi}}) := \bar{\boldsymbol{\phi}}^\top \boldsymbol{z} = \frac{1}{M}\sum_m \boldsymbol{\phi'}_m^\top \boldsymbol{z}$ and $h^P(\boldsymbol{z};\boldsymbol{\phi}_m) := \frac{1}{M}(\boldsymbol{\phi}_m^\top \boldsymbol{z} - \boldsymbol{\phi'}_m{}^\top \boldsymbol{z})$, where $\boldsymbol{\phi}'_m$ is the client m's model in the previous round/iteration of learning.

IRMG can not be applied directly to federated learning for the following reasons. First, centralized training of the feature extractor is intractable since clients' data are not allowed to be aggregated to the server. Second, to perform the iterative optimization of IRMG, the clients are required to communicate every step, which is not feasible in FL due to communication constraints.

C Implementation Details

Implementation. We adopt ConvNet (LeCun et al., 1998) following the existing works (Acar et al., 2021; McMahan et al., 2017; TensorFlow team, 2016). For EMNIST/FMNIST, it contains 2 Conv layers and 2 FC layers. The Conv layers have 32 and 64 channels, respectively. The FC layers are with 50 neurons as the hidden size and 10 neurons for 10 classes as outputs, respectively. For CIFAR-10/100, it contains 3 Conv layers and 2 FC layers. The Conv layers have 32, 64, and 64 channels, respectively. The FC layers are with 64 neurons as the hidden size and $10/100$ neurons for $10/100$ classes as outputs, respectively. To implement hypernetworks in FED-ROD, we use a simple 2-FC ReLU network with hidden size 16 for EMNIST/FMNIST/CIFAR-100 and 32 for CIFAR-10.

We use standard pre-processing, where EMNIST/FMNIST and CIFAR-10/100 images are normalized. EMNIST/FMNIST is trained without augmentation. The 32×32 CIFAR-10/100 images are padded 2 pixels each side, randomly flipped horizontally, and then randomly cropped back to 32×32.

We train every method for 100 rounds. We initialize the model weights from normal distributions. As mentioned in (Li et al., 2020b), the local learning rate must decay along the communication rounds. We initialize it with 0.01 and decay it by 0.99 every round, similar to (Acar et al., 2021). Throughout the experiments, we use the SGD optimizer with weight decay $1\text{e}{-5}$ and a 0.9 momentum. The mini-batch size is 40 (16 for EMNIST). In each round, clients perform local training for 5 epochs. We report the mean over five times of experiments with different random seeds.

For FEDPROX (Li et al., 2020a), the strength of regularization λ is selected from $[1\text{e}{-2}, 1\text{e}{-3}, 1\text{e}{-4}]$. For FEDDYN (Acar et al., 2021), the strength of regularization λ is selected from $[1\text{e}{-1}, 1\text{e}{-2}, 1\text{e}{-3}]$ as suggested in (Acar et al., 2021). For PER-FEDAVG (Fallah et al., 2020), the meta-learning rate $\hat{\beta}$

Table 7: EMNIST and FMNIST results in G-FL accuracy and P-FL accuracy (%). $\star$: methods with no G-FL models and we combine their P-FL models. §: official implementation.

Dataset	EMNIST			FMNIST					
Non-IID	Writers			Dir(0.1)			Dir(0.3)		
Test Set	G-FL	P-FL		G-FL	P-FL		G-FL	P-FL	
Method / Model	GM	GM	PM	GM	GM	PM	GM	GM	PM
FedAvg	97.0±0.05	96.9±0.05	97.2±0.06	81.1±0.12	81.0±0.14	91.5±0.14	83.4±0.15	83.2±0.15	90.5±0.21
FedProx	97.0±0.05	97.0±0.05	97.0±0.05	82.2±0.15	82.3±0.13	91.4±0.10	84.5±0.14	84.5±0.17	89.7±0.19
SCAFFOLD	97.1±0.11	97.0±0.12	97.1±0.09	83.1±0.25	83.0±0.30	89.0±0.32	85.1±0.27	85.0±0.29	90.4±0.34
FedDyn §	97.3±0.12	97.3±0.10	97.3±0.10	83.2±0.15	83.2±0.16	90.7±0.20	86.1±0.18	86.1±0.17	91.5±0.19
MTL $\star$	75.4±0.85	75.0±0.78	85.6±0.77	36.1±0.65	36.0±0.66	87.3±0.75	53.1±0.70	53.4±0.69	78.3±0.80
LG-FedAvg $\star$§	80.1±0.34	80.0±0.24	95.6±0.15	54.8±0.41	54.5±0.44	89.5±0.64	66.8±0.40	66.8±0.42	84.4±0.55
FedPer $\star$	93.3±0.14	93.1±0.20	97.2±0.11	74.5±0.24	74.4±0.25	91.3±0.48	79.9±0.20	79.9±0.22	90.4±0.41
Per-FedAvg	95.1±0.24	-	97.0±0.14	80.5±0.60	-	82.8±1.20	84.1±0.75	-	86.7±0.99
pFedMe §	96.3±0.11	96.0±0.10	97.1±0.11	76.7±0.33	76.7±0.35	83.4±0.41	79.0±0.35	79.0±0.35	83.4±0.45
Ditto	97.0±0.05	97.0±0.06	97.4±0.09	81.5±0.24	81.5±0.27	89.4±0.41	83.3±0.20	83.2±0.22	90.1±0.34
FedFOMO $\star$	80.5±0.75	80.4±0.78	95.9±0.67	34.5±1.57	34.3±1.59	90.0±0.77	70.1±0.56	69.9±0.55	89.6±0.70
FedRep $\star$§	95.0±0.08	95.1±0.11	97.5±0.05	79.5±0.30	80.1±0.31	91.8±0.29	80.6±0.28	80.5±0.34	90.5±0.35
Local only	-	-	64.2±0.68	-	-	85.9±0.69	-	-	85.0±0.80
FED-ROD (linear)	**97.3**±0.10	**97.3**±0.09	**97.5**±0.09	**83.9**±0.20	**83.9**±0.21	**92.7**±0.24	**86.3**±0.16	**86.3**±0.18	**94.5**±0.20
FED-ROD (hyper)	**97.3**±0.10	**97.3**±0.11	**97.5**±0.08	**83.9**±0.18	**83.9**±0.18	**92.9**±0.26	**86.3**±0.17	**86.3**±0.18	**94.8**±0.19
+ FedDyn	**97.4**±0.08	**97.4**±0.11	**97.5**±0.11	**85.9**±0.22	**85.7**±0.22	**95.3**±0.36	**87.5**±0.26	**87.5**±0.26	**94.6**±0.35

is selected from $[1\mathrm{e}{-2}, 1\mathrm{e}{-3}, 1\mathrm{e}{-4}]$. For pFedMe (Dinh et al., 2020), the strength of regularization λ is selected from $[15, 20, 30]$. FED-ROD introduces no extra hyperparameters on top of FEDAVG.

For the generic and personalized heads of FED-ROD, we study using $1 \sim 4$ FC layers but do not see a notable gain by using more layers. We attribute this to the well-learned generic features. Thus, for all our experiments on FED-ROD, we use a single FC layer for each head.

We run our experiments on four GeForce RTX 2080 Ti GPUs with Intel i9-9960X CPUs.

Evaluation. Both datasets and the non-IID Dirichlet simulation are widely studied and used in literature (Acar et al., 2021; Hsu et al., 2019; Lin et al., 2020). We use the standard balanced test set $\mathcal{D}_{\text{test}}$ for evaluation on generic FL (G-FL):

$$\textbf{G-FL accuracy}: \quad \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_i \mathbf{1}(y_i = \hat{y}_{i,G}), \tag{19}$$

where $\hat{y}_{i,G}$ here is the predicted label (*i.e.*, $\arg\max$ over the logits). For evaluation on personalized FL (P-FL), we still apply $\mathcal{D}_{\text{test}}$ but weight instances w.r.t. each client's class distribution:

$$\textbf{P-FL accuracy}: \quad \frac{1}{M} \sum_m \frac{\sum_i \mathcal{P}_m(y_i) \mathbf{1}(y_i = \hat{y}_{i,P})}{\sum_i \mathcal{P}_m(y_i)}. \tag{20}$$

We do so instead of separating $\mathcal{D}_{\text{test}}$ into separate clients' test sets in order to avoid the variance caused by how we split test data (except the EMNIST dataset that each client has its own test set with the writer's styles). What we compute is essentially the expectation over the splits. We have verified that the difference of the two evaluation methods is negligible.

In Table 1 of the main paper and some other tables in the appendix, we evaluate G-FL by an FL algorithm's generic (usually the global) model, denoted as **GM**. We evaluate P-FL by an FL algorithm's personalized models (or local models of a G-FL algorithm), denoted as **PM**. For P-FL, we also report the generic model's accuracy following the literature to demonstrate the difference.

Due to the space limit of the main paper, we provide the standard deviations of the results of Table 1 in Table 7, Table 8, and Table 9 here.

Table 8: CIFAR-10 results in G-FL accuracy and P-FL accuracy (%). ⋆: methods with no G-FL models and we combine their P-FL models. §: official implementation.

Non-IID	Dir(0.1)			Dir(0.3)		
Test Set	G-FL	P-FL		G-FL	P-FL	
Method / Model	GM	GM	PM	GM	GM	PM
FedAvg	57.6±0.43	57.1±0.42	90.5±0.48	68.6±0.38	69.4±0.41	85.1±0.45
FedProx	58.7±0.21	58.9±0.45	89.7±0.48	69.9±0.39	69.8±0.39	84.7±0.42
SCAFFOLD	61.2±0.56	60.8±0.59	90.1±0.65	71.1±0.61	71.5±0.60	84.8±0.67
FedDyn §	63.4±0.40	63.9±0.38	92.4±0.45	72.5±0.37	73.2±0.39	85.4±0.44
MTL ⋆	12.1±3.55	12.7±3.78	90.6±0.98	13.5±1.89	13.7±1.93	80.2±1.01
LG-FedAvg ⋆§	29.5±1.46	28.8±1.46	90.8±0.61	46.7±0.45	46.2±0.47	82.4±0.65
FedPer ⋆	50.4±0.47	50.2±0.48	89.9±0.50	64.4±0.44	64.5±0.46	84.9±0.55
Per-FedAvg	60.7±0.77	-	82.7±1.41	70.5±0.81	-	80.7±1.23
pFedMe §	50.6±0.56	50.7±0.58	76.6±0.60	62.1±0.60	61.7±0.57	70.5±0.66
Ditto	58.1±0.49	58.3±0.47	86.8±0.61	69.7±0.44	69.8±0.46	81.5±0.59
FedFOMO ⋆	30.5±1.72	31.2±1.74	90.5±0.85	45.3±1.69	45.1±1.66	83.4±0.81
FedRep ⋆§	56.6±0.34	56.2±0.35	91.0±0.50	67.7±0.41	67.5±0.33	85.2±0.45
Local only	-	-	87.4±0.69	-	-	75.7±0.78
FED-ROD (linear)	**68.5**±0.35	**68.5**±0.35	**92.7**±0.54	**76.9**±0.37	**76.8**±0.37	**86.4**±0.49
FED-ROD (hyper)	**68.5**±0.38	**68.5**±0.39	**92.5**±0.55	**76.9**±0.34	**76.8**±0.35	**86.8**±0.55
+ FedDyn	**68.2**±0.42	**68.2**±0.44	**92.7**±0.57	**74.6**±0.43	**74.6**±0.43	**85.6**±0.58

Table 9: CIFAR-100 results in G-FL accuracy and P-FL accuracy (%). ⋆: methods with no G-FL models and we combine their P-FL models. §: official implementation.

Non-IID	Dir(0.1)			Dir(0.3)		
Test Set	G-FL	P-FL		G-FL	P-FL	
Method / Model	GM	GM	PM	GM	GM	PM
FedAvg	41.8±0.67	41.6±0.71	70.2±0.66	46.4±0.44	46.2±0.41	61.7±0.40
FedProx	41.7±0.51	41.6±0.54	70.4±0.60	46.5±0.48	46.4±0.41	61.5±0.50
SCAFFOLD	42.3±0.73	42.1±0.77	70.4±0.69	46.5±0.68	46.5±0.65	61.7±0.65
FedDyn §	43.0±0.39	43.0±0.47	72.0±0.38	47.5±0.41	47.4±0.44	62.5±0.35
MTL ⋆	9.5±6.55	9.3±5.98	60.7±1.45	10.8±8.71	10.7±6.78	49.9±2.33
LG-FedAvg ⋆§	23.5±2.50	23.4±2.14	66.7±1.00	34.5±2.56	33.9±3.01	55.4±1.11
FedPer ⋆	37.6±0.65	37.6±0.63	71.0±0.55	40.3±0.51	40.1±0.53	62.5±0.55
Per-FedAvg	39.0±0.89	-	66.6±1.12	44.5±0.79	-	58.9±1.30
pFedMe §	38.6±0.67	38.5±0.65	63.0±0.80	41.4±0.71	41.1±0.68	53.4±0.70
Ditto	41.7±0.56	41.8±0.54	68.5±0.71	46.4±0.45	46.4±0.46	58.8±0.38
FedFOMO ⋆	35.4±2.00	35.3±1.87	68.9±0.98	39.6±1.89	39.3±1.74	58.4±1.15
FedRep ⋆§	40.7±0.51	40.7±0.55	71.5±0.49	46.0±0.37	46.0±0.40	62.1±0.43
Local only	-	-	40.0±1.03	-	-	32.5±0.99
FED-ROD (linear)	**45.9**±0.44	**45.8**±0.41	**72.2**±0.51	**48.5**±0.39	**48.5**±0.38	**62.3**±0.40
FED-ROD (hyper)	**45.9**±0.41	**45.8**±0.39	**72.3**±0.48	**48.5**±0.42	**48.5**±0.45	**62.5**±0.52
+ FedDyn	**46.2**±0.50	**46.2**±0.51	**72.5**±0.55	**48.4**±0.49	**48.4**±0.47	**62.5**±0.52

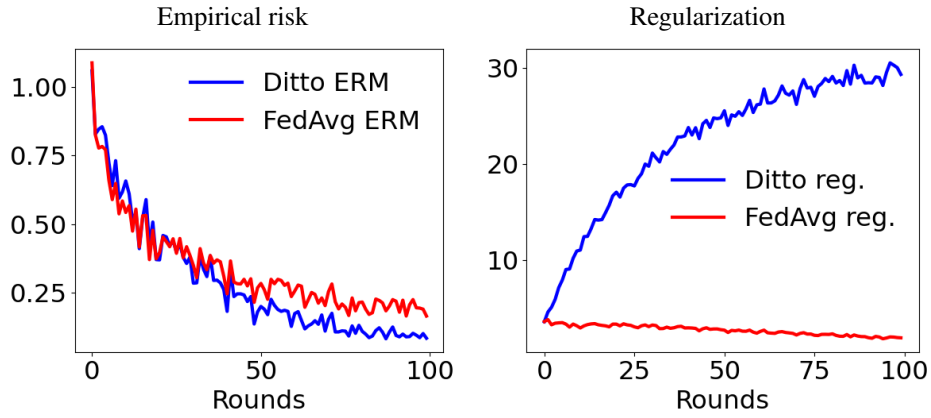


Figure 7: Comparison of the empirical risk and regularization between personalized models of DITTO and local models of FEDAVG. The dataset is CIFAR-10, with Dir(0.3).

D ADDITIONAL EXPERIMENTS AND ANALYSES

Here we provide additional experiments and analyses omitted in the main paper. We validate our claims in the main paper and the designs of our proposed FED-ROD via the following experiments:

- subsection D.1: personalized models emerge from local training of generic federated learning (cf. subsection 3.2, subsection 3.3, and subsection 5.1 in the main paper).
- subsection D.2: balanced risk minimization (BRM) improves generic-FL performance (cf. subsection 5.1 in the main paper).
- subsection D.3: the roles of FED-ROD's generic and personalized heads (cf. subsection 4.2 in the main paper).
- subsection D.4: personalization with hypernetworks (cf. subsection 5.1 in the main paper).
- subsection D.5: robustness to class-imbalanced global data.
- subsection D.6: compatibility of FED-ROD with other G-FL algorithms (cf. subsection 5.1 in the main paper).
- subsection D.7: comparison to personalized FL algorithms (cf. subsection 5.1 in the main paper).
- subsection D.8: ablation studies and discussions on FED-ROD (cf. subsection 5.1 in the main paper).

D.1 PERSONALIZED MODELS EMERGE FROM LOCAL TRAINING OF GENERIC FEDERATED LEARNING

As mentioned in section 3 in the main paper, personalized FL algorithms usually impose an extra regularizer (cf. Equation 3 and Equation 4 of the main paper) during local training, but do not re-initialize the local models by the global models at every round. In contrast, generic FL algorithms like FEDAVG do not impose extra regularization but re-initialize the local models at every round. Here in Figure 7, we monitor the two loss terms, $\sum_m \frac{|\mathcal{D}_m|}{|\mathcal{D}|}\mathcal{L}_m(\boldsymbol{w}_m)$ and $\sum_m \frac{|\mathcal{D}_m|}{|\mathcal{D}|}\|\boldsymbol{w}_m - \bar{\boldsymbol{w}}\|_2^2$ (cf. Equation 3 and Equation 4 of the main paper), for FEDAVG and a state-of-the-art personalized FL algorithm Ditto (Li et al., 2021a) at the end of each local training round. (Ditto does include the L_2 regularizer in training the personalized models.) Ditto achieves a lower empirical risk (*i.e.*, the first term), likely due to the fact that it does not perform re-initialization. Surprisingly, FEDAVG achieves a much smaller regularization term (*i.e.*, the second term) than Ditto, even if it does not impose such a regularizer in training. We attribute this to the strong effect of regularization by re-initialization: as mentioned in subsection 3.3 of the main paper, re-initialization is equivalent to setting the regularization coefficient λ as infinity. We note that, the reason that the regularization term of Ditto increases along the communication rounds is because ever time the global model $\bar{\boldsymbol{w}}$

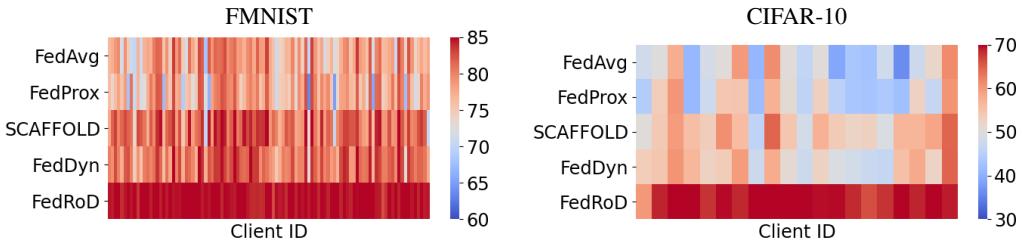


Figure 8: The G-FL accuracy by the local models $\boldsymbol{w}_m$ of different generic methods. There are $100/20$ clients for FMNIST/CIFAR-10, respectively. Both datasets use Dir(0.3).

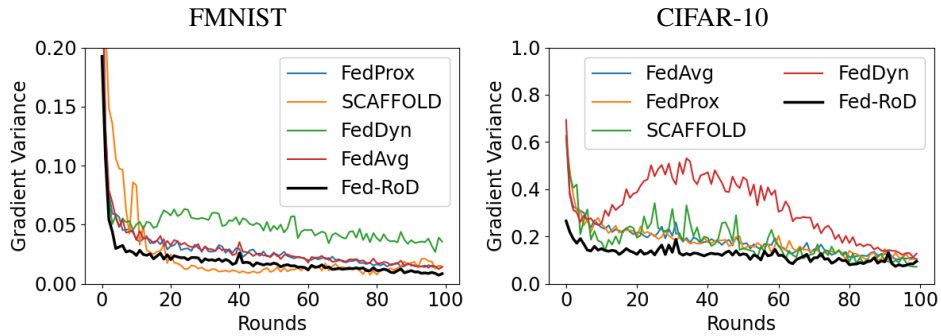


Figure 9: Variances of local model updates w.r.t. the global model. For both datasets, we use Dir(0.3).

is updated, it moves sharply away from the local model $\boldsymbol{w}_m$. Thus, even if the regularization term is added into local training, it cannot be effectively optimized. This analysis suggests that the local models of generic FL algorithms are more regularized than the personalized models of personalized FL algorithms. The local models of generic FL algorithms are thus strong candidates to be evaluated in the personalized FL setting.

D.2 Balanced risk minimization (BRM) improves generic-FL performance

To understand why FED-ROD outperforms other generic methods in the G-FL accuracy, we visualize each *local* model $\boldsymbol{w}_m$'s G-FL accuracy after local training in Figure 8 (both datasets with Dir(0.3)). Methods rely on ERM suffer as their local models tend to diverge. Figure 9 further shows that the variances of local weight update $\Delta \boldsymbol{w}_m = \boldsymbol{w}_m - \bar{\boldsymbol{w}}$ across clients are smaller for FED-ROD, which result from a more consistent local training objective.

In Figure 10, we further compare the G-FL accuracy among FEDAVG, FED-ROD with the original BSM loss, and FED-ROD with the Meta-BSM loss introduced in subsection B.5 along the training process (*i.e.*, training curve). The local models of FEDAVG tend to diverge from each other due to the non-IID issue, resulting in high variances and low accuracy of G-FL. The global aggregation does improve the G-FL accuracy, validating its importance in federated learning. The local training in FED-ROD (BSM) not only leads to a better global model, but also has smaller variances and higher accuracy for the local models (as their objectives are more aligned). With the help of meta dataset and meta-learning, FED-ROD (Meta-BSM) yields even better G-FL performance for both global models and local models, and has much smaller variances among local models' performance, demonstrating the superiority of using meta-learning to learn a balanced objective.

D.3 The roles of FED-ROD's generic and personalized heads

To demonstrate that FED-ROD's two heads learn something different, we plot in Figure 11 every local model's generic prediction and personalized prediction on its and other clients' data (*i.e.*, P-FL accuracy). The generic head performs well in general for every client's test data. The personalized head could further improve for its own data (diagonal), but degrade for others' data.

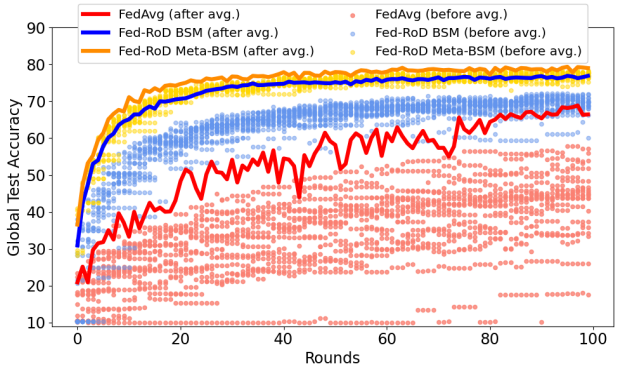


Figure 10: Training curves of different FL algorithms. We show the G-FL accuracy along the training process, using models before (*i.e.*, local models) and after global aggregation. The dataset is CIFAR-10 Dir(0.3).

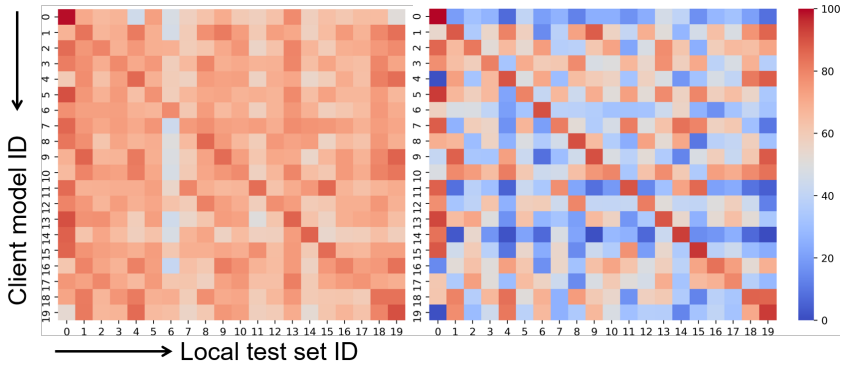


Figure 11: P-FL accuracy of G-head (left) and G-head + P-head (right) using the local models of FED-ROD, evaluated on each client's test data. Here we use CIFAR-10 Dir(0.3) with 20 clients.

D.4 PERSONALIZATION WITH HYPERNETWORKS

FED-ROD (hyper) learns the personalized head with hypernetworks as introduced in subsection B.4. The goal is to learn a hypernetwork such that it can directly generate a personalization prediction head given client's class distribution, without further local training. Figure 12 shows the training (convergence) curves on CIFAR-10 Dir(0.3). The hypernetwork (globally aggregated, before further local training) can converge to be on par with that after local training. In the main paper (cf. Figure 6), we also show that it servers as a strong starting point for future clients — it can generate personalized models simply with future clients' class distributions. That is, the clients may not have labeled data, but provide the hypernetwork with their preference/prior knowledge. It can also be used as the warm-start model for further local training when labeled data are available at the clients.

Table 4 provides the P-FL results for the new 50 clients studied in subsection 5.1 and Figure 6 in the main paper. Except for FED-ROD (hyper), the accuracy before local training is obtained by the global model. The best personalized model after local training is selected for each client using a validation set. FED-ROD (hyper) notably outperforms other methods before or after local training.

D.5 CLASS-IMBALANCED GLOBAL DISTRIBUTIONS

In the real world, data frequency naturally follows a long-tailed distribution, rather than a class-balanced one. Since the server has no knowledge and control about the whole collection of the clients' data, the clients data may collectively be class-imbalanced. This adds an additional challenge for the server to learn a fair and class-balanced model. We follow the setup in (Cao et al., 2019) to transform FMNIST and CIFAR-10 training sets into class-imbalanced versions, in which the sample sizes per class follow an exponential decay. The imbalanced ratio (IM) is controlled as the ratio between

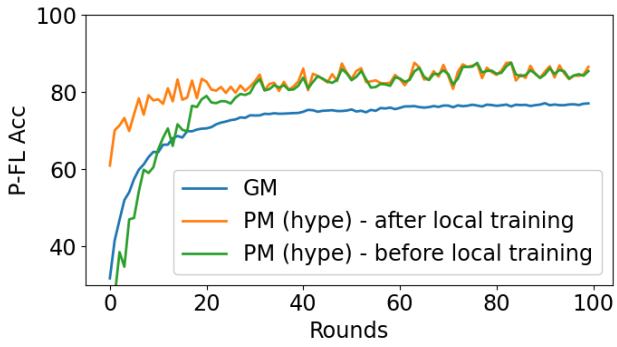


Figure 12: P-FL accuracy of hypernetwork before and after local training.

Table 10: Class-imbalanced global training distribution. *: methods with no global models and we combine their P-FL models. Gray rows: meta-learning with 100 labeled server data.

Dataset	FMNIST						CIFAR-10					
Non-IID / Imbalance Ratio	Dir(0.3), IM10			Dir(0.3), IM100			Dir(0.6), IM10			Dir(0.6), IM100		
Test Set	G-FL	P-FL		G-FL	P-FL		G-FL	P-FL		G-FL	P-FL	
Method / Model	GM	GM	PM	GM	GM	PM	GM	GM	PM	GM	GM	PM
FedAvg (McMahan et al., 2017)	80.2	80.2	85.1	71.6	71.5	86.9	50.9	50.2	76.5	40.1	40.0	78.2
FedProx (Li et al., 2020a)	81.0	81.0	82.3	70.4	70.2	87.0	58.6	58.7	76.5	37.6	38.0	76.6
SCAFFOLD (Karimireddy et al., 2020b)	81.1	81.1	82.2	72.0	71.8	86.9	58.7	58.7	76.6	38.4	38.4	77.6
FedDyn (Acar et al., 2021)	83.3	83.2	86.4	77.2	77.1	87.5	62.5	62.3	80.4	46.6	46.5	80.9
MTL (Smith et al., 2017)*	45.3	45.4	75.6	47.7	47.5	77.5	17.9	18.2	63.4	14.4	14.9	65.9
LG-FedAvg (Liang et al., 2020)*	62.8	62.4	82.4	74.0	74.1	83.2	31.5	31.5	62.8	24.9	24.8	66.3
Per-FedAvg (Fallah et al., 2020)	80.1	-	82.5	72.0	-	78.5	46.3	-	77.2	31.7	-	74.3
pFedMe (Dinh et al., 2020)	78.9	78.9	81.6	69.3	69.2	71.6	46.2	46.2	54.2	31.7	31.8	50.6
Ditto (Li et al., 2021a)	81.0	81.0	83.7	71.8	71.6	86.5	51.0	50.9	73.1	40.3	40.2	75.4
FedFOMO (Zhang et al., 2021)*	65.5	65.2	89.5	64.5	64.4	**90.1**	42.7	42.6	76.6	23.6	23.8	76.7
Local only	-	-	76.1	-	-	79.8	-	-	72.1	-	-	74.5
Fed-RoD (BSM)	81.3	81.3	89.5	76.8	76.8	89.8	63.3	63.3	**80.1**	48.3	48.3	**81.1**
Fed-RoD (BSM) + FedDyn	**84.5**	**84.5**	**90.0**	**80.0**	**79.9**	88.6	**65.5**	**65.5**	79.7	**49.6**	**49.6**	81.0
FedAvg + Meta (Zhao et al., 2018)	80.5	80.5	85.1	71.8	71.8	86.9	51.5	51.3	77.4	40.4	40.2	**84.0**
Fed-RoD (Meta-BSM)	**86.5**	**86.5**	**90.2**	**82.5**	**82.5**	**90.9**	**72.3**	**72.1**	**82.1**	**61.9**	**60.9**	83.5

sample sizes of the most frequent and least frequent classes. Here we consider IM$=10$ and IM$=100$. The generic test set remains class-balanced.

Table 10 shows that Fed-RoD remains robust on both generic accuracy and client accuracy consistently. We see that FedDyn also performs well, especially on FMNIST of which the setup has more clients (100) but a lower participation rate (20%). By combining FedDyn with Fed-RoD, we achieve further improvements.

Essentially, the generic FL methods (except for Fed-RoD) are optimizing toward the overall class-imbalanced distribution rather than the class-balanced distribution. In Table 11, we further examine the G-FL accuracy on a class-imbalanced test set whose class distribution is the same as the global training set. Fed-RoD still outperforms other methods, demonstrating that Fed-RoD learns a robust, generic, and strong model.

D.6 Compatibility of Fed-RoD with other G-FL algorithms

As mentioned in the main paper, other G-FL algorithms like FedDyn (Acar et al., 2021) can be incorporated into Fed-RoD to optimize the generic model (using the balanced risk). We show the results in Table 12, following Table 1 of the main paper. Combining Fed-RoD with SCAFFOLD (Karimireddy et al., 2020b), FedDyn (Acar et al., 2021), and FedProx (Li et al., 2020a) can lead to higher accuracy than each individual algorithm along in several cases.

Table 11: G-FL accuracy on class-imbalanced test data. Here we use CIFAR-10 Dir(0.3).

Method	IM10	IM100
FedAvg (McMahan et al., 2017)	61.8	73.0
FedProx (Li et al., 2020a)	63.1	72.9
SCAFFOLD (Karimireddy et al., 2020b)	65.4	70.4
FedDyn (Acar et al., 2021)	68.6	73.3
Fed-RoD	71.9	76.0

Table 12: Main results in G-FL accuracy and P-FL accuracy (%), following Table 1 of the main paper. Fed-RoD is compatible with other generic FL methods.

Dataset	FMNIST						CIFAR-10					
Non-IID	Dir(0.1)			Dir(0.3)			Dir(0.1)			Dir(0.3)		
Test Set	G-FL	P-FL		G-FL	P-FL		G-FL	P-FL		G-FL	P-FL	
Method / Model	GM	GM	PM	GM	GM	PM	GM	GM	PM	GM	GM	PM
FedAvg (McMahan et al., 2017)	81.1	81.0	91.5	83.4	83.2	90.5	57.6	57.1	90.5	68.6	69.4	85.1
FedProx (Li et al., 2020a)	82.2	82.3	91.4	84.5	84.5	89.7	58.7	58.9	89.7	69.9	69.8	84.7
SCAFFOLD (Karimireddy et al., 2020b)	83.1	83.0	89.0	85.1	85.0	90.4	61.2	60.8	90.1	71.1	71.5	84.8
FedDyn (Acar et al., 2021)§	83.2	83.2	90.7	86.1	86.1	91.5	63.4	63.9	92.4	72.5	73.2	85.4
Fed-RoD (linear)	83.9	83.9	92.7	86.3	86.3	94.5	68.5	68.5	92.7	76.9	76.8	86.4
Fed-RoD (hyper)	83.9	83.9	92.9	86.3	86.3	94.8	68.5	68.5	92.5	76.9	76.8	86.8
+ FedProx	83.3	83.3	93.8	85.8	85.7	92.2	70.6	70.5	92.5	74.5	74.5	85.7
+ SCAFFOLD	84.3	84.3	94.8	88.0	88.0	94.7	72.0	71.8	92.6	77.8	77.7	86.9
+ FedDyn	85.9	85.7	95.3	87.5	87.5	94.6	68.2	68.2	92.7	74.6	74.6	85.6
FedAvg + Meta (Zhao et al., 2018)	83.1	83.1	91.5	84.4	84.3	90.5	58.7	58.9	90.5	69.2	69.2	85.3
Fed-RoD (Meta-BSM)	**86.4**	**86.4**	**94.8**	**89.1**	**89.1**	**94.8**	**72.5**	**72.5**	**92.8**	**80.1**	**80.1**	**86.6**

D.7 Comparison to personalized FL algorithms

From Table 1 in the main paper and Table 10, the personalized FL algorithms are usually outperformed by local models of generic FL algorithms in terms of the P-FL accuracy (*i.e.*, the PM column). The gap is larger when client data are more IID, especially for P-FL methods whose personalized models do not explicitly rely on weight averaging of other clients' models (*e.g.*, MTL, LG-FedAvg, and pFedMe). Some P-FL methods can not even outperform local training alone. A similar observation is also reported in FedFOMO (Zhang et al., 2021). These observations justify the benefits of FL that similar clients can improve each other by aggregating a global model and updating it locally, while the benefits might decay for very dissimilar clients.

To further demonstrate the effect of building a global model and re-initialing the local/personalized models using it (cf. section 3 in the main paper), we investigate Ditto (Li et al., 2021a), a state-of-the-art personalized FL algorithm. We found that Ditto learns *two* local models. One of them is used to build the global model exactly like FedAvg. The global model is then used to regularize the other local model (cf. Equation 4 in the main paper), which is used for personalized prediction. To differentiate these two local models, we call the former the local model (LM), and the latter the personalized model (PM). We note that, the PM model is kept locally and is never re-initialized by the global model. In Table 13, we show the P-FL accuracy using the LM and PM models. The LM model trained in the same way as FedAvg (with re-initialization) surprisingly outperforms the PM model.

We further replicate the experiments in (Li et al., 2021a) on robustness against adversary attacks in Table 14. Besides comparing LM and PM, we also evaluate the global model GM for P-FL accuracy. With out adversarial attacks, the LM model outperforms the PM model. However, with

Table 13: The P-FL accuracy by the two local models of Ditto (Li et al., 2021a).

Method	FMNIST		CIFAR-10	
	Dir(0.1)	Dir(0.3)	Dir(0.1)	Dir(0.3)
PM	89.4	90.1	86.8	81.5
LM	90.8	90.6	90.8	86.2

Table 14: DITTO with adversary attacks. We report the averaged personalized accuracy on benign clients.

Attack	PM	LM	GM
None	94.2	94.7	91.7
Label poisoning	93.6	54.5	84.8
Random updates	93.2	54.5	88.7
Model replacement	63.6	49.8	42.2

Table 15: FED-ROD on CIFAR-10, Dir(0.3).

Test Set	G-FL	P-FL	
Network	GM	GM	PM
ConvNet (LeCun et al., 1998)	76.9	76.8	86.8
VGG11 (Simonyan & Zisserman, 2015)	82.2	82.1	88.2
ResNet8 (He et al., 2016)	80.3	80.0	86.6
ResNet20 (He et al., 2016)	84.0	83.5	88.5

adversarial attacks, the PM model notably outperforms the other two models. We surmise that, when there are adversarial clients, the resulting generic model will carry the adversarial information; re-initializing the local models with it thus would lead to degraded performance.

D.8 ADDITIONAL STUDIES AND DISCUSSIONS

Different network architectures. FED-ROD can easily be applied to other modern neural network architectures. In Table 15, we show that FED-ROD can be used with deeper networks.

FED-ROD is not merely fine-tuning. FED-ROD is not merely pre-training the model with BSM and then fine-tuning it with ERM for two reasons. First, for FED-ROD (linear), the P-head is learned dynamically with the updating feature extractor across multiple rounds. Second, for FED-ROD (hyper), the hypernetwork has to undergo the local training and global aggregation iterations over multiple rounds. In Table 2 of the main paper, we report the fine-tuning baseline. On CIFAR-10 Dir(0.3), it has 84.5% for P-FL (PM), lower than 86.4% and 86.8% by FED-ROD (linear) and FED-ROD (hyper). Note that, hypernetworks allow fast adaptation for new clients.

Comparison to the reported results in other personalized FL papers. Existing works usually report FEDAVG's personalized performance by evaluating its global model (*i.e.*, the GM column in Table 1 of the main paper). In this paper, we evaluate FEDAVG's *local* model $\boldsymbol{w}_m$ (*i.e.*, the PM column in Table 1 of the main paper), which is locally trained for epochs. We see a huge performance gap between these two models. In (Fallah et al., 2020), the authors investigated a baseline "FEDAVG + update", which fine-tunes FEDAVG's *global* model $\bar{\boldsymbol{w}}$ with only few mini-batches for each client. The resulting personalized models thus capture less personalized information than $\boldsymbol{w}_m$ in FEDAVG. For a fair comparison, we also strengthen PER-FEDAVG (Fallah et al., 2020) by updating with more epochs.

Effects of local sample size to P-FL performance In subsection 3.2 and Table 1, we show that local models of generic FL algorithms are strong personalized models. Indeed, the local sample size is an important factor in the P-FL performance. If a client has enough training samples, training its own model (the *local only* baseline) can already be strong without any federated learning. On the other hand, when each client does not have enough samples to train a good model on its own. It will be crucial to have a generic model learned from federated learning as the starting point of personalization.

To confirm our observation when clients have insufficient samples, we further conduct the following experiments. First, we enlarge the number of clients for CIFAR-10 and FMNIST experiments by five times. That is, each client's data size becomes one-fifth on average. Second, we point out that the experiments on CIFAR-100 in Table 1 are with 20 clients. CIFAR-100 has the same total number of training images as CIFAR-10 but with 10 times more classes. In other words, the number of images per class is one-tenth. Table 16 shows the results: all the experiments are based on Dir(0.3). Even when the *local only* models perform worse in P-FL, the local models of FEDAVG still perform on a par with personalized FL algorithms like FEDPER, and FED-ROD can still achieve the best P-FL

Table 16: Main results in G-FL accuracy and P-FL accuracy (%), following Table 1 of the main paper. FED-ROD is compatible with other generic FL methods.

Dataset	FMNIST			CIFAR-10			CIFAR-100		
Test Set	G-FL	P-FL		G-FL	P-FL		G-FL	P-FL	
Method / Model	GM	GM	PM	GM	GM	PM	GM	GM	PM
Local only	-	-	72.9	-	-	76.9	-	-	32.5
FEDAVG (McMahan et al., 2017)	78.1	77.9	85.7	64.2	64.0	77.4	46.4	46.2	61.7
FEDPER (Arivazhagan et al., 2019)	72.5	72.4	85.5	57.6	55.9	78.0	40.3	40.1	62.5
FED-ROD (hype)	82.6	82.6	90.1	72.7	72.7	82.7	48.5	48.5	62.5

accuracy. We attribute the superior personalized performance by FEDAVG and FED-ROD to the implicit regularization discussed in subsection 3.3.

We also want to point out that, even if each client has insufficient data, the P-FL performance of *local only* may still have higher accuracy than the GM of FEDAVG on the personalized accuracy, especially when the non-IID condition becomes severe (e.g., Dir (0.1)). When the non-IID condition is severe, it is harder to train a single GM model to perform well in the personalized setting.