Federated Learning with Heterogeneous and Private Label Sets

Federated Learning with Heterogeneous and Private Label Sets

Adam Breitholtz[1,3] (🖂), Edvin Listo Zec[2], and Fredrik D. Johansson[1]

[1] Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
[2] Research Institutes of Sweden, RISE
[3] adambre@chalmers.se

Abstract. Although common in real-world applications, heterogeneous client label sets are rarely investigated in federated learning (FL). Furthermore, in the cases they are, clients are assumed to be willing to share their entire label sets with other clients. Federated learning with *private* label sets, shared only with the central server, adds further constraints on learning algorithms and is, in general, a more difficult problem to solve. In this work, we study the effects of label set heterogeneity on model performance, comparing the public and private label settings—when the union of label sets in the federation is known to clients and when it is not. We apply classical methods for the classifier combination problem to FL using centralized tuning, adapt common FL methods to the private label set setting, and discuss the justification of both approaches under practical assumptions. Our experiments show that reducing the number of labels available to each client harms the performance of all methods substantially. Centralized tuning of client models for representational alignment can help remedy this, but often at the cost of higher variance. Throughout, our proposed adaptations of standard FL methods perform well, showing similar performance in the private label setting as the standard methods achieve in the public setting. This shows that clients can enjoy increased privacy at little cost to model accuracy.

Keywords: Label set heterogeneity · Federated learning · Distributional shift· Domain adaptation

1 Introduction

Federated learning (FL) enables collaborative model training across distributed clients without centralizing their private data [16]. While promising, the effectiveness of FL is often challenged by statistical heterogeneity, where the data distributions vary significantly across clients. A particularly common and disruptive form of this is *label shift* [15], where the distribution of class labels differs from one client to another. Even more challenging is *label set heterogeneity* [7], where clients' local label sets are disjoint subsets of a global label set.

In applications where access to instances of particular classes holds a competitive advantage, clients may be unwilling to reveal the identities of the classes they observe. Consider, for instance, a consortium of competing pharmaceutical companies wanting to train a model to predict which drug compounds individual patients will have adverse reactions to [3, 12]. Each company has proprietary data on its own set of compounds and reactions, some of which are used by other clients in the federation, and some which are not. They are willing to collaborate to build a more powerful, generalizable model, but would never share the full list of compounds they classify with other clients, as this would reveal information about their R&D pipeline. Instead, clients must communicate model updates with the central server pertaining *only* to their *private label set*.

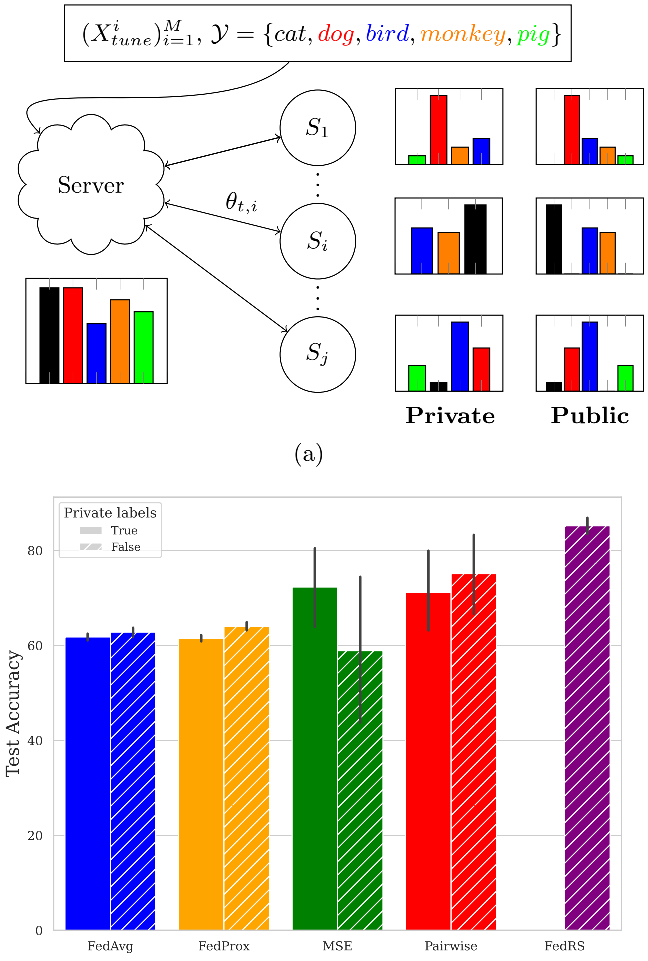
Learning with heterogenous client label distributions has been tackled by several methods, including model distillation [5], contrastive learning [7], and latent space alignment based on class names [23]. However, the literature is more sparse when considering label sets which are not identical across clients, although there are some works which consider it [15]. Moreover, classical methods for federated learning, adapted for label shift or otherwise, can not be applied directly with private label sets. In this setting, the models learned on the clients will necessarily be incompatible for regular aggregation since they will make classifiers for different label sets. Therefore, there is a need to develop methods to deal with this complication to ensure that learning is successful.

In this work, we investigate the effects of client label sparsity and heterogeneity on federated learning performance when client label sets are shared by the whole federation (public) and when they are unknown to other clients (private). We define the private label set problem in Section 3 and adapt popular FL model aggregation strategies for it in Section 4.1. We show that such methods are not well justified when client representations are poorly aligned and propose an alternative method based on the literature on classifier combination in Section 4.2, tuning the central classifier for heterogeneous client representations using an unlabeled dataset at the server. We conduct experiments in image classification on two data sets and show empirically how the sparsity and privacy of label sets affect performance (Section 5). We find that both the private and public label settings are more challenging when clients hold smaller and more diverse subsets of the global label set. Finally, in the private label setting, our proposed adaptations achieve comparable performance to methods for public labels, implying that clients can retain more privacy at little to no cost in accuracy.

2 Related Work

The question of how to combine classifiers trained on disparate label sets have been studied in the binary setting previously in the context of centralized (non-federated) learning. See for example [20] for a comparison of several methods which focuses on combining binary classifiers based on the classifier probabilites.

A similar line of work is the Open-set literature, where the label sets in the clients may be incomplete and the sets may not match between clients. In [2] they learn distribution estimators in the clients to approximate the overall label distribution using uncertainty of the global model.



(a)

(b)

Fig. 1. a): A schematic view of the two settings which we consider in our work. The private setting where the clients are unaware of the full label set and the public setting where this is known. **b):** Results on CIFAR10 where each client has 5 labels available in their respective dataset. The tuning methods with MSE and Pairwise losses perform the best in the private setting. Errorbars represent a 95% confidence interval. Note, FedRS is not applicable with private labels as it needs knowledge of the full label set.

[23] deals with clients which do not share the same label set. They propose having the clients share the names of the labels and aligning the embedding of these names across the representations of clients. However, this requires sharing the names of the labels which may be undesirable, especially in a private label set setting. In a similar vein, [15] restricts the softmax to account for the missing

labels in the clients. The algorithm proposed hinges on knowing which labels a client has to account for which precludes its use in the private label set setting.

Moreover, our setting is also related to shift in client label distributions. There are many works which aim to handle cases when there is a imbalance between client label distributions. This can be done by using regularization which penalizes large deviations in client updates [14, 13] or using control variates to steer the learning [10]. Other techniques include clustering clients with similar data distributions and training separate models for each cluster [4, 17, 19] and meta-learning to enable models to quickly adapt to new data distributions with minimal updates [1, 9]. However, these techniques are not adapted to the private label set setting.

Another related field is that of semi supervised federated learning where some works make use of unlabeled data in FL settings [8]. In these works the unlabeled data is usually available on the client side, which differs from our setting.

3 Problem setup

We consider the problem of federated learning (FL) of a single model h, trained to classify points $\boldsymbol{x} \in \mathcal{X}$ into classes $y \in \mathcal{Y} = \{1, ..., |\mathcal{Y}|\}$ given inputs $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d$. A central server coordinates learning on m clients, indexed by $k = 1, ..., m$, each observing labeled data from an unknown distribution $p_k(\boldsymbol{X}, Y)$. Central to our setting is that clients *do not* have all labels in their data, i.e., clients k are exposed only to a subset $\mathcal{Y}_k \subset \mathcal{Y}$ of labels.

Our goal is to learn a probabilistic classifier $h : \mathcal{X} \to \Delta_{\mathcal{Y}}$, where $\Delta_{\mathcal{Y}}$ is the simplex over *all* classes, that minimizes the expected prediction risk with respect to a loss function $L : \Delta_{\mathcal{Y}} \times \mathcal{Y} \to \mathbb{R}$,

$$\underset{h}{\text{minimize}}\ R(h) \quad R(h) \coloneqq \mathbb{E}[L(h(\boldsymbol{X}), Y)] \ . \tag{1}$$

Here, the expectation $\mathbb{E}$ is defined over an unknown distribution $p(\boldsymbol{X}, Y)$, assumed to be a convex combination of clients $p(\boldsymbol{X}, Y) = \sum_{k=1}^{m} w_k p_k(\boldsymbol{X}, Y)$ where $w \in \Delta_m$ assigns a weight to each client. In the classical FL setting, it is assumed implicitly that the weight is proportional to the number of samples n_k held by the client, $w_k = n_k / (\sum_{k'=1}^{m} n_{k'})$, and the risk is computed over a distribution that matches the client aggregate, as exploited in the aggregation scheme of federated averaging (FedAvg) [16]. However, our methods can be adapted to targeted federated learning where p cannot be expressed by a convex combination of clients [22]. The model $h(x) = \psi(\phi(x))$ typically consists of feature extractor ϕ and a classifier ψ, typically parameterized by a neural network with parameters θ_ϕ and a linear-softmax classifier with parameters θ_ψ, respectively. That is, $h_\theta(x) = \sigma(\theta_\psi^\top \phi(x))$ where $\theta = (\theta_\phi, \theta_\psi)$.

We will consider two settings for the label set (illustrated in Figure 1):

- **Public labels**: All clients know the full global label set $\mathcal{Y}$. This is the standard FL setting but with emphasis on label set heterogeneity.

- **Private labels**: Each client know only their local label set $\mathcal{Y}_k$ and all communication with the central server is restricted to this set. That is, classifier parameters $\theta_\psi(y)$ for labels $y \notin \mathcal{Y}_k$ are not shared with client k.

In both cases, the central server knows the full label set and the label sets of all clients to allow for tailoring communication to clients with heterogeneous and private label sets. We expand on methods for handle the private label case next.

4 Methods for heterogeneous label sets

When the label set is public, we can simply use existing FL methods to learn our classifiers. The issue of label set heterogeneity still remains, and aligning the models to combat any effects of misaligned representations may be warranted. However, in the case where the label set is private, some further modifications have to be made. We detail this and a method of tuning models for alignment in the following sections.

4.1 Model averaging with private label sets

The main challenge addressed in this work is *private label set heterogeneity*: each client k observes labels from a subset $\mathcal{Y}_k \subseteq \mathcal{Y}$ and are *unaware* of other labels $\mathcal{Y} \setminus \mathcal{Y}_k$. Without loss of generality, we assume that for all clients k, every label in $\mathcal{Y}_k$ is observed with positive probability, $\forall y \in \mathcal{Y}_k : p_k(Y = y) > 0$, and other labels are unobserved, $\forall y \notin \mathcal{Y}_k : p_k(Y = y) = 0$.

In the private setting, standard methods (e.g., FedAvg [16], FedProx [14], FedRS [15]) cannot be applied without modification as clients do not have access to the full set of parameters θ_ψ of the shared classifier ψ. Moreover, the server can only receive updates from client k to parameters concerning their subset of labels $\mathcal{Y}_k$. To overcome this obstacle, we propose a simple modification to common model averaging strategies that handles the lack of a full classifier by using the restricted classifiers and reweighting them.

Client-side modification In each round t, each client k is sent the full set of current encoder parameters θ_ϕ^t and the subset of current classifier parameters corresponding to their label set, $\theta_\psi^t[\mathcal{Y}_k] \coloneqq [\theta_\psi^t(y) : y \in \mathcal{Y}_k]^\top \in \mathbb{R}^{|\mathcal{Y}_k|}$. Clients then proceed with local updates as normal.

Server-side modification In each round, t, the server receives parameter updates $(\theta_{\phi,k}^t, \theta_{\psi,k}^t)$ from each client k and averages the classifier parameters for each label y based on the clients which have the label in their label set, weighted according to their sample size (see Algorithm 1). Encoder parameter updates $\theta_{\phi,k}^t$ are averaged as normal.

Surprisingly, this simple method is well-justified under the softmax classifier model, provided that the classifier $h(x) = \sigma(\theta_\psi^\top \phi(x))$ is well-specified and clients' conditional label distributions (mechanisms) are what we call *subset consistent.*

Algorithm 1: FedAvg with private label sets

```
Data: Client label sets $\{\mathcal{Y}_k\}$ and reverse indices $\{I_k\}$
Result: Classifier $h(x) = \sigma(\theta_\psi^\top \phi(x))$
Initialize central parameters $\theta^0 = (\theta_\phi^0, \theta_\psi^0)$
for each round $t = 0, ..., T-1$ do
    for each client $k = 1, ..., m$ do
        Distribute $(\theta_\phi^t, \theta_\psi^t[\mathcal{Y}_k])$ to client $k$
        Receive client update $(\theta_{\phi,k}^t, \theta_{\psi,k}^t)$
    end
    $\theta_\phi^{t+1} = \sum_{k=1}^m \theta_{\phi,k}^t \frac{n_k}{n}$   where   $n = \sum_{k=1}^m n_k$
    for each label $y \in \mathcal{Y}$ do
        $\theta_\psi^{t+1}(y) = \sum_{k:y\in\mathcal{Y}_k} \theta_{\psi,k}^t(I_k(y)) \frac{n_k}{n'_y}$   where   $n'_y = \sum_{k:y\in\mathcal{Y}_k} n_k$
    end
end
Return classifier with parameters $\theta = (\theta_\phi^T, \theta_\psi^T)$
```

Assumption 1 (Subset-consistent labeling mechanisms) *The labeling mechanisms of clients $k = 1, .., n$, each with a distributions $p_k(\boldsymbol{X}, Y)$ on a label set $\mathcal{Y}_k$, are* subset-consistent *if the target label distribution $p(\boldsymbol{X}, Y)$ satisfies*

$$\forall k, \boldsymbol{x} : p_k(Y = y \mid Y \in \mathcal{Y}_k, \boldsymbol{X} = \boldsymbol{x}) = p(Y = y \mid Y \in \mathcal{Y}_k, \boldsymbol{X} = \boldsymbol{x}) \ .$$

Now, suppose that $h_\theta(x)$ is well-specified for the true labeling function $p(Y \mid X)$ given an optimal encoder ϕ, that is, there are parameters θ_ψ such that

$$p(Y = y \mid X = x) = \frac{e^{-\theta_\psi(y)^\top \phi(x)}}{\sum_{y'} e^{-\theta_\psi(y')^\top \phi(x)}} = \sigma(\theta_\psi^\top \phi(x))_y \ .$$

Then, the subset-conditional outcome can be parameterized as a softmax classifiers with parameters $\theta_\psi[\mathcal{Y}_k]$, the subset of θ_ψ restricted to $\mathcal{Y}_k$,

$$\begin{aligned} p(Y = y \mid X = x, Y \in \mathcal{Y}_k) &= \frac{p(Y = y \mid X = x)}{\sum_{y' \in \mathcal{Y}_k} p(Y = y' \mid X = x)} = \frac{e^{-\theta_y^\top \phi(x)}}{\sum_{y' \in \mathcal{Y}_k} e^{-\theta_{y'}^\top \phi(x)}} \\ &= \sigma(\theta_\psi[\mathcal{Y}_k]^\top \phi(x))_y, \end{aligned} \tag{2}$$

since the normalization terms over the full label set cancel. As a result, the optimal model in this circumstance has the same parameters $\theta(y)$ both centrally and in all clients k with $y \in \mathcal{Y}_k$. Consequently, given an optimal encoder $\phi(x)$ in the sense above, *any* convex combination of unbiased estimates $\hat{\theta}_{\psi,k}$ of client-optimal parameters is unbiased for the server-optimal parameters θ. Client weighting based on sample size (as in Algorithm 1) achieves the largest effective sample size (smallest variance) [22].

Remark. In the deterministic case, where $\forall \boldsymbol{x}, \exists y^* : p(Y = y^* \mid \boldsymbol{X} = \boldsymbol{x}) = 1$, Assumption 1 corresponds to the often-used *covariate shift* assumption [18] since the event $Y \in \mathcal{Y}_k$ does not alter the distribution of Y for a given on $\boldsymbol{x}$. In this case, aggregating *perfect* client models $h_k(y \mid x)$ is trivial for a given x, since all of them will return 1 for the correct label. In general, for stochastic labels $p_k(y \mid Y \in \mathcal{Y}_k, \boldsymbol{x}) \neq p_l(y \mid Y \in \mathcal{Y}_l, \boldsymbol{x})$ for $\mathcal{Y}_k \neq \mathcal{Y}_l$. In either case, Assumption 1 allows both marginal distributions $p_k(\boldsymbol{X})$ and $p_k(Y)$ to vary with k.

Based on the simple modifications above, we can also adapt the FedProx [14] algorithm and other centralized model-averaging strategies. For FedProx, we simply omit a comparison of the final layers in the regularization term on the client as their sizes do not match.

The approach detailed in this section is by itself a viable method and will produce a classifier for all classes. However, when representations are not optimal for all clients at once, or when there isn't a single classifier that is optimal in all clients, the justification from (2) fails, and simply averaging client parameters. may not be the best strategy. We explore an alternative strategy next.

4.2 Representation alignment by central tuning

The fundamental problem of federated learning is the aggregation of multiple client models into a single central model that is beneficial to the whole federation. The classical approach of parameter averaging, and its adaptation to private label sets above, is specific to a few model classes (e.g., neural networks) and poorly justified when the averaged representation is suboptimal for some clients. Stepping back, the aggregation problem may be viewed as a special case of classifier combination or couplings [20, 6]. Classifier combination methods were developed to combine several *binary* classifiers, e.g., support vector machines, on different pairs of labels into a single multi-class classifier. Today, this technique is rare as multi-class classifiers are trained routinely using neural networks with softmax outputs or (ensembles of) decision trees. However, in federated learning with heterogenous and private label sets, we face the same problem again since no client nor the server has access to labeled data from all classes.

Traditionally, classifier combination operates on the classifier functions themselves not on their parameters. In our case, the classifiers are estimates of the conditional label probability $p_k(Y \mid \boldsymbol{X})$ specific to each client k and their label sets $\mathcal{Y}_k$. It is appropriate to ask whether there exists a perfect combination of perfect client classifiers, one that yields minimal error on the target distribution $p(\boldsymbol{X}, Y)$. To understand this, we draw inspiration from the binary-to-multi-class problem of classifier combination [6] and note an important distinction to our setting: usually, classifier combination applies to multiple classifiers trained on different subsets of the same data, or at least on data from the same distribution. This implies a structure between the probability distributions that the classifiers aim to fit. For example, with $\mathcal{Y} = \{0, 1, 2\}$, a binary classifier can be used to distinguish classes 0 and 1 by training on samples $(\boldsymbol{x}, y)$ labeled with $y \in \{0, 1\}$. By design, $p(Y = 1 \mid Y \in \{0, 1\}, \boldsymbol{X} = \boldsymbol{x}) = p(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})/p(Y \in \{0, 1\} \mid \boldsymbol{X} = \boldsymbol{x})$.

In general, the clients in federated learning may have completely unrelated label distributions. However, if we suppose again that Assumption 1 holds, a perfect combination of perfect classifiers may be found.

Proposition 1 (Perfect classifier combination) *Let Assumption 1 hold for a set of clients $k = 1, ..., m$ such that clients jointly cover all labels, $\cup_{k=1}^m \mathcal{Y}_k = \mathcal{Y}$. Then, the perfect central classifier $p(Y = y \mid \boldsymbol{X} = \boldsymbol{x})$ can be aggregated from perfect client classifiers $\{p_k(Y = y \mid \boldsymbol{X} = \boldsymbol{x})\}_{k=1}^m$.*

Proof. By Assumption 1, for all clients k, inputs $\boldsymbol{x} \in \mathcal{X}$, and outputs $y \in \mathcal{Y}_k$,

$$p_k(y \mid Y \in \mathcal{Y}_k, \boldsymbol{x}) = \frac{p(y \mid \boldsymbol{x})}{p(Y \in \mathcal{Y}_k \mid \boldsymbol{x})} = \frac{p(y \mid \boldsymbol{x})}{\sum_{y' \in \mathcal{Y}_k} p(y' \mid \boldsymbol{x})} \ . \tag{3}$$

In other words, $\forall \boldsymbol{x}, y, \exists k : p(y \mid \boldsymbol{x}) = c(\boldsymbol{x}) p_k(y \mid Y \in \mathcal{Y}_k, \boldsymbol{x})$ with $c(\boldsymbol{x})$ a normalizing constant. □

The result for softmax classifiers in (2) is a special case of this result.

In practice, of course, we cannot expect to have perfect models of each client to combine—especially not *during* federated learning. However, Proposition 1 gives direction for what a good aggregated model should satisfy. Consider an estimated client model $h_k(y \mid \boldsymbol{x}) \approx p_k(y \mid Y \in \mathcal{Y}_k, \boldsymbol{x})$ and a good central model $h(y \mid \boldsymbol{x}) \approx p(y \mid \boldsymbol{x})$. By (3), it should hold that,

$$\forall k \in [m], y, y' \in \mathcal{Y}_k : h_k(y \mid \boldsymbol{x}) h(y' \mid \boldsymbol{x}) \approx h_k(y' \mid \boldsymbol{x}) h(y \mid \boldsymbol{x}) \ .$$

This is a generalization of the argument in [20] to multi-class subset classifiers. We may use this to construct an aggregation criterion for the central model h, given a set of client models $\{h_k(y \mid \mathcal{Y}_k, \boldsymbol{x})\}_{k=1}^m$, first for a fixed input $\boldsymbol{x}$,

$$\underset{h_{\boldsymbol{x}} \in \Delta_{\mathcal{Y}}}{\text{minimize}} \sum_{k=1}^m w_k \sum_{\substack{y, y' \in \mathcal{Y}_k \\ y \neq y'}} \left(h_k(y \mid \boldsymbol{x}) h_{\boldsymbol{x}}(y') - h_k(y' \mid \boldsymbol{x}) h_{\boldsymbol{x}}(y)\right)^2 \ . \tag{4}$$

If all client models are perfect, the minimizer of (4) is a perfect central model at $\boldsymbol{x}$ under the conditions of Proposition 1. For high-dimensional or continuous $\boldsymbol{x}$, it is not feasible to fit a separate central classifier to each possible input. Instead, we may use function approximation by fitting a classifier $h(y \mid \boldsymbol{X})$ from a class $\mathcal{H} \subset \{h : \mathcal{X} \to \Delta_{\mathcal{Y}}\}$ to minimize the expected error over $p(\boldsymbol{X})$.

$$\underset{h \in \mathcal{H}}{\text{minimize}} \sum_{k=1}^m w_k \sum_{\substack{y, y' \in \mathcal{Y}_k \\ y \neq y'}} \mathbb{E}_{\boldsymbol{X}} \left[\left(h_k(y \mid \boldsymbol{X}) h(y' \mid \boldsymbol{X}) - h_k(y' \mid \boldsymbol{X}) h(y \mid \boldsymbol{X})\right)^2 \right] \tag{5}$$

We call this the *pairwise* tuning loss and will use this as one of our objectives when combining classifiers centrally. In practice, the marginal distribution $p(\boldsymbol{X})$ is unknown and the expectation is intractable to compute, so we must solve (5) with respect to the empirical expectation $\hat{\mathbb{E}}[\boldsymbol{X}]$ over a sample of data. Consequently, to use this method, we require that the central server has access to an *unlabeled* data set of points $\boldsymbol{x}_1, ..., \boldsymbol{x}_m$ drawn from $p(\boldsymbol{X})$. Since access to tuning data is not required by methods based on parameter averaging, we must bear that in mind when comparing the empirical performance of the two approaches.

For additional comparison, we also consider tuning-based classifier combination using the direct *MSE* loss used in [20],

$$\underset{h\in\mathcal{H}}{\text{minimize}} \sum_{k=1}^{m} w_k \sum_{y\in\mathcal{Y}_k} \mathbb{E}_{\boldsymbol{X}} \left[\left(h_k(y \mid \boldsymbol{X}) - h(y \mid \boldsymbol{X}) \right)^2 \right] . \tag{6}$$

In summary, we solve one of the two optimization problems above at each update to tune the classifier to be more aligned with the predictions of the client models. The tuned classifier is sent back to the clients and training proceeds as normal.

5 Experiments

We use the well-known datasets CIFAR-10 [11] and Fashion-MNIST [21] for constructing our experiments. We perform ablations where we vary the number of labels that a client has access to from 2-10. This entails choosing a random set of labels for each client which they then get distributed from the dataset equally. This means that, absent further intervention, the clients will not have an identical amount of labeled examples across the ablation points. To control for this, we perform a subsampling step where we subsample the client dataset randomly to consistently have 2000 samples in each client. When evaluating the impact of tuning on an unlabeled dataset centrally (Section 4.2), we use an unlabeled dataset with 5000 samples for CIFAR10 and 6000 for FashionMNIST. We use the standard test set splits for both datasets, both have 10000 samples.

In the public label set setting, we use FedAvg, FedProx [14] and FedRS [15] as baselines. In the private setting, we adapt FedAvg and FedProx to compare this approach to the central tuning (see Sections 4.1–4.2 for further details). As FedRS depends on knowledge of the full label set on the clients, we cannot use this method in the private setting.

When performing central tuning, we train the server classifier for 3 epochs using one of two loss functions (Pairwise or MSE) in equations (5) and (6), respectively. The model aggregation then follows that of FedAvg (or our adaptation of FedAvg in the private setting). We aggregate the results for different labels per client over 10 independent random seeds and the error bars denote a 95% bootstrapped confidence interval over these splits. For the ablation over epochs per client, we aggregate over 3 random seeds. Further details, including the choice of hyperparameters for each method, can be found in Appendix A.

5.1 Experimental Results

We present the detailed results of our experiments below. For each method, we show the test accuracy of the model snapshot that achieves the highest validation accuracy during a run and then aggregate this across several seeds. More

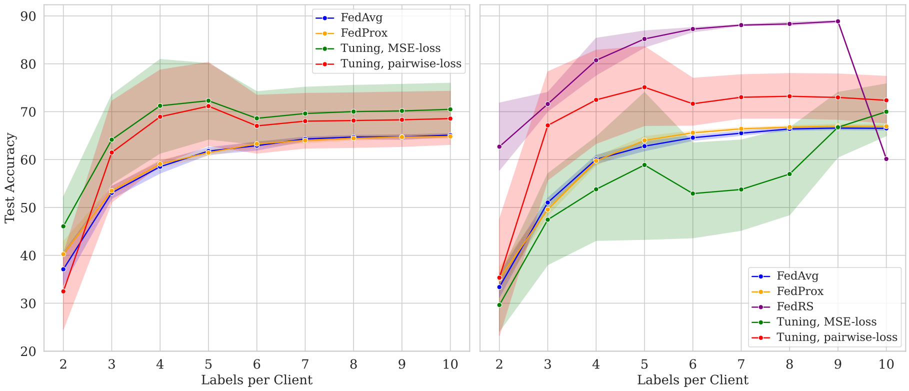


Fig. 2. The performance of the methods in both the private (left) and public (right) settings on CIFAR10. Note that in the regular setting the pairwise loss performs better than FedAvg and the MSE loss while in the private setting the relationship is reversed.

results, including an ablation over client epochs, can be found in Appendix C.

CIFAR10: We present the results varying the amount of labels per client in Figure 2 with the specific case of 5 labels per client being shown in Figure 1b. We clearly see performance decreasing with decreasing number of labels. This effect is not due to decreasing sample size as that is fixed in the experiments. In the public label setting we see that FedRS performs the best while the tuning approach with the pairwise loss performs better than FedAvg and FedProx. Tuning with MSE loss seems to struggle here while, in the private setting, it performs the best. In the private setting, we can see that the tuning approach is superior to the adapted methods, although their variance is higher. Moreover, the MSE loss performs slightly better than the pairwise loss. Interestingly, the pairwise tuning seems to outperform FedAvg and FedProx in the public setting suggesting that the tuning of the representation can be of use in this setting also. This is likely because representational (mis)alignment can be an issue for federated learning whether labels are private or public. It is noteworthy that the adaptation of FedAvg and FedProx seem to exhibit a surprising robustness against the challenges of the private label sets, since they aggregate models from clients with disparate label sets. However, as the clients have an identical amount of labels, each of the feature extractors are trained to output the same label amounts which could help explain the robustness.

Fashion-MNIST: As we can see in Figure 3, the tuning methods do not outperform the adapted methods in the private setting on Fashion-MNIST. However, their large variance suggests that with more careful training they might perform at least equivalently. This may be due to the task being simple and an alignment of classifiers is unnecessary. We show results for 3 labels per client in Figure 4, where we see that the methods perform similarly in the private setting.

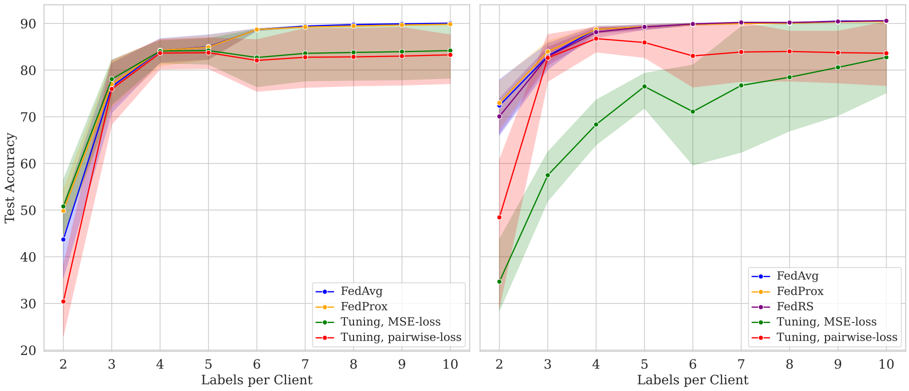


Fig. 3. The performance of the methods in both the private and public settings on FashionMNIST. We note that the tuning approaches do not outperform the adapted methods in the private setting.

The relationship that the pairwise loss performs better than the MSE loss in the public setting and worse in the private setting holds true here also. This could be due to the pairwise loss having cases where the loss is large due to lack of labels in clients. See Appendix B for a discussion of this issue. We see a robustness of FedAvg and Fedprox to the private label sets here as well. This indicates that the adapted methods are a pragmatic choice that also works well in this restricted setting.

In the public setting, we see that the tuning methods underperform the other baselines, with FedAvg, FedProx, and FedRS all performing about equally well. This suggests that there is either limited misalignment between client representations or that it does not adversely affect performance. Instead, the central tuning seems to interfere with the successful learning during federation in the public setting, possibly due to increased variance in the central model. We can observe this variance in the figures over communication rounds in Appendix C.1

6 Discussion

This work investigated the impact of label set heterogeneity on Federated Learning, with a specific focus on the challenging and practical *private label set* setting where clients are unaware of the global label space. Our experiments reveal several key insights into the behavior of both standard and adapted FL algorithms under these conditions.

A primary finding is the surprising robustness of our adaptation of FedAvg to private label sets. We hypothesize that even when individual clients train on a small subset of labels, the shared feature extractor learns a common, semantically rich representation space. The simple aggregation strategy for the classifier

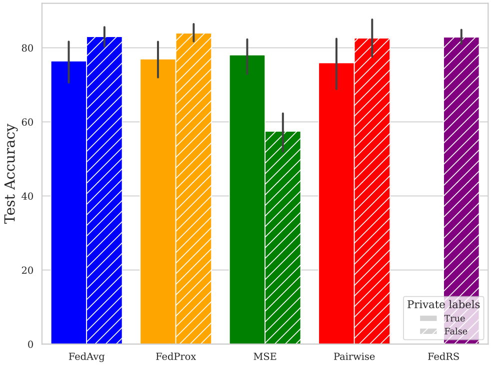


Fig. 4. The results for the private and public settings for 3 labels per client on the FashionMNIST dataset.

weights (Algorithm 1), which combines knowledge on a per-class basis, proves to be a powerful and efficient method for stitching together these partial views into a coherent whole. This establishes adapted FedAvg as a formidable baseline, suggesting that complex alignment mechanisms may not always be necessary if there is sufficient label overlap across the client population.

Further, the tuning approaches do seem to work well for the private setting in some cases while performing worse than adapted methods in others. We also observe that the tuning yields better performance when the sparsity is more extreme. This may be due to the alignment problem being harder with an increasing amount of labels. Notably, we observed a performance reversal between the two tuning losses. In the public setting, the pairwise loss was superior, whereas in the more challenging private setting, the MSE loss performed better. We attribute this phenomenon to a critical vulnerability in the pairwise loss, detailed in Appendix B. When a label is globally absent from all participating clients in a round (a scenario far more likely with fewer labels per client) the pairwise loss generates problematic gradients by comparing against a class for which no client has information. The MSE loss, by directly comparing the global model's predictions to each client's predictions on their known classes, appears more resilient to this issue. It provides a more stable, albeit less constrained, learning signal in the face of extreme sparsity.

A key limitation of the tuning methods are the fact that solving the optimization problem for each communication round could become difficult computationally as the amount of labels, and the number of clients, increases. Also, the existence of unlabeled data is an additional burden to bear that may be impractical in application. The adapted methods do not share these limitations and could be an alternative if tuning methods are computationally infeasible or if there does not exist an unlabeled dataset centrally.

In future work, more realistic datasets could be considered which naturally exhibit label set heterogeneity. In addition, some other FL methods could perhaps be adapted to the private label set setting. Moreover, there could be further consideration of and comparison with other tuning losses.

Acknowledgments. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation

The computations and data handling were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

Bibliography

[1] Chen, F., Luo, M., Dong, Z., Li, Z., He, X.: Federated meta-learning with fast convergence and efficient communication. arXiv:1802.07876 (2018)

[2] Deng, Z., Luo, L., Chen, H.: Scale federated learning for label set mismatch in medical image classification. In: Greenspan, H., Madabhushi, A., Mousavi, P., Salcudean, S., Duncan, J., Syeda-Mahmood, T., Taylor, R. (eds.) Medical Image Computing and Computer Assisted Intervention – MICCAI 2023. pp. 118–127 (2023)

[3] Edwards, I.R., Aronson, J.K.: Adverse drug reactions: definitions, diagnosis, and management. The lancet **356**(9237), 1255–1259 (2000)

[4] Ghosh, A., Chung, J., Yin, D., Ramchandran, K.: An efficient framework for clustered federated learning. Advances in Neural Information Processing Systems **33**, 19586–19597 (2020)

[5] Gudur, G.K., Perepu, S.K.: Federated learning with heterogeneous labels and models for mobile activity monitoring. arXiv preprint arXiv:2012.02539 (2020)

[6] Hastie, T., Tibshirani, R.: Classification by pairwise coupling. Advances in neural information processing systems **10** (1997)

[7] Huang, C., Chen, X., Zhang, Y., Wang, H.: Fedcrl: Personalized federated learning with contrastive shared representations for label heterogeneity in non-iid data. arXiv preprint arXiv:2404.17916 (2024)

[8] Jeong, W., Yoon, J., Yang, E., Hwang, S.J.: Federated semi-supervised learning with inter-client consistency & disjoint learning. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=ce6CFXBh30h

[9] Jiang, Y., Konečný, J., Rush, K., Kannan, S.: Improving federated learning personalization via model agnostic meta learning. arXiv preprint arXiv:1909.12488 (2019)

[10] Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T.: SCAFFOLD: Stochastic controlled averaging for federated learning. In: Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119. PMLR (2020)

[11] Krizhevsky, A.: Learning multiple layers of features from tiny images. Master's thesis, University of Toronto (2009)

[12] Lavan, A.H., Gallagher, P.: Predicting risk of adverse drug reactions in older adults. Therapeutic advances in drug safety **7**(1), 11–22 (2016)

[13] Li, T., Hu, S., Beirami, A., Smith, V.: Ditto: Fair and robust federated learning through personalization. In: International conference on machine learning. pp. 6357–6368. PMLR (2021)

[14] Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. In: Proceedings of Machine Learning and Systems. vol. 2 (2020)

[15] Li, X.C., Zhan, D.C.: Fedrs: Federated learning with restricted softmax for label distribution non-iid data. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining (2021)

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

[17] Sattler, F., Müller, K.R., Samek, W.: Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. IEEE transactions on neural networks and learning systems **32**(8), 3710–3722 (2020)

[18] Shimodaira, H.: Improving predictive inference under covariate shift by weighting the log-likelihood function. Journal of statistical planning and inference **90**(2), 227–244 (2000)

[19] Vardhan, H., Ghosh, A., Mazumdar, A.: An improved federated clustering algorithm with model-based clustering. Transactions on Machine Learning Research (2024)

[20] Wu, T.F., Lin, C.J., Weng, R.: Probability estimates for multi-class classification by pairwise coupling. Advances in Neural Information Processing Systems **16** (2003)

[21] Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)

[22] Zec, E.L., Breitholtz, A., Johansson, F.D.: Overcoming label shift in targeted federated learning. arXiv preprint arXiv:2411.03799 (2024)

[23] Zhang, J., Zhang, X., Zhang, X., Hong, D., Gupta, R.K., Shang, J.: Navigating alignment for non-identical client class sets: A label name-anchored federated learning framework. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 3297–3308 (2023)

A Experimental details

Here we detail the specifics of our experiments. In general we do a 80/20 train and validation split. For the tuning methods we train the classifier for three epochs over the unlabeled dataset each round. Both the central tuning and the client training uses the Adam optimizer with identical learning rates.

A.1 Model

We use a simple CNN model for both experiments.

A.2 Hyperparameters

B Effect of pairwise loss function with a missing label

Consider the loss function:

$$\text{loss\_tensor} = (h_{k,y_j} \cdot p_i - h_{k,y_i} \cdot p_j)^2$$

Layer	# Filters
Convolution	3
Convolution	3
2×2 Max pooling	-
Convolution	3
Convolution	3
2×2 Max pooling	-
Flatten	-
Fully connected	-
Dropout (0.5)	

Table 1. Layers of the CNN used in the experiments.

Learning rate	1×10^{-3}
Batch size	64
Epochs per round	1
Number of communication rounds	100
FedProx μ	1×10^{-2}
FedRS α	0.5

Table 2. hyperparameters used during training.

where:

- h_{k,y_i} is the probability output by client model k for class i (mapped to the client's local label space).
- h_{k,y_j} is the probability output by client model k for class j (mapped to the client's local label space).
- p_i is the probability output by the global model for class i.
- p_j is the probability output by the global model for class j.

Scenario: A single label, label 9, is not present in any client's dataset. All other labels (0-8) are present in at least one client.

Consequences:

1. **Zero client probabilities for label 9:** Because no client has seen label 9, their models will output near-zero (or zero) probabilities for this label:

$$h_{k,y_9} \approx 0 \quad \forall k$$

2. **Problematic loss calculation when label 9 is involved:** The loss calculation becomes problematic when either $i = 9$ or $j = 9$. Let's analyze both cases:
   - **Case 1:** $i = 9$

   $$\text{loss\_tensor} = (h_{k,y_j} \cdot p_9 - h_{k,y_9} \cdot p_j)^2 \approx (h_{k,y_j} \cdot p_9 - 0 \cdot p_j)^2 = (h_{k,y_j} \cdot p_9)^2$$

   The loss depends on the global model's probability for label 9 (p_9) and the client's probability for other labels j. The global model is penalized if p_9 is non-zero, even though no client provides information about label 9.

- **Case 2:** $j = 9$

  $$\text{loss\_tensor} = (h_{k,y_9} \cdot p_i - h_{k,y_i} \cdot p_9)^2 \approx (0 \cdot p_i - h_{k,y_i} \cdot p_9)^2 = (h_{k,y_i} \cdot p_9)^2$$

  Similar to Case 1, the loss depends on p_9 and client probabilities for other labels. The global model is penalized, and gradients related to label 9 are based on "noise" from the clients.

3. **Global model degradation:** The global model's representation for label 9 is negatively impacted. The loss pushes p_9 towards zero because that's the only way to reduce the loss when paired with the near-zero client probabilities. This harms the global model's ability to generalize, even for labels present in client data.
4. **Unfair penalization:** The global model receives gradients that are based on a comparison against the absent label, creating unstable behaviour.

C Additional empirical results

Here we show some additional results.

C.1 Test accuracy over communication rounds

Here we present the test accuracy over time for the public and private settings. We compare FedAvg to the tuning methods we propose. We can see in Figures 5—8 that the convergence of the methods seem to occur at similar times in the training. To note is the increased variance in the tuning methods.

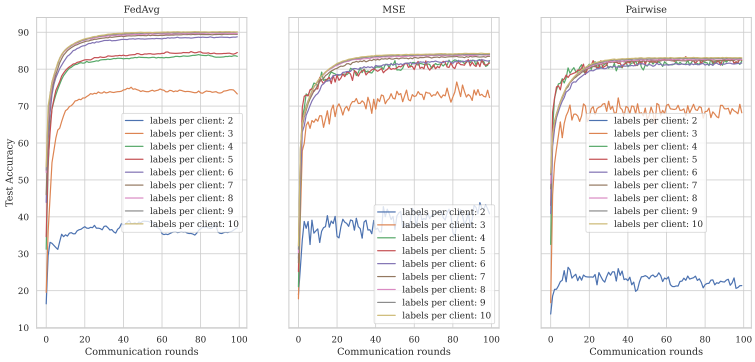


Fig. 5. Test accuracy over rounds in the FashionMNIST task. The labels sets are private.

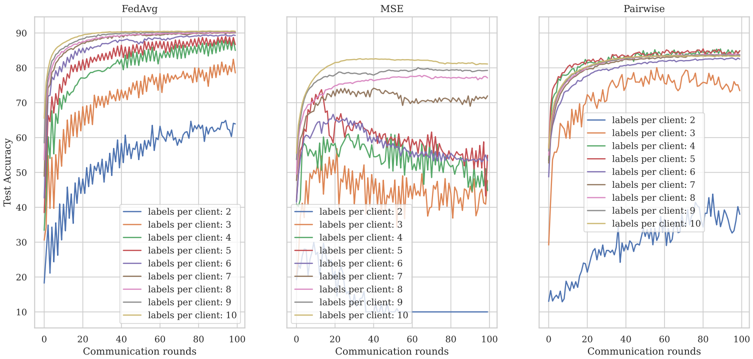


Fig. 6. Test accuracy over rounds in the FashionMNIST task. The labels sets are public.

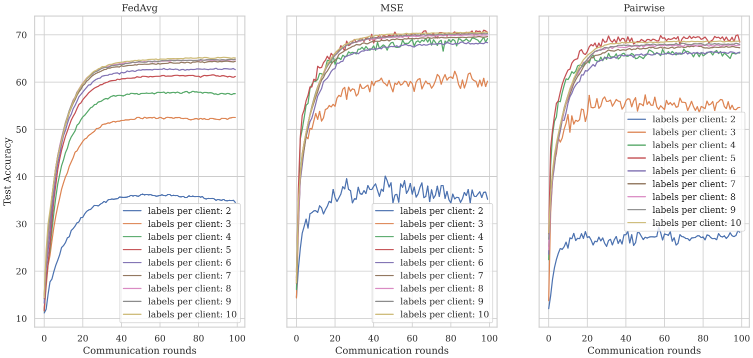


Fig. 7. Test accuracy over rounds in the CIFAR10 task. The labels sets are private.

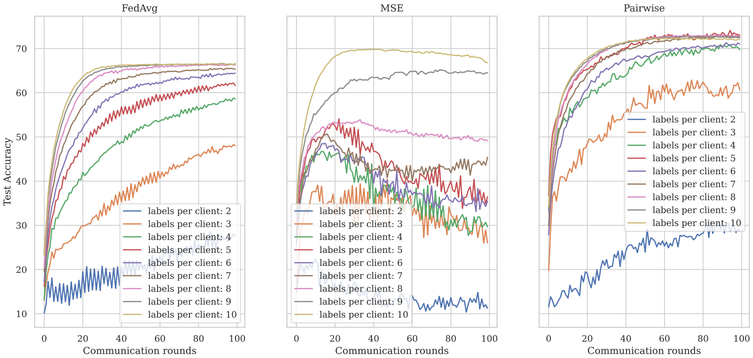


Fig. 8. Test accuracy over rounds in the FashionMNIST task. The labels sets are Public.

C.2 Labels per client barplots

Here are additional barplots to compare performance at different labels per client.

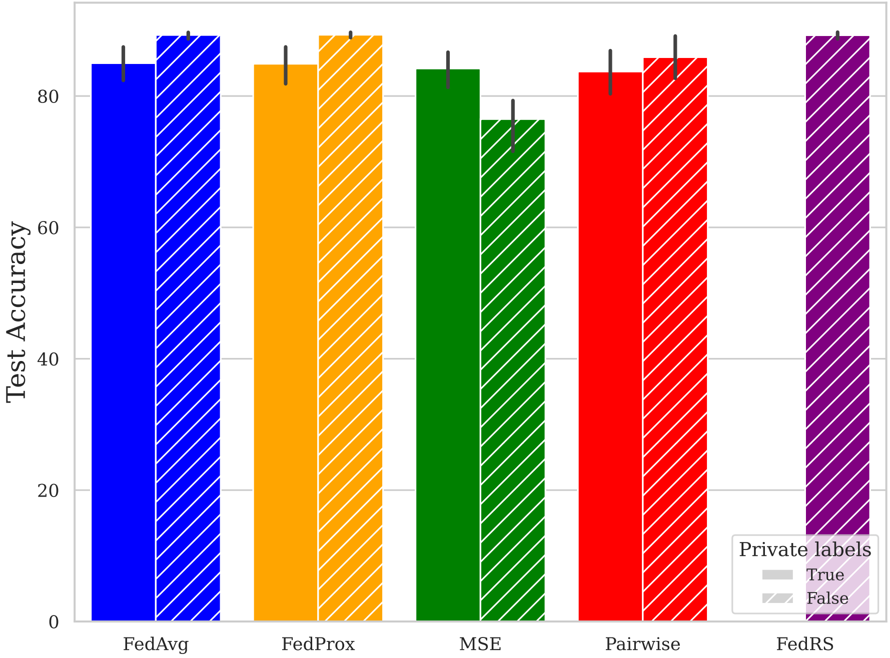


Fig. 9. The results for the private and public settings for 5 labels per client on the FashionMNIST dataset.

C.3 Local epochs

Here we present the results of experiments on CIFAR10 with varying labels per client in figures 11–13. Note, that the central tuning step is done here with a SGD optimizer and not Adam as in other experiments. We see that our ablations across different values of epochs per round do not seem to meaningfully impact performance. Furthermore, the same pattern of the MSE loss performing better in the private setting than the pairwise is seen here.

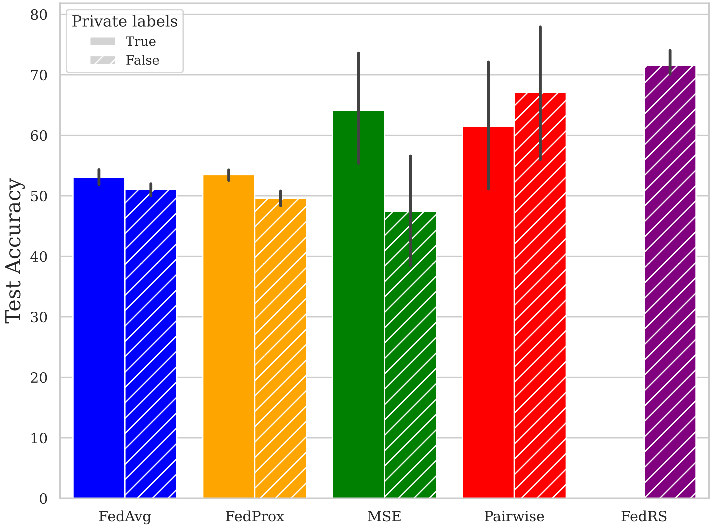


Fig. 10. The results for the private and public settings with 3 labels per client on the CIFAR10 dataset.

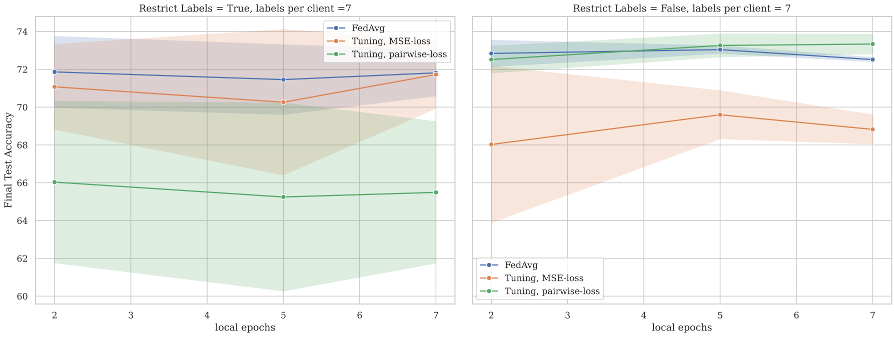


Fig. 11. Ablation with differing amounts of local epochs on CIFAR10. We do not observe any particular drop in performance for either FedAvg or the tuning methods.

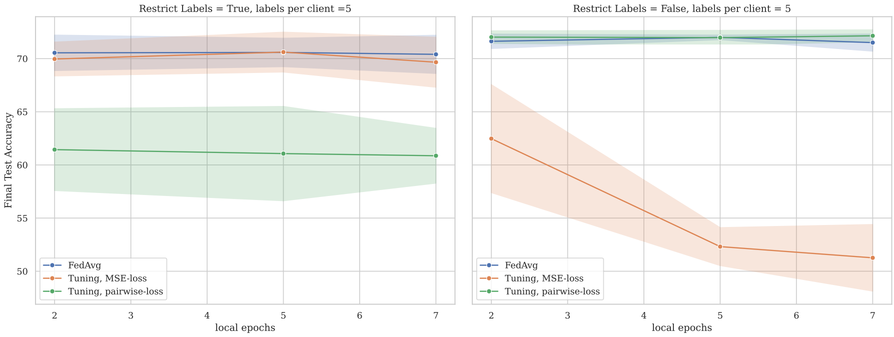


Fig. 12. Comparison of different labels per client on CIFAR10 with E=5.

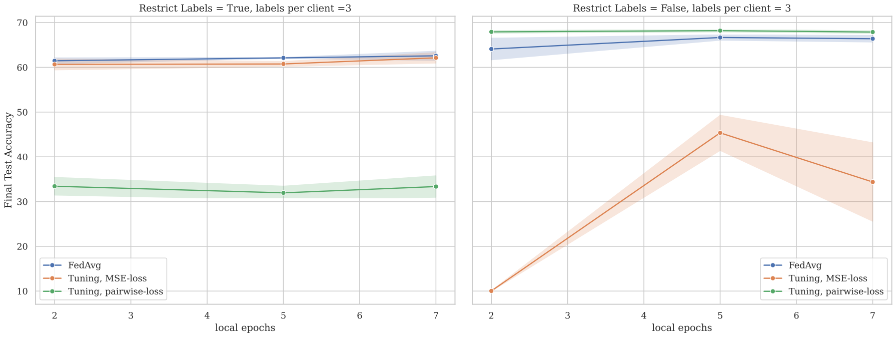


Fig. 13. Comparison of different labels per client on CIFAR10 with E=3.