Orchestra: Unsupervised Federated Learning via Globally Consistent Clustering

Ekdeep Singh Lubana [1] **Chi Ian Tang** [2] **Fahim Kawsar** [3] **Robert P. Dick** [1] **Akhil Mathur** [3]

Abstract

Federated learning is generally used in tasks where labels are readily available (e.g., next word prediction). Relaxing this constraint requires design of unsupervised learning techniques that can support desirable properties for federated training: robustness to statistical/systems heterogeneity, scalability with number of participants, and communication efficiency. Prior work on this topic has focused on directly extending centralized self-supervised learning techniques, which are not designed to have the properties listed above. To address this situation, we propose *Orchestra*, a novel unsupervised federated learning technique that exploits the federation's hierarchy to orchestrate a distributed clustering task and enforce a globally consistent partitioning of clients' data into discriminable clusters. We show the algorithmic pipeline in Orchestra guarantees good generalization performance under a linear probe, allowing it to outperform alternative techniques in a broad range of conditions, including variation in heterogeneity, number of clients, participation ratio, and local epochs.

1. Introduction

Federated Learning (FL) (McMahan et al., 2017) enables collaborative training of machine learning models while avoiding transfer of raw data from clients to server. As remarked by Li et al. (2019), recent work on this topic focuses on addressing the issues of statistical and systems heterogeneity (Li et al., 2020a; Karimireddy et al., 2020; Hsu et al., 2019; Zhao et al., 2018; Hsieh et al., 2020); achieving scalability, privacy, and fairness for participating clients (Charles et al., 2021; Bonawitz et al., 2017; Li et al., 2020b; 2021b; Smith et al., 2017); and improving communication efficiency (Acar et al., 2021; Konečný et al., 2016; Reddi et al., 2021; Lai et al., 2021). However, existing FL algorithms generally assume a participating client holds high quality labels that can be used for gradient-based local training (McMahan et al., 2017). In *cross-silo* settings, where large organizations collaborate to train a model (Kairouz et al., 2021), one can expect each client has an expert to locally label their data. However, in the more constrained scenario of *cross-device* FL (Kairouz et al., 2021), clients are generally edge devices and users need to actively interact with the device to label their data locally. This interaction can be hard to arrange beyond specific applications (e.g., next word prediction), thereby hindering FL's adoption with more complex modalities such as vision.

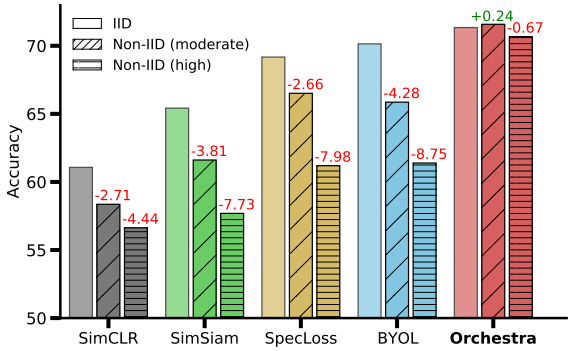


Figure 1. **Robustness to Heterogeneity.** We use CIFAR-10 and 100 clients to compare our proposed method (Orchestra) against federated versions of centralized SSL techniques (Chen et al., 2020; Chen & He, 2021; HaoChen et al., 2021; Grill et al., 2020). Under the linear probe protocol (Chen et al., 2020), we see that these direct extension methods are sensitive to heterogeneity, while Orchestra remains robust and achieves better absolute accuracy.

One possible solution to address this problem is use of unsupervised representation learning algorithms alongside federated training. Some prior works have tried this route by developing direct extensions of self-supervised learning (SSL) techniques from centralized settings, such as SimCLR (Chen et al., 2020), BYOL (Grill et al., 2020), and SimSiam (Chen & He, 2021). However, by using stateful clients (Zhuang et al., 2021; 2022), requiring large batch-sizes (He et al., 2021), or sharing representations across

Work performed while the first two authors were interns at Nokia Bell Labs, UK. [1]EECS Department, University of Michigan, Ann Arbor, USA [2]University of Cambridge, UK [3]Nokia Bell Labs, Cambridge, UK. Correspondence to: Ekdeep Singh Lubana <eslubana@umich.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

clients (Wu et al., 2021; Zhang et al., 2020), these methods are either *only applicable in the cross-silo setting* or *undermine clients' privacy* by enabling inversion of representations (Dosovitskiy & Brox, 2016; Nash et al., 2019) (see also Appendix, Tab. 3). If one addresses these limitations by removing state-based operations and constraining training to local data, these methods become mere applications of centralized SSL objectives with federated training. Since centralized SSL techniques are known to be sensitive to heavy-tailed gradient distributions (Tian et al., 2021b) and require large batch-sizes (Chen et al., 2020), it is unlikely their direct extensions will function well in the high heterogeneity and resource constrained setting of *cross-device* FL. We demonstrate this behavior in Figure 1, where we show direct extension methods lose noticeable performance with increased heterogeneity in a cross-device FL setting.

To resolve the limitations noted above and tackle the lack of labelled data in FL, we argue an unsupervised learning framework needs to be developed that is mindful of the challenges seen in federated training. We thus propose *Orchestra*, a novel clustering-based SSL technique that exploits the federation's hierarchy to orchestrate a global partitioning of data distributed across participating clients. Our clustering based perspective arises out of a generalization analysis of models capable of clustering distributed client data into discriminable, low-similarity partitions. As we show, such a model necessarily has a small test error and its performance *generally improves* with *increase* in heterogeneity (e.g., see Figure 1). To exploit this result, we propose to use the server to compute centroids capable of partitioning clients' data into a predefined number of clusters, subsequently asking the participating clients to use these centroids and locally train a model that enforces a sample's cluster assignments on its augmentations. Experiments show that Orchestra scales well with number of clients, achieves strong communication-efficiency, and thrives under heterogeneity. While our focus in this work is resource-constrained, cross-device FL settings, we find Orchestra also outperforms prior works in cross-silo settings. Our primary contributions follow.

- **Orchestrating Unsupervised FL:** We propose Orchestra, an unsupervised learning technique that addresses the lack of labelled data in FL. The theoretical results motivating our method are discussed in §3 and its practical implementation is provided in §4.
- **Unsupervised Hyperparameter Tuning:** Since FL algorithms can be sensitive to training hyperparameters (Karimireddy et al., 2020; Khodak et al., 2021), we propose a tuning method for self-supervised FL techniques in §5. Our method relies on unlabelled data and finds configurations that yield high (low) representational similarity for related (unrelated) samples.
- **Extensive Empirical Analysis (§6):** We extensively compare Orchestra with several federated versions of centralized SSL techniques. We show, unlike Orchestra, direct extension techniques are often sensitive to several important FL parameters (e.g., participation ratio, local epochs).

2. Preliminaries

Self-Supervised Learning: SSL is a recent paradigm in unsupervised learning wherein either an application-relevant signal is promoted by predicting properties of the data or an application-irrelevant signal is discarded by discriminating perturbed data (Tsai et al., 2021). In vision, examples of predictive tasks include colorization (Vondrick et al., 2018), rotation prediction (Gidaris et al., 2018), or predicting patch permutations (Noroozi & Favaro, 2016). Due to high redundancy in visual data, defining task-relevant signal can be difficult and hence predictive tasks rarely yield good results (Doersch et al., 2015; LeCun, 2019). In contrast, discriminative SSL tasks have revolutionized visual pretraining. Such tasks train the model to enforce invariances to artificial transformations defined using data augmentations, enabling good performance if these transformations encode task-relevant priors (Wen & Li, 2021; Tian et al., 2020). Popular techniques are based on contrastive learning (Chen et al., 2020; He et al., 2020; HaoChen et al., 2021), similarity-promotion (Grill et al., 2020; Chen & He, 2021)), redundancy reduction (Zbontar et al., 2021), and clustering (Asano et al., 2020; Caron et al., 2020).

As mentioned in §1, several FL papers directly extend the centralized SSL techniques listed above, but their methods are either only applicable in cross-silo settings due to use of stateful clients and large batch sizes, or undermine clients' privacy by sharing representations across clients (see also Table 3). We highlight a notable recent exception by Lu et al. (2022), who assume the server has knowledge of client-level class priors, allowing it to retrieve the exact labels by solving a classic label proportions problem (Quadrianto et al., 2008). This strong assumption *may* be justifiable in cross-silo settings if participating clients actively agree to share information about expected class priors, but cannot be met in the cross-device setting, where clients are passive.

Clustering and Representation Learning: Prior work has studied the task of clustering a predefined set of vectors distributed across a federated network (Dennis et al., 2021) or clustering clients for designing better client selection strategies (Ghosh et al., 2020). In contrast, our work addresses the problem of learning clustering-friendly representations (Yang et al., 2017). This problem is known to be difficult even in the centralized setting due to the existence of degenerate solutions, requiring either several expensive reassignment steps (Caron et al., 2018; 2019), degeneracy regularization with autoencoders (Dizaji et al., 2017; Fard et al., 2018), or partition constraints on large prototype memories (Huang et al., 2020; Zhan et al., 2020).

For the reasons listed above, we highlight that clustering-based centralized SSL methods like SeLa (Asano et al., 2020) or SwAV (Caron et al., 2020) are not directly applicable to federated settings because they avoid degenerate solutions using periodic cluster re-assignments every few training steps on a large memory module with 4K–1M samples, while also assuming a uniform class prior on the data. Since communication is expensive in FL, each client has only a few samples, and class priors are highly non-uniform. Such requirements make SeLa and SwAV infeasible for federated learning. We note that Orchestra avoids problems noted above by exploiting the federation's hierarchy. Specifically, Orchestra uses the server in a federated manner to find centroids that can partition the clients' data into discriminable clusters. The clients then locally solve an unsupervised clustering problem alongside a predictive SSL task to avoid degenerate solutions. Together, these steps allow Orchestra to learn "clusterable" representations.

Theory of SSL: Recently, substantial advances have been made towards demystifying SSL methods from the perspectives of learning theory (Arora et al., 2019), information theory (Tsai et al., 2021), causality (Kugelgen et al., 2021), and dynamical systems (Tian et al., 2021a). We use tools proposed in these works to motivate principles behind Orchestra. Specifically, our analysis in §3 is based on recent works by HaoChen et al. (2021) and Wei et al. (2021), who derive a holistic framework that allows analysis of generalization error of an unsupervised model. To derive this result, the authors assume there exists a classifier that is able to predict the label of an augmented sample, given the original sample, up to a small error. By being able to predict the labels of both an augmented sample and an original sample (a no-transform augmentation), this assumption implies there exists a latent space where related samples are sufficiently similar and underlying classes are sufficiently dissimilar.

Notations: Before continuing, we define our settings. Assume we have $N > 2$ samples $X \sim \mathcal{X}$ that are distributed across K clients. We assume a ground-truth labelling function $Y(x) : \mathcal{X} \to [M]$. The i^{th} sample $x^{(i)} \in X$ is assigned to client $k^{(i)} \in [K]$; client k has N^k samples denoted X^k. We define a stochastic augmentation function $\mathcal{T} : \mathcal{X} \to \tilde{\mathcal{X}}$ that transforms its input x to the space of augmented samples by randomly selecting a transform from a large, finite set of predefined transformation functions. The set of augmented samples is denoted as $\tilde{X} := \{\mathcal{T}(x) : x \in X\}$. We define a parametric representation function $f : \mathcal{X} \to \mathrm{R}^D$ and compute its error under a linear probe as $\mathcal{E}(f) := \min_W \mathbb{E}_{x\in\mathcal{X}}[\arg\max(W^T f(x)) = y(x)]$. Note this definition uses the *optimal* linear classifier for a distribution, making its guarantees stronger than a linear probe derived from training data only. The set of representations on a set χ is denoted as $R_\chi = \{f(x) : x \in \chi\}$. We use $\mathcal{C}(\mathcal{B}, G)$ to denote a clustering algorithm that returns G clusters with centroids $\mu \in \mathrm{R}^{D\times G}$ on its input $\mathcal{B}$. μ_g denotes centroid of cluster g. Cluster assignment probabilities are computed as $P_f(x) := \sigma(s_f(x, \mu))$, where $\sigma(.)$ denotes the softmax function and $s_f(x, \mu) = \mu^T f(x)$. $\mathcal{H}(., .)$ denotes cross-entropy between two discrete distributions. $\mathcal{F}$ denotes a hypothesis class that has a global minimizer of the following loss (HaoChen et al., 2021): $L_{\text{spec}} = -2\mathbb{E}_{x\in X,\tilde{x}\sim\mathcal{T}(x)}[s_f(x, \mu)^T s_f(\tilde{x}, \mu)] + \mathbb{E}_{x,y\in X}[(s_f(x, \mu)^T s_f(y, \mu))^2]$.

3. Clusterability: Symphony Behind Orchestra

In this section, we motivate our reasons for using clustering as the principle behind Orchestra. We begin by analyzing an intentionally *idealized framework* where clients are allowed to share their representations with the server. Our goal is to determine whether evaluating the "clusterability" of representations of a federated model trained in an unsupervised manner can provide insights into the quality of the model. To this end, we first define the following.

Definition 3.1. (δ, Inter-Cluster Mixing) Assume we compute G clusters on a dataset $\mathcal{B}$. Then, inter-cluster mixing is defined as $\delta := \max_{g\in G} \max_{b\in\{\mathcal{B}-g\}} \left(\mu_g^T b\right)$.

Analogous to the concept of modularity in pairwise clustering (Newman, 2008), δ is a similarity measure that can be used for analyzing partition-based clustering algorithms, which explicitly compute centroids and assignments. A smaller δ denotes better separation between clusters, indicating better "clusterability" of the dataset.

Proposition 3.2. *Assume $f \in \mathcal{F}$. Compute $\mathcal{G} > 4M + 2$ clusters $\mu = \mathcal{C}(\{R_{X^k} : k \in [K]\}, \mathcal{G})$ s.t. all clusters are equally sized. Then, if f minimizes $\mathcal{L} := \mathbb{E}_{k\in[K]}\left[\mathbb{E}_{x\in X_k,\tilde{x}\sim\mathcal{T}(x)}\left[\mathcal{H}\left(P_f(x), P_f(\tilde{x})\right)\right]\right]$, we have*

$$\mathcal{E}(f) < \zeta_{\mathcal{X}} + \mathcal{O}\left(2\delta + (G-1)\delta^2\right). \tag{1}$$

Here $\zeta_{\mathcal{X}}$ is a constant that measures the similarity of latent variables of two classes from distribution $\mathcal{X}$, while $\mathcal{O}$ hides constants that primarily depend on the dataset size N. Cluster size constraints are employed to ensure that for all classes in the dataset, there is at least one non-trivially sized cluster that contains samples corresponding to that class. If the class-priors were known beforehand, one could enforce size constraints proportional to them, yielding a tighter bound (Wei et al., 2021). However, in unsupervised settings, such priors are unlikely to be known and hence we are forced to use a uniform prior. *Intuitively, Prop. 3.2 says that if the representations learned by f are sufficiently diverse to enable computation of $\mathcal{G}$ "global" clusters with small inter-cluster mixing δ, then f must have a small linear probe error. Further, the smaller δ is (more "clusterable" representations), the smaller f's error is.*

Overall, Prop. 3.2 gives us a target to optimize for while designing our unsupervised FL technique: we must design

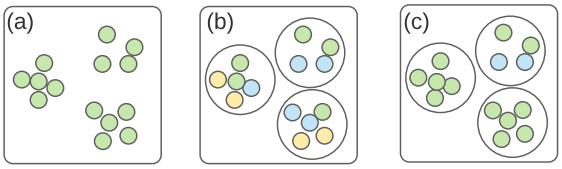


Figure 2. **Global Clusters from Different Settings.** Squares denote global clusters. Smaller/larger circles denote samples/local clusters. Colors denote assignments from the idealized setting. (a) Idealized setting. (b) Under low heterogeneity, samples from different ideal clusters exist on a client and can be merged together during local clustering. This leads to global clusters inconsistent with the idealized setting. (c) As heterogeneity increases, samples from fewer idealized clusters exist on each client, thereby reducing inconsistencies and yielding global clusters similar to (a).

a method that minimizes δ and consequently learns good representations. However, the illustrative method above is not yet practical. In particular, requiring that representations of all data be shared with the server can lead to high communication costs over multiple FL rounds. Further, if the server is semi-honest (Kairouz et al., 2021), sharing representations can undermine clients' privacy via inversion of representations (Dosovitskiy & Brox, 2016; Mahendran & Vedaldi, 2015). Thus, we need to design a framework that offers a result similar to Prop. 3.2 and is yet practical for federated settings. To this end, we propose to perform a *local clustering operation* on our clients.

Specifically, we compute $L^{(k)}$ *local* centroids from $N^{(k)}$ representations at each client k, share these centroids with the server, and run another clustering operation there to partition the overall set of local centroids into G *global* clusters. This scheme reduces the communication cost per client to just $L^{(k)}$, which can be small if few centroids are used. Further, it provides a general operation where, depending on a system's constraints, different levels of privacy can be added locally without affecting other parts of the pipeline. E.g., for strong privacy guarantees at possibly high loss in utility, locally differentially private (local-DP) clustering methods can be used (Balcan et al., 2017; Chang et al., 2021); for slightly weaker guarantees but higher utility, K-*anonymous* clustering methods can be used (Aggarwal et al., 2010; LeFevre et al., 2006; Byun et al., 2007). Most importantly, we find local clustering can provide us a generalization guarantee similar to Prop. 3.2.

Proposition 3.3. *Assume $f \in \mathcal{F}$. Denote the set of local centroids as $\mu^L = \{\mathcal{C}(R_{X^k}, L^{(k)}) : k \in [K]\}$ and compute new global centroids $\mu^G = \mathcal{C}(\mu^L, G)$ s.t. all clusters are equally sized. Assume at least a fraction c samples are "consistently" assigned, i.e., they match their assignments from the idealized setting. Then, if f minimizes the loss $\mathcal{L} := \mathbb{E}_{k\in[K]}\left[\mathbb{E}_{x\in X_k,\tilde{x}\sim\mathcal{T}(x)}\left[\mathcal{H}\left(P_f(x), P_f(\tilde{x})\right)\right]\right]$,*

$$\mathcal{E}(f) < \zeta_{\mathcal{X}} + \mathcal{O}\left(\gamma(1-c^2) + (2\delta + (G-1)\delta^2)\right). \quad (2)$$

Here $\gamma < 1.5$ is a constant and the term $1 - c^2$ signifies the influence of "inconsistent" assignments. In particular, consider a case where samples from multiple idealized clusters are present on a client; during local clustering, such samples can get assigned to the same local cluster. When the centroid of this local cluster is used for global clustering, it inevitably forces samples with different idealized setting assignments to the same global cluster (see Figure 2). This increases inconsistencies with the idealized setting (smaller c), consequently increasing the upper bound in Prop. 3.3.

Interestingly, we observe that heterogeneity in FL setups is beneficial in addressing this challenge! As shown by Dennis et al. (2021), if the federation has heterogeneity such that a client's data predominantly belongs to only a few clusters (e.g., a smartphone may have several images of the same location), the probability that assignments from centralized clustering match assignments found using global clustering of local centroids approaches 1. If we consider model representations to be a predefined set of vectors distributed across clients, this result directly becomes applicable to our settings: with increase in heterogeneity, c approaches 1 and consequently the bound in Prop. 3.3 matches the bound in Prop. 3.2. *That is, local clustering can use higher heterogeneity to its advantage and achieve guarantees similar to the idealized setting.*

In summary, limitations discovered in the idealized setting (i.e., sharing representations with the server) can be circumvented by relying on local clustering. Furthermore, this general, modular operation exploits heterogeneity to its benefit, preserving guarantees provided by the idealized setting. Thus, our target for optimization via FL remains the same as in Prop. 3.2: we need to develop a training process that produces consistent representations across augmentations, induces sufficient number of global clusters, and has low inter-cluster mixing δ. *Our insight uncovered in Prop. 3.3 is that it is sufficient if these global clusters are computed over local centroids, instead of client representations themselves.*

4. Method: How to Conduct an Orchestra

We now detail the steps underlying our proposed unsupervised FL technique, *Orchestra*. See appendix §C.2 for a formal algorithm, implementation details, and github link.

Pipeline: As shown in Prop. 3.3, we must ensure δ is small. To this end, every round, we have clients convert their data into representations and run a clustering algorithm to partition them. The local centroids computed by the clients are then shared with the server, which runs another clustering algorithm on these aggregated centroids. To enforce size constraints from Prop. 3.3 and obtain maximally dissimilar clusters, we use Sinkhorn-Knopp based clustering (Genevay et al., 2019). These methods rethink clustering as an op-

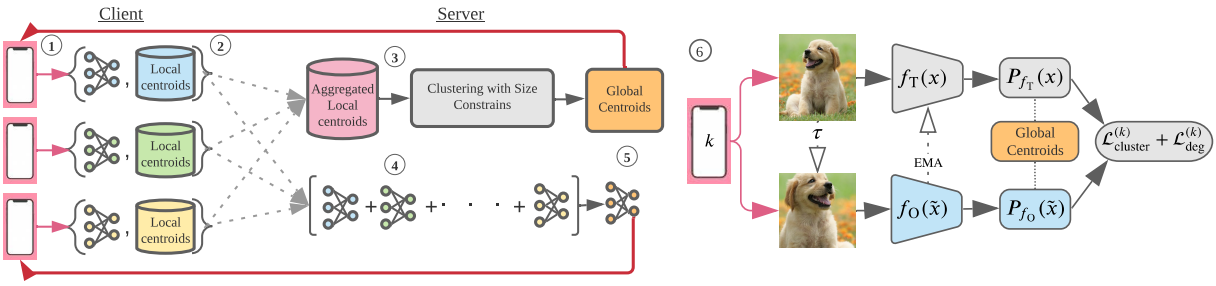


Figure 3. **Pipeline:** ① Orchestra first prompts its clients to compute representations (reps.) on local data. ② Local centroids are computed via a Sinkhorn-Knopp based clustering algorithm (Genevay et al., 2019) to constrain clusters to be equally sized. This operation is extremely cheap (0.009% of client runtime; see §C.2) and enables a K-anonymity privacy guarantee, though local-DP also works well at the expense of some utility (see §D.4). ③ Local centroids from all clients are aggregated at the server, which again uses Sinkhorn-Knopp clustering (Genevay et al., 2019) to compute equally-sized global clusters. Note that Sinkhorn-Knopp is used here to satisfy constraints of Prop. 3.3, not to anonymize the data. ④ Standard FL model averaging step. ⑤ The global centroids are returned to the clients, who use them for local training. **Local Training:** ⑥ An input x is randomly sampled and transformed to $\tilde{x}$. x is converted to rep. $f_{\mathrm{T}}(x)$ using an EMA model to enable assignment prediction; $\tilde{x}$ is converted to $f_{\mathrm{O}}(x)$ using the online model. Cluster assignments $P_{f_{\mathrm{T}}}(x), P_{f_{\mathrm{O}}}(\tilde{x})$ are computed by matching reps. with global centroids, which are then matched using a cross-entropy loss (Equation 4). A predictive SSL task is used to avoid degenerate solutions (Equation 3).

timal transport problem and are generally guaranteed to obtain good approximations of maximally dissimilar clusters. We then communicate the resulting global centroids with the clients, who use them to minimize the cross-entropy between the cluster assignments of a sample and its augmentations. By using the same set of global centroids across all clients, Orchestra's pipeline globally moves cluster members closer to each other, hence reducing δ every round. More details of the pipeline are provided in Figure 3.

Local Training: As noted by prior works in the centralized setting (Dizaji et al., 2017; Caron et al., 2018), the unstable training dynamics of clustering-based representation learning often yields degenerate solutions. This problem can arise during local training and disallows reduction of δ beyond a point. We now address this problem to complete the design of Orchestra (see Figure 3).

1. *Preventing Degenerate Solutions:* During local training, we minimize the KL-divergence between assignments of a sample and its augmentations. Early in the training process, the cluster centroids inevitably correspond to random features and therefore cannot yet yield a sufficiently discriminative signal for training, resulting in degenerate solutions. Centralized methods avoid this problem by adding a degeneracy regularizer in the form of a predictive SSL task that prevents the model from outputting a constant representation. These works primarily use denoising autoencoders (Xie et al., 2016; Chang et al., 2017; Yang et al., 2017) for this purpose, but other predictive SSL tasks can also be used, e.g., Rotation Prediction (Gidaris et al., 2018), Colorization (Vondrick et al., 2018), Jigsaw (Noroozi & Favaro, 2016), or Context Prediction (Doersch et al., 2015). However, except for rotation prediction, predictive SSL tasks can have high resource requirements because they require larger models (autoencoders, colorization) or computation of representations from several patches of an input (Jigsaw, Context Prediction). Rotation Prediction is best suited to our federated settings because it adds only an extra forward/backward pass and a linear layer worth of memory cost. Thus, every training step, we rotate each sample x in a batch to $\mathcal{R}(x,\theta)$ by sampling an angle θ from the set $\alpha = \{0°, 90°, 180°, 270°\}$ via uniform distribution $\mathcal{U}_4$. A linear layer $W_r \in \mathbb{R}^{D\times 4}$ is then trained to predict the sample's rotation from its representation. Specifically, if $W_{r,i}$ and α_i index W_r and α, we get the following objective.

$$\mathcal{L}^{(k)}_{\text{deg}} = \mathbb{E}_{x\in X^k, i\in\mathcal{U}_4}\left[\arg\max\left(W^T_{r,i} f(\mathcal{R}(x,\alpha_i))\right) = i\right] \quad (3)$$

2. *Predicting Assignments:* After training for a few iterations, the model representations change; possibly changing a sample's cluster assignment. This pushes the model to learn varying assignments for the same sample over time, forcing it to predict uniform assignments for all samples (Zhan et al., 2020). Prior works in centralized settings have proposed to solve this problem by calculating assignments alongside clusters and keeping them fixed until the next clustering operation, which happens every few iterations. In our setting, this solution would require more frequent communication between the server and the clients, which will be expensive. Instead, we propose to use two models: an online model that is trained using gradient descent and another model whose parameters are an exponential moving average (EMA) of the online model: $\mathrm{T}^t = m\mathrm{T}^{t-1} + (1-m)\mathrm{O}$, where m is a scalar close to 1, T denotes parameters of EMA model, and O denotes parameters of the online model. While centralized SSL works also use such EMA models (Grill et al., 2020; Caron et al., 2021), our primary motivation for this design choice is that its representations evolve slowly over time and hence its assignments remain consistent with the

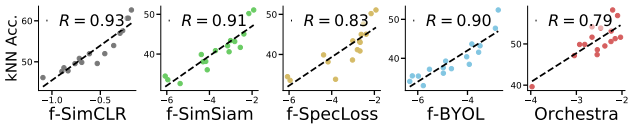


Figure 4. **Hyperparameter tuning.** We find linear combination of similarity scores (Equation 5) is highly predictive of kNN accuracy across all methods. Here, x-axis denotes Align + 0.2 * Unif and τ is set to 0.2, based on Wang and Isola (2020). For all methods, ResNet-18 models are trained on CIFAR-10 using different settings of learning rates (0.3, 0.01, 0.003, 0.001), heterogeneity (10 or $\sim$3.5 classes per client), and number of clients (10 or 100).

original ones. This yields the following loss function for promoting clusterability of representations:

$$\mathcal{L}^{(k)}_{\text{cluster}} = \mathbb{E}_{x\in X^k, \tilde{x}\sim\mathcal{T}(x)}\left[\mathcal{H}\left(P_{f_\text{T}}(x), P_{f_\text{O}}(\tilde{x})\right)\right]. \tag{4}$$

Notice that asymptotically, the target model and online model will share the exact same parameters. Our results in Prop. 3.2 and 3.3 are primarily designed for this regime, hence they continue to hold for Eq. 4.

5. Hyperparameters: Tuning our Instruments

Properly tuning hyperparmeters is of paramount importance in FL (Khodak et al., 2021). However, due to lack of labelled data, this can be particularly hard for unsupervised FL and can lead to conflicting results. E.g., we note that while Zhuang et al. (2021) find BYOL outperforms its competitors in federated settings, He et al. (2021) claim it collapses to degenerate solutions. This discrepancy likely stems from the use of a different EMA in the two works (0.99 vs. 0.9).

To avoid such inconsistencies and ensure fair comparisons, we propose to tune the hyperparameters of all methods in an unsupervised manner. Specifically, we compute the following two similarity scores:

$$\begin{aligned}\text{Align}(f) &= \mathbb{E}_{k\in[K]}\left[\mathbb{E}_{x\in X_k,\tilde{x}\sim\mathcal{T}(x)}\left[s(x,\tilde{x})\right]\right], \text{ and}\\ \text{Unif}(f,\tau) &= -\mathbb{E}_{k\in[K]}\left[\mathbb{E}_{x\in X_k}\left[\log \mathbb{E}_{y\in X_k}\left[e^{s(x,y)/\tau}\right]\right]\right],\end{aligned} \tag{5}$$

where $s(x,y) = f(x)^T f(y)/\|f(x)\|\|f(y)\|$ denotes cosine similarity of representations and τ is a vMF distribution parameter (Banerjee et al., 2005). Proposed by Wang and Isola (2020), the two scores are respectively large if the representations are similar for augmentations of a sample (high alignment) and dissimilar across samples (high uniformity).

Different versions of these scores show up in generalization bounds of SSL methods, under the assumption of task-relevant augmentations (Sanushi et al., 2022; Arora et al., 2019; Wei et al., 2021; Huang et al., 2021; HaoChen et al., 2021). Given this intricate relationship, we argue these scores are likely to be predictive of the quality of model representations. We verify this claim by plotting the kNN accuracy ($k = 200$) achieved by different methods in different settings as a function of a linear combination of the two scores. As shown in Figure 4, across all methods, the similarity scores are highly predictive of achieved accuracy (R = 0.87, on average). Thus, for all methods in our experiments, we propose to choose hyperparameters that maximize the linear combination of the similarity scores above.

6. Experiments: The Concert

This section compares Orchestra with federated versions of several discriminative SSL methods: SimCLR (Chen et al., 2020), SpecLoss (HaoChen et al., 2021), SimSiam (Chen & He, 2021), and BYOL (Grill et al., 2020). Results on rotation prediction (RotPred) (Gidaris et al., 2018) and supervised FedAvg (McMahan et al., 2017) are also provided. For cross-silo experiments, we use implementation tricks by recent self-supervised FL (Zhuang et al., 2021) papers to make our baselines even more competitive.

Setup. We use CIFAR-10/-100 datasets. To partition data across K clients, we sample class priors from a Dirichlet distribution (Hsu et al., 2019). A smaller Dirichlet parameter α yields more heterogeneous splits. All methods are implemented using PyTorch and the Flower framework (Beutel et al., 2020). We use ResNet-18 as the backbone architecture; Projector/Predictor architectures follow original papers. Orchestra uses a 2-layer Projector, but no Predictor. We compute 64/128 global clusters and 8/16 local clusters for CIFAR-10/-100. All results are averaged over 3 seeds. Standard deviations are shown in figures, but omitted from tables and deferred to the appendix due to space constraints. Learning rate is tuned for all SSL methods as per §5; for FedAvg, we borrow values from Charles et al. (2021). We tune the EMA value m for BYOL. Batch-size is set to 16 (256) for cross-device (cross-silo) settings. Unless stated otherwise, we set α to 0.1, number of local epochs E to 10, communication rounds C to 100, and participation ratio R to 0.5 (1.0) for cross-device (cross-silo) experiments. Please refer to §C for more details on the experiment setup.

Evaluation Protocol. We primarily use the standard linear probe protocol, where the model is frozen and a linear classifier is learned on top of the backbone (Chen et al., 2020). When comparing communication efficiency, we use a kNN accuracy probe (Chen & He, 2021). For semi-supervised evaluation, we fine-tune the entire model using limited labelled data (1% or 10% labels).

6.1. Linear and Semi-Supervised Evaluation

We first assess the accuracies of different methods using both a linear probe protocol and semi-supervised evaluation. Though our primary motivation in designing Orchestra is cross-device FL, for this experiment, we evaluate its per-

Table 1. Accuracy (%) in non-IID (α=0.1) cross-device (100 clients) and cross-silo (10 clients) settings on CIFAR datasets. For cross-device settings, due to a lack of baselines, we use FL-extensions of several centralized techniques; for cross-silo settings, we follow implementations of these techniques proposed by recent works that use stateful clients and divergence-aware predictor updates (Zhuang et al., 2021). We evaluate models using the popular linear probe technique and semi-supervised fine-tuning with 1%/10% labelled data. Consistent with prior work (Zhuang et al., 2021), we note that linear probe can outperform semi-supervised evaluation on CIFAR-10.

Dataset	CIFAR-10						CIFAR-100					
Setting	Cross-Device (K = 100)			Cross-Silo (K=10)			Cross-Device (K = 100)			Cross-Silo (K=10)		
	Linear	1%	10%	Linear	1%	10%	Linear	1%	10%	Linear	1%	10%
f-SimCLR	58.36	41.95	44.64	69.29	57.76	68.27	34.52	45.47	51.88	44.33	57.61	67.84
f-SimSiam	61.61	49.99	56.43	75.12	64.04	72.25	34.96	47.17	55.13	43.16	53.38	63.19
f-SpecLoss	66.51	55.66	62.09	80.71	70.88	77.96	37.60	47.11	50.91	**56.59**	62.15	72.09
f-BYOL	65.85	56.05	64.15	76.08	65.55	73.18	38.47	52.89	58.56	49.64	57.34	66.13
Orchestra	**71.58**	**60.33**	**66.20**	**82.14**	**71.30**	**79.51**	**40.37**	**54.01**	**59.07**	55.89	**63.73**	**73.06**
RotPred (pred)	44.44	34.71	46.15	55.68	45.84	51.32	16.85	15.79	19.52	25.0	27.64	28.81
FedAvg (sup)	80.85	82.76	80.34	79.22	86.81	87.12	58.71	59.07	64.01	65.59	62.48	71.59

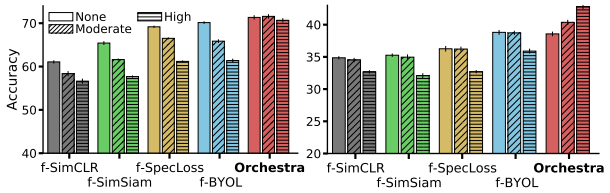


Figure 5. **Sensitivity to Statistical Heterogeneity** on CIFAR-10 (left) and CIFAR-100 (right). Except for Orchestra, we find all methods lose performance with increased heterogeneity.

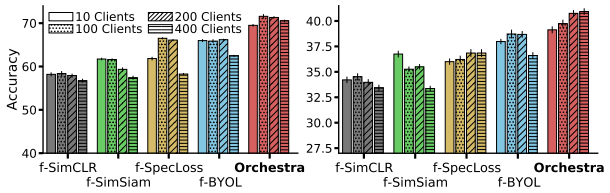


Figure 6. **Scalability with number of clients** on CIFAR-10 (left) and CIFAR-100 (right). Orchestra outperforms other methods and is generally robust to number of clients in the federation.

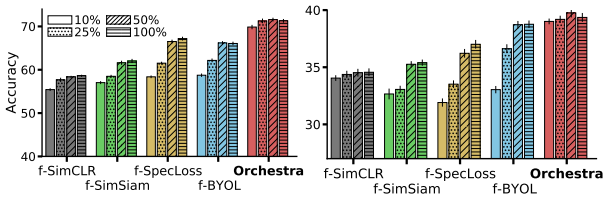


Figure 7. **Sensitivity to participation ratio** on CIFAR-10 (left) and CIFAR-100 (right). While the accuracies of alternative techniques decrease at smaller participation ratios, we find Orchestra suffers minimal degradation.

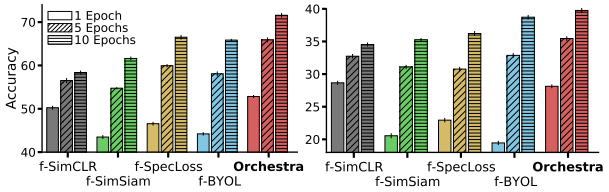


Figure 8. **Robustness to local epochs** on CIFAR-10 (left) and CIFAR-100 (right). We find Orchestra achieves similar accuracy to other methods in half the number of local epochs.

formance in cross-silo settings as well. Our cross-device setting has 100 clients, while the cross-silo setting has 10 clients. α is set to 0.1. Results are shown in Table 1.

We make several observations. (i) Orchestra often outperforms alternative techniques by a large margin under the linear probe protocol, where primarily the quality of learned representations is evaluated. (ii) Under semi-supervised settings, the gap reduces, but remains high when only 1% labels are used. (iii) Even though Orchestra was designed with cross-device settings in mind, we find that it also outperforms its competitors in cross-silo settings. We expect Orchestra's performance can be improved further with stateful operations and other enhancements allowed in cross-silo settings. This is left to future work.

6.2. Attributes of Federated Learning

Statistical Heterogeneity. We use 100 clients and analyze three levels of heterogeneity by setting α to 10^5 (none/IID), 10^{-1} (moderate), and 10^{-3} (high). Results are provided in Figure 5. We see that FL-extensions of centralized SSL methods are often sensitive to heterogeneity, especially on CIFAR-10. In contrast, Orchestra is robust and its performance generally improves with increase in heterogeneity. This observation matches our expectation from Prop. 3.3, where we showed Orchestra's theoretical guarantees improve under increased heterogeneity. We also note that since CIFAR-100 has fewer samples per class, increase in heterogeneity appears to enable greater gains via reduction in inconsistent assignments.

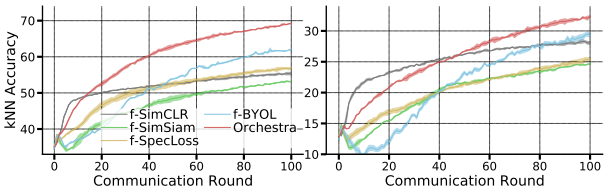


Figure 9. **Communication efficiency** on CIFAR-10 (left) and CIFAR-100 (right). We plot kNN accuracy w.r.t. comm. rounds. Orchestra has the fastest round-to-accuracy response for moderate to high accuracies; SimCLR performs well on smaller accuracies.

Number of Clients. We next consider the effects of changing the number of clients. Following prior work (Zhuang et al., 2021), we linearly scale the number of local epochs to ensure that the total number of training iterations remains constant across all settings. Results are shown in Figure 6. We find that unlike other methods, which suffer from fluctuations in performance depending on number of clients, Orchestra achieves similar performance for all settings.

Participation Ratio. Since only a fraction of participants can be expected to be connected to the server at a given time, participation ratio is an important attribute of cross-device FL. As shown in Figure 7, all methods except Orchestra suffer large decreases in accuracy when participation ratio is decreased. In contrast, Orchestra maintains accuracy even for the smallest participation ratios.

Local Epochs. Resource constraints may force one to limit local training to few epochs, making robustness to limited epochs a valuable attribute. Figure 8 shows that reducing local epochs causes all methods to lose accuracy, but Orchestra is the most resistant to this loss. For example, on CIFAR-10, Orchestra with 5 local epochs matches the performance of other methods with 10 local epochs, i.e., using Orchestra the training cost halves.

Communication Efficiency. Client-server communication is one of the major bottlenecks for FL in cross-device settings. Hence, it is ideal if an FL technique converges in fewer rounds. We measure the kNN accuracy of different methods during training and plot it as a function of the number of rounds. As shown in Figure 9, Orchestra takes the fewest rounds to achieve moderate to high accuracy; meanwhile, for smaller values, SimCLR is often the fastest to converge, but achieves poor ultimate accuracy.

6.3. Attributes Specific to Orchestra

Effect of Heterogeneity on Communication Efficiency. Our theoretical result indicates and empirical results corroborate the expectation that Orchestra thrives under heterogeneity. We provide another demonstration of this result in Figure 10, where we show the progress of kNN accuracy as a model is trained using Orchestra. We find that not only does heterogeneity help Orchestra improve performance, it can also improve its speed of optimization.

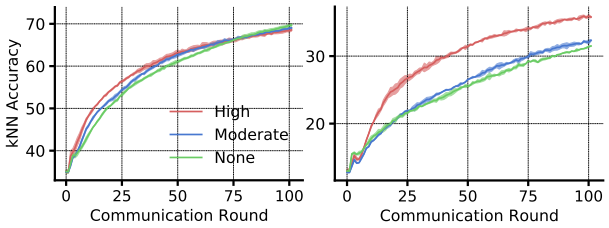


Figure 10. **Heterogeneity improves communication efficiency** for Orchestra on CIFAR-10 (left) and CIFAR-100 (right), i.e., Orchestra thoroughly exploits heterogeneity to maximize efficiency.

Table 2. **Sensitivity to number of global (G) and local (L) clusters.** We change G and L one-by-one, keeping the other fixed. Base values are reported in parentheses. "Reps" denotes the idealized setting, where representations are shared without local clustering. As shown, Orchestra has minimal sensitivity to both, the number of global and local clusters. Further, the performance is generally very close to the idealized setting.

	CIFAR-10 (G=32, L=8)				CIFAR-100 (G=256, L=16)			
G	8	16	64	128	64	128	256	512
Acc	70.25	71.05	71.04	71.38	39.89	40.27	40.37	40.29
L	2	4	16	Reps	8	16	64	Reps
Acc	70.41	71.06	71.28	71.95	40.09	40.37	40.25	41.19

Number of Clusters. Orchestra's underlying principles are rooted in Prop. 3.2 and 3.3, which provide the goal of finding discriminable clusters. However, another important attribute in those bounds is the number of global clusters G. If the G is smaller than the number of classes, generalization can suffer. Similarly, having few local clusters L can lead to inconsistent assignments (larger c) and hurt generalization. More local clusters can help avoid this, but if the number approaches the size of the local dataset, privacy (anonymity) is lost because all clusters will contain only one datapoint. We thus study sensitivity of Orchestra to G and L. Table 2 shows that Orchestra is robust to the number of global clusters as long as it even slightly exceeds the number of classes, achieving similar performance in all settings. We similarly see that Orchestra is essentially robust to the number of local centroids, but can see minimal performance loss if very few clusters are used. Finally, these results also indicate the values of G and L need not be precisely tuned, as most settings yield good performance.

7. Conclusion

To enable wider adoption of federated learning (FL) for complex modalities such as vision, we need to reduce its reliance on labeled data. Towards this goal, we presented *Orchestra*, an unsupervised FL technique that orchestrates a distributed clustering task and enforces a globally consistent partitioning of clients' data, while remaining mindful of the core challenges seen in FL setups. Built on strong theoretical foundations, Orchestra is scalable, achieves strong communication-efficiency, thrives under heterogeneity, and remains robust to various FL parameters.

Acknowledgements

We thank the teams at Nokia Bell Labs, UK and Flower.dev for several useful discussions during the course of this project. We also thank the reviewers and AC for multiple useful suggestions that helped improve the paper. ESL's time at University of Michigan was partly supported via NSF award CNS-2008151.

References

Acar, D. A. E., Zhao, Y., Matas, R., Mattina, M., Whatmough, P., and Saligrama, V. Federated Learning Based on Dynamic Regularization. In *Proc. Int. Conf. on Learning Representations (ICLR)*, 2021.

Aggarwal, C. C. On k-anonymity and the curse of dimensionality. In *Proc. Conf. on Very Large Data Bases (VLDB)*, 2005.

Aggarwal, G., Panigrahy, R., Feder, T., Thomas, D., Kenthapadi, K., Khuller, S., and Zhu, A. Achieving Anonymity via Clustering. *ACM Trans. on Algorithms*, 2010.

Arora, S., Khandeparkar, H., Khodak, M., Plevrakis, O., and Saunshi, N. A Theoretical Analysis of Contrastive Unsupervised Representation Learning. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2019.

Asano, Y. M., Rupprecht, C., and Vedaldi, A. Self-labelling via simultaneous clustering and representation learning. In *Proc. Int. Conf. on Learning Representations (ICLR)*, 2020.

Balcan, M.-F., Blum, A., and Yang, K. Co-Training and Expansion: Towards Bridging Theory and Practice. In *Proc. Adv. on Neural Information Processing Systems (NeurIPS)*, 2004.

Balcan, M.-F., Dick, T., Liang, Y., Mou, W., and Zhang, H. Differentially Private Clustering in High-Dimensional Euclidean Spaces. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2017.

Banerjee, A., Dhillon, I., Ghosh, J., and Sra, S. Clustering on the Unit Hypersphere using von Mises-Fisher Distributions. *Journal of Machine Learning Research (JMLR)*, 2005.

Bardes, A., Ponce, J., and LeCun, Y. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. *arXiv*, abs/2105.04906, 2021.

Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Parcollet, T., de Gusmão, P. P., and Lane, N. D. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In *Proc. ACM SIGSAC Conf. on Computer and Communications Security (CCS)*, 2017.

Bureau, U. C. Census Bureau Sets Key Parameters to Protect Privacy in 2020 Census Results, 2020. URL https://www.census.gov/newsroom/press-releases/2021/2020-census-key-parameters.html.

Byun, J.-W., Kamra, A., Bertino, E., and Li, N. Efficient K-Anonymization Using Clustering Techniques. In *Proc. ACM Int. Conf. on Database Systems (DASFAA)*, 2007.

Caron, M., Bojanowski, P., Joulin, A., and Douze, M. Deep Clustering for Unsupervised Learning of Visual Features. In *Proc. Euro. Conf. on Computer Vision (ECCV)*, 2018.

Caron, M., Bojanowski, P., Mairal, J., and Joulin, A. Unsupervised Pre-Training of Image Features on Non-Curated Data. In *Proc. Int. Conf. on Computer Vision (ICCV)*, 2019.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *Proc. Adv. on Neural Information Processing Systems (NeurIPS)*, 2020.

Caron, M., Touvron, H., Misra, I., Jegou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging Properties in Self-Supervised Vision Transformer. *arXiv*, abs/2104.14294, 2021.

Chang, A., Ghazi, B., Kumar, R., and Manurangsi, P. Locally Private k-Means in One Round. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2021.

Chang, J., Wang, L., Meng, G., Xiang, S., and Pan, C. Deep Adaptive Image Clustering. In *Proc. Int. Conf. on Computer Vision (ICCV)*, 2017.

Charles, Z., Garrett, Z., Huo, Z., Shmulyian, S., and Smith, V. On Large-Cohort Training for Federated Learning. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*, 2021.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2020.

Chen, X. and He, K. Exploring Simple Siamese Representation Learning. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Cohen, E., Kaplan, H., Mansour, Y., Stemmer, U., and Tsfadia, E. Differentially-private clustering of easy instances. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2021.

Dennis, D. K., Li, T., and Smith, V. Heterogeneity for the Win: One-Shot Federated Clustering. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2021.

Dhillon, I., Guan, Y., and KulisHenaff, B. Kernel k-means: spectral clustering and normalized cuts. In *Proc. Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, 2004.

Dizaji, K. G., Herandi, A., Deng, C., Cai, W., and Huang, H. Deep Clustering via Joint Convolutional Autoencoder Embedding and Relative Entropy Minimization. In *Proc. Int. Conf. on Computer Vision (ICCV)*, 2017.

Doersch, C., Gupta, A., and Efros, A. A. Unsupervised Visual Representation Learning by Context Prediction. In *Proc. Int. Conf. on Computer Vision (ICCV)*, 2015.

Dosovitskiy, A. and Brox, T. Inverting Visual Representations with Convolutional Networks. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Dwork, C. and Roth, A. *The Algorithmic Foundations of Differential Privacy*, volume 9. Now Publishers Inc., 2014.

Fard, M. M., Thonet, T., and Gaussier, E. Deep k-Means: Jointly clustering with k-Means and learning representations. *arXiv*, abs/1806.10069, 2018.

Genevay, A., Dulac-Arnold, G., and Vert, J.-P. Differentiable Deep Clustering with Cluster Size Constraints. *arXiv*, abs/1910.09036, 2019.

Ghosh, A., Chung, J., Yin, D., and Ramchandran, K. An Efficient Framework for Clustered Federated Learning. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*, 2020.

Gidaris, S., Singh, P., , and Komodakis, N. Unsupervised Representation Learning by Predicting Image Rotations. In *Proc. Int. Conf. on Learning Representations (ICLR)*, 2018.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., Piot, B., kavukcuoglu, k., Munos, R., and Valko, M. Bootstrap your own latent: A new approach to self-supervised Learning. In *Proc. Adv. on Neural Information Processing Systems (NeurIPS)*, 2020.

HaoChen, J., Wei, C., Gaidon, A., and Ma, T. Provable Guarantees for Self-Supervised Deep Learning with Spectral Contrastive Loss. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*, 2021.

He, C., Yang, Z., Mushtaq, E., Lee, S., Soltanolkotabi, M., and Avestimehr, S. SSFL: Tackling Label Deficiency in Federated Learning via Personalized Self-Supervision. *arXiv*, abs/2110.02470, 2021.

He, K., Fan, H., Wu, Y., Xie, S., and Girschick, R. Momentum Contrast for Unsupervised Visual Representation Learning. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Hsieh, K., Phanishayee, A., Mutlu, O., and Gibbons, P. B. The Non-IID Data Quagmire of Decentralized Machine Learning. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2020.

Hsu, H., Qi, H., and Brown, M. Measuring the Effects of Non-Identical Data Distribution for Federated Visual Classification. *arXiv*, abs/1909.06335, 2019.

Huang, J., Gong, S., and Zhu, X. Deep Semantic Clustering by Partition Confidence Maximisation. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Huang, W., Yi, M., and Zhao, X. Towards the Generalization of Contrastive Self-Supervised Learning. *arXiv*, abs/2111.00743, 2021.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konecný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Qi, H., Ramage, D., Raskar, R., Raykova, M., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. Advances and Open Problems in Federated Learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2020.

Khodak, M., Tu, R., Li, T., Li, L., Balcan, M.-F., Smith, V., and Talwalkar, A. Federated Hyperparameter Tuning: Challenges, Baselines, and Connections to Weight-Sharing. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*, 2021.

Konečný, J., McMahan, H. B., Yu, F. X., Richtarik, P., Suresh, A. T., and Bacon, D. Federated Learning: Strategies for Improving Communication Efficiency.

In *NeurIPS Workshop on Private Multi-Party Machine Learning*, 2016.

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

Kugelgen, J., Sharma, Y., Gresle, L., Brendel, W., Scholkopf, B., Besserve, M., and Locatello, F. Self-Supervised Learning with Data Augmentations Provably Isolates Content from Style. *arXiv*, abs/2106.04619, 2021.

Lai, F., Zhu, X., Madhyastha, H. V., and Chowdhury, M. Oort: Efficient Federated Learning via Guided Participant Selection. In *Proc. USENIX Sym. on Operating Systems Design and Implementation (OSDI)*, 2021.

LeCun, Y. Tutorial on Self-Supervised Learning, 2019. URL https://www.youtube.com/watch?v=SaJL4SLfrcY.

LeFevre, K., Dewitt, D., and Ramakrishnan, R. Mondrian Multidimensional K-Anonymity. In *Proc. IEEE Int. Conf. on Data Engineering (ICDE)*, 2006.

Li, J., Zhou, P., Xiong, C., and Hoi, S. Prototypical Contrastive Learning of Unsupervised Representations. In *Proc. Int. Conf. on Learning Representations (ICLR)*, 2021a.

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated Learning: Challenges, Methods, and Future Directions. *arXiv*, abs/1908.07873, 2019.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated Optimization in Heterogeneous Networks. In *Proc. Conf. on Machine Learning and Systems (MLSys)*, 2020a.

Li, T., Sanjabi, M., Beirami, A., and Smith, V. Fair Resource Allocation in Federated Learning. In *Proc. Int. Conf. on Learning Representations (ICLR)*, 2020b.

Li, T., Hu, S., Beirami, A., and Smith, V. Ditto: Fair and Robust Federated Learning Through Personalization. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2021b.

Lu, N., Wang, Z., Li, X., Niu, G., Dou, Q., and Sugiyama, M. Federated Learning from Only Unlabeled Data with Class-conditional-sharing Clients . In *Int. Conf. on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=WHA8009laxu.

Mahendran, A. and Vedaldi, A. Understanding Deep Image Representations by Inverting Them. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

McLachlan, G. and Krishnan, T. *The EM Algorithm and Extensions*, volume 2. Wiley, 2008.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proc. Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, 2017.

Nash, C., Kushman, N., and Williams, C. K. I. Inverting Supervised Representations with Autoregressive Neural Density Models. In *Proc. Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, 2019.

Newman, M. Modularity and community structure in networks. In *Proc. Natl. Acad. of Sciences (PNAS)*, 2008.

Noroozi, M. and Favaro, P. Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles. In *Proc. European Conf. on Computer Vision (ECCV)*, 2016.

Purushwalkam, S. and Gupta, A. Demystifying Contrastive Self-Supervised Learning: Invariances, Augmentations and Dataset Biases. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*, 2020.

Quadrianto, N., Smola, A. J., Caetano, T. S., and Le, Q. V. Estimating Labels from Label Proportions. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2008.

Reddi, S. J., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečný, J., Kumar, S., and McMahan, H. B. Adaptive Federated Optimization. In *Proc. Int. Conf. on Learning Representations (ICLR)*, 2021.

Sanushi, N., Ash, J., Goel, S., Misra, D., Zhang, C., Arora, S., Kakade, S., and Krishnamurthy, A. Understanding Contrastive Learning Requires Incorporating Inductive Biases. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2022.

Smith, V., chiang, C.-K., Sanjabi, M., and Talwalkar, A. Federated Multi-Task Learning. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*, 2017.

Stemmer, U. Locally Private k-Means Clustering. In *Proc. Sym. on Discrete Algorithms (SODA)*, 2020.

Tian, Y., Sun, C., Poole, B., Krishnan, D., and Isola, P. What Makes for Good Views for Contrastive Learning? In *Proc. Adv. on Neural Information Processing Systems (NeurIPS)*, 2020.

Tian, Y., Chen, X., and Ganguli, S. Understanding self-supervised Learning Dynamics without Contrastive Pairs. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2021a.

Tian, Y., Henaff, O., and Van Der Oord, A. Divide and Contrast: Self-supervised Learning from Uncurated Data. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021b.

Tsai, Y.-H., Wu, Y., Salakhutdinov, R., and Morency, L.-P. Self-supervised Learning from a Multi-view Perspective. In *Proc. Int. Conf. on Learning Representations (ICLR)*, 2021.

Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, A., and Murphy, K. Tracking Emerges by Colorizing Videos. In *Proc. European Conf. on Computer Vision (ECCV)*, 2018.

Wang, T. and Isola, P. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2020.

Wei, C., Shen, K., Chen, Y., and Ma, T. Theoretical Analysis of Self-Training with Deep Networks on Unlabeled Data. In *Proc. Int. Conf. on Learning Representations (ICLR)*, 2021.

Wen, Z. and Li, Y. Towards Understanding the Feature Learning Process of Self-supervised Contrastive Learning. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2021.

Wu, Y., Wang, Z., Zeng, D., Li, M., Shi, Y., and Hu, J. Distributed Unsupervised Visual Representation Learning with Fused Features. *arXiv*, abs/2111.10763, 2021.

Wu, Y., Wang, Z., Zeng, D., Li, M., Shi, Y., and Hu, J. Federated contrastive representation learning with feature fusion and neighborhood matching, 2022. URL https://openreview.net/forum?id=6LNPEcJAGWe.

Wu, Z., Xiong, Y., Yu, S., and Lin, D. Unsupervised Feature Learning via Non-Parametric Instance-level Discrimination. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Xie, J., Girshick, R., and Farhadi, A. Unsupervised Deep Embedding for Clustering Analysis. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2016.

Yang, B., Fu, X., Sidiropoulos, N. D., and Hong, M. Towards K-means-friendly Spaces: Simultaneous Deep Learning and Clustering. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2017.

Zbontar, J., Jing, L., Misra, I., Lecun, Y., and Deny, S. Barlow Twins: Self-Supervised Learning via Redundancy Reduction. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2021.

Zhan, X., Xie, J., Liu, Z., Ong, Y. S., and Loy, C. C. Online Deep Clustering for Unsupervised Representation Learning. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Zhang, F., Kuang, K., You, Z., Shen, T., Xiao, J., Zhang, Y., Wu, C., Zhuang, Y., and Li, X. Federated Unsupervised Representation Learning. *arXiv*, abs/2010.08982, 2020.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. Federated Learning with Non-IID Data. *arXiv*, abs/1806.00582, 2018.

Zhuang, W., Gan, X., Wen, Y., Zhang, S., and Yi, S. Collaborative Unsupervised Visual Representation Learning from Decentralized Data. In *Proc. Int. Conf. on Computer Vision (ICCV)*, 2021.

Zhuang, W., Wen, Y., and Zhang, S. Divergence-aware Federated Self-Supervised Learning. In *Int. Conf. on Learning Representations*, 2022. URL https://openreview.net/forum?id=oVE1z8NlNe.

Appendix

In this appendix, we provide more description about the federated learning settings studied in this paper (§A), and contextualize prior works based on FL properties studied in them (§B). Thereafter, we provide the formal algorithm for Orchestra and elaborate on our experiment, training, and evaluation setups (§C). Next, §D contains T-SNE visualizations, ablation results, and shows the performance of Orchestra under a state-of-the-art differentially private local clustering technique (Chang et al., 2021). The appendix ends by providing a recap of the key notations used in the paper and proofs for Propositions 3.2 and 3.3.

A. Properties of Cross-Silo and Cross-Device FL

Federated learning setups vary across different applications and circumstances. In general, they can be categorized into two main settings: *Cross-silo* and *Cross-device*, as described in a recent federated learning survey by (Kairouz et al., 2021).

The cross-silo setting shares several properties of datacenter distributed learning, where the number of clients is often limited, and the clients have a large amount of data and high-level of computational resources, are stateful, almost always available with few failures. However, these requirements often are not applicable to many real-life FL setups, and the cross-silo setting is more relevant when large organizations collaborate.

On the other hand, the cross-device setting has much fewer requirements on the clients: they can be any number of mobile or IoT devices which have limited computational resources, be stateless, unavailable at any time, and unreliable. The relaxation on the requirement of clients make it more applicable to real-world FL setups, but any proposed solution must tackle the challenges that this setting brings.

Many existing centralized unsupervised representation learning algorithms are relatively straight-forward to be adapted to work in a cross-silo setup, where the clients are often computationally powerful and can be assumed as stateful. However, clients in the cross-device setting often have much less resources, and this brings many challenges in adapting centralized algorithms. For example, the requirement of large batch-sizes (He et al., 2021) is difficult to meet in the cross-device setting as mobile or IoT devices have limited runtime memory. Similarly, sharing representations across clients as done in (Wu et al., 2021; Zhang et al., 2020) is disallowed due to user privacy concerns. Hence, designing a method which scales to hundreds of participating clients, is stateless, works with small batch sizes and uneven distribution of data, and preserves user privacy is crucial to the success of *cross-device* unsupervised representation learning. This was the primary motivation behind the design of Orchestra.

	Properties					
Methods	Stateless	Privacy Preserving	Supports small Batch Size	# Clients	Cross-silo	Cross-Device
FedCA (Zhang et al., 2020)	✓	×	× (128)	5	✓	×
FedU (Zhuang et al., 2021)	×	✓	× (128)	5	✓	×
FedEMA (Zhuang et al., 2022)	×	✓	× (128)	5	✓	×
SSFL (He et al., 2021)	✓	✓	× (256)	10	✓	×
FCL (Wu et al., 2022)	✓	Using an additional encryption module	× (128)	5-10	✓	×
Orchestra	✓	✓	✓ (16)	**10-400**	✓	✓

Table 3. Comparison of Orchestra with prior federated unsupervised learning approaches. Orchestra is unique in its capability to learn from unlabeled data in a small batch size regime without requiring any stateful operations or sharing local representations with the server. Moreover, unlike prior approaches which were evaluated only in small scale setups with 5-10 clients, Orchestra can easily scale to large-scale cross-device settings with hundreds of participating clients.

B. More Related Work

In Table 3, we list various unsupervised FL approaches proposed in the recent literature and compare their properties with Orchestra. As evident, Orchestra is unique in its capability to learn from unlabeled data in a *small batch size* regime without requiring any stateful operations or sharing local representations with the server. Moreover, unlike prior approaches which

were evaluated only in small scale setups with 5-10 clients, Orchestra can easily scale to large-scale settings with hundreds of participating clients, making it particularly apt for *cross-device* FL.

Self-Supervised Learning: SSL techniques are a recent paradigm in centralized unsupervised learning, wherein a task-relevant signal is extracted from the data itself and used to guide the training of a model. In vision, specifically, this task-relevance prior is encoded by designing pretext tasks. Earlier examples of such tasks include predictive tasks, such as rotation prediction (Gidaris et al., 2018) or predicting the patch order in a shuffled image (Noroozi & Favaro, 2016). Recently, though, the task of instance discrimination has revolutionized unsupervised visual training (Wu et al., 2018). Herein, task-relevant information in encoded by enforcing invariances to a set of data augmentations (Purushwalkam & Gupta, 2020; Wen & Li, 2021; Kugelgen et al., 2021). Popular methods include contrastive techniques (SimCLR (Chen et al., 2020), MoCo (He et al., 2020), SpecLoss (HaoChen et al., 2021)); similarity-based techniques (BYOL (Grill et al., 2020), SimSiam (Chen & He, 2021), DINO (Caron et al., 2021)); redundancy reduction methods (Barlow Twins (Zbontar et al., 2021), VICReg (Bardes et al., 2021)); and clustering based methods (SeLa (Asano et al., 2020), SwAV (Caron et al., 2020), PCL (Li et al., 2021a)).

C. Experimental Details

C.1. Data

Datasets. Our experiments focus on the widely-used CIFAR-10 and CIFAR-100 datasets (Krizhevsky & Hinton, 2009). Both datasets consist of 60,000 images, divided into two partitions: 50,000 for training and 10,000 for testing. CIFAR-10 consists of 10 object classes whereas CIFAR-100 have 100 object classes; with training samples equally distributed across all classes. For federated training, we partitioned both the datasets across K clients. To this end, we sample class priors from a Dirichlet distribution (Hsu et al., 2019) controlled by the Dirichlet parameter α. A smaller value of α yields more non-IID splits across clients. More specifically, we experimented with three different levels of heterogeneity by setting α to 10^5 (IID), 10^{-1} (moderately non-IID), and 10^{-3} (highly non-IID). To quantify the level of heterogeneity, we provide average number of classes in Table 4. We consider two definitions for a class to be present on a client: if at least 1 sample from the class is present on the client and if at least 1% samples belong to the class. $\sim$ denotes an experiment that was not conducted.

Transformations. During the local training stage shown in Figure 3, we need to generate an augmentation of each input sample. We use the same set of augmentations proposed by Grill et al. (2020). For the baseline SSL methods, we follow the augmentations proposed in their original papers.

Table 4. **Average number of classes per client.** We use three values of α: 10^5 (IID), 10^{-1} (moderately non-IID), and 10^{-3} (highly non-IID). We define a class to be present on a client in two ways: (i) at least 1 sample from the class is present on the client; (ii) at least 1% samples on the client belong to the class. $\sim$ denotes an experiment that was not conducted.

		CIFAR-10			CIFAR-100		
	α	10^{-3}	10^{-1}	10^{5}	10^{-3}	10^{-1}	10^{5}
	Heterogeneity	High	Moderate	None	High	Moderate	None
Cross-Device	1 sample	1.08	4.69	10	3.56	29.3	100
(100 clients)	1 %	1.05	3.13	10	2.13	17.92	100
Cross-Silo	1 sample	$\sim$	6.25	10	$\sim$	48.63	100
(10 clients)	1 %	$\sim$	4.87	10	$\sim$	35.5	100

C.2. Algorithm, Implementation, and Training Details

Algorithm: Below we provide a detailed algorithm of our pipeline, as described in §4 and outlined in Figure 3. Orchestra involves federation and communication between clients and the server, and Algorithm 1 describes local training happening on the clients, while Algorithm 2 outlines the computation happening on the server. We refer to the backbone and projector models jointly as the 'feature encoder'. Broadly, each client trains its feature encoders locally with both clustering and degeneracy losses, and returns the local clusters. The server aggregates the feature encoders from all clients using any federated averaging algorithm (e.g., FedAvg), and performs further clustering on the local centroids returned by all the clients to obtain global centroids. The global centroids are then passed back to the clients for another round of local training.

When Orchestra starts, the server has to initialize the global centroids to be used for training (Line 2 of Algorithm 2). However, as the server does not have any prior data about the clients, the initial global centroids are initialized with a small federation step with the clients: the initialized feature encoder f_T^G is passed to the clients, and the clients compute the representations by passing the local data through the encoder (similar to Line 5 of Algorithm 1). These representations are then clustered (Line 18 of Algorithm 1) and the centroids are sent to the server, where a further clustering is performed (Line 8 of Algorithm 2), forming the initial global centroids μ^G.

The clustering algorithm $\mathcal{C}$ used in both the local and global clustering processes is based on the Sinkhorn-knopp algorithm (Genevay et al., 2019). We run local clustering on a memory module that stores 128 most recent representations computed during local training, i.e., representations computed during last 8 iterations. This allow us to avoid the cost of computing model representations for local clustering again. Since we use the target model for clustering purposes, these representations can be expected to be essentially the same as the representations that the model with latest parameters would compute. Combined, this trick makes local clustering *very efficient, with no extra inference/memory costs*. In fact, we demonstrate this in Figure 11 by successfully deploying Orchestra on three NVIDIA Jetson embedded devices (with RAM as low as 4GB). We plot time consumed by a client running other methods, relative to when it runs Orchestra, and percent time consumed by local clustering in a round of Orchestra. As can be seen, Orchestra's latency per round is similar to other methods, and clustering accounts for $\leq 0.009\%$ of the training cost, *confirming Orchestra's practicality for cross-device FL.*

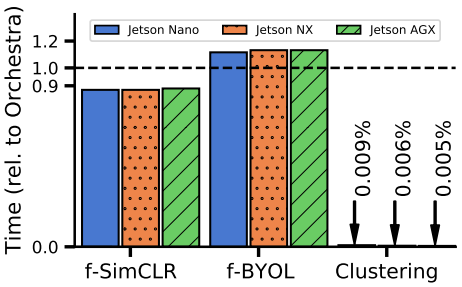


Figure 11. Client runtime per round, relative to Orchestra.

Implementation and Code: Orchestra is implemented using PyTorch and the Flower federated learning framework (Beutel et al., 2020). Our primary results in cross-device FL settings are presented for $K = 100$ clients, which we simulate on 8 NVIDIA V100 GPUs using Flower's Virtual Client Engine. Consistent with the paradigm of cross-device FL, we use a small batch size of 16 on each client, set the number of local epochs E to 10, communication rounds to 100, and participation ratio to 0.5. For cross-silo experiments, a batch size of 256 and participation ratio of 1.0 is employed. While evaluating the scalability and communication efficiency of Orchestra in cross-device settings (§6.2), we also show results for different values of K and E.

A working source code of Orchestra is provided available at following github link.

Algorithm 1 Orchestra - Local Training

Require: Data X^k on client k, global centroids μ^G, batch size N, exponential update rate m, target encoder f_T, online encoder f_O, stochastic augmentation function $\mathcal{T}$, rotation function $\mathcal{R}$, clustering function with size constraints $\mathcal{C}$
for sampled minibatch $\{x\}_{i=1}^N \subset X^k$ **do**
  for $i \in \{1, \ldots, N\}$ **do**
    Compute transformed sample: $\tilde{x_i} \leftarrow \mathcal{T}(x_i)$
    Compute representations: $f_\text{T}(x_i)$ and $f_\text{O}(\tilde{x_i})$
    Compute cluster assignment according to global centroids: $P_{f_\text{T}}(x_i)$ and $P_{f_\text{O}}(\tilde{x_i})$
    Select an random rotation angle index: $a_i \sim \mathcal{U}_4$
    Convert to one hot encoding: $R_i \leftarrow \text{onehot}(a_i)$
    Rotate sample: $\hat{x_i} \leftarrow \mathcal{R}(x_i, \alpha_{a_i})$
    Compute representation: $f_\text{O}(\hat{x_i})$
    Predict rotation: $\hat{R_i} \leftarrow \text{softmax}\left(W_r^T f_O(\hat{x_i})\right)$
  end for
  Compute clustering loss: $\mathcal{L}_\text{cluster} = -\frac{1}{N}\sum_{i=1}^N(\mathcal{H}\left(P_{f_\text{T}}(x_i), P_{f_\text{O}}(\tilde{x_i})\right))$
  Compute degeneracy loss: $\mathcal{L}_\text{deg} = -\frac{1}{N}\sum_{i=1}^N(\mathcal{H}(R_i, \hat{R_i}))$
  Update f_O to minimize $\mathcal{L} = \mathcal{L}_\text{cluster} + \mathcal{L}_\text{deg}$
  Update $f_\text{T} \leftarrow m f_\text{T} + (1-m) f_\text{O}$
end for
Cluster $\mu_k^L \leftarrow \mathcal{C}(\{f_\text{T}(x) | x \in X^k\}, L^{(k)})$
return trained encoders f_T, f_O, local clusters μ_k^L

Algorithm 2 Orchestra - Server Federation

Require: Number of rounds N, Number of global centroids G, clustering function with size constraints $\mathcal{C}$
Initialize global target encoder f_{T}^{G}, global online encoder f_{O}^{G}, global centroids μ^{G}
for round $i \in \{1, \ldots, N\}$ **do**
  Orchestrate local training with $f_{\mathrm{T}}^{G}, f_{\mathrm{O}}^{G}, \mu^{G}$
  Collect results: $\{(f_{\mathrm{T}}^{k}, f_{\mathrm{O}}^{k}, \mu_{k}^{L}) | k \in [K]\}$
  Aggregate encoders: $f_{\mathrm{T}}^{G} \leftarrow \frac{1}{K}\sum_{i=1}^{K} f_{\mathrm{T}}^{k}$ and $f_{\mathrm{O}}^{G} \leftarrow \frac{1}{K}\sum_{i=1}^{K} f_{\mathrm{O}}^{k}$
  Aggregate centroids: $\mu^{L} \leftarrow \{\mu | k \in [K], \mu \in \mu_{k}^{L}\}$
  Update global centroids by clustering: $\mu^{G} \leftarrow \mathcal{C}(\mu^{L}, G)$
end for
return trained encoder f_{T}^{G}

C.3. Network Architectures

Orchestra uses ResNet-18 as the network architecture for the Backbone, and a 2-layer multi-layer perceptron (MLP) with 512 units in each layer for the Projector, following SimCLR (Chen et al., 2020). No Predictor network is used. For all the baselines, we use ResNet-18 as the Backbone, while Projector and Predictor architectures are borrowed from their respective papers.

C.4. Hyperparameter Tuning and Scaling Schemes

Tuning: As mentioned in section 5, we find configuration that maximize a linear combination of alignment and uniformity scores (see Equation 5). We set τ to 0.2, similar to Figure 4. Tuning is performed by running a model for 20 communication rounds with 10 local epochs. Other settings such as batch-size and participation ratio depend on whether we are in a cross-silo or a cross-device setup and are defined above. In the following, we report our hyperparameter grids and the retrieved values from tuning different methods. We denote learning rate using η and EMA value using m.

1. SimCLR: η: {0.03 (Cross-silo), 0.01, 0.003 (Cross-device), 0.001}
2. SimSiam: η: {0.03 (Cross-silo), 0.01 (Cross-device), 0.003, 0.001}
3. SpecLoss: η: {0.03 (Cross-silo), 0.01, 0.003 (Cross-device), 0.001}
4. BYOL: η: {0.03 (Cross-silo), 0.01 (Cross-device), 0.003, 0.001}; m: {0.9, 0.99 (Cross-silo), 0.996 (Cross-device)}
5. Orchestra: η: {0.03, 0.01 (Cross-silo), 0.003 (Cross-device), 0.001}; m: {0.9, 0.99 (Cross-silo), 0.996 (Cross-device)}

Scaling Schemes for number of clients and participation ratio experiments: We tune our methods for two baseline settings: (i) Cross-device with 100 clients; (ii) Cross-silo with 10 clients. For experiments where we change other number of clients and participation ratio, we follow previous works and scale number of local epochs or learning rate to avoid the costs of tuning a method again. Specifically, we have the following schemes.

- **Number of Clients:** When varying number of clients, we follow Zhuang et al. (2021) and linearly scale number of local epochs. Given other variables remain fixed, this ensures a constant training budget in terms of number of iterations. For example, if L is our base learning rate for a setting with K clients, we scale the number of local epochs for a setting with K_{new} clients as follows: $L_{\text{new}} = L \cdot \frac{K_{\text{new}}}{K}$.
- **Participation Ratio:** When varying participation ratio, we follow Charles et al. (2021) and quadratically scale the learning rate. Given other variables remain fixed, this helps achieve consistent performance across a large number of settings for number of clients. For example, if η is our base learning rate for a setting with R participation ratio, we scale the learning rate for a setting with R_{new} participation ratio as follows: $\eta_{\text{new}} = \eta \cdot \sqrt{\frac{R_{\text{new}}}{R}}$.

C.5. Evaluation Protocol

To evaluate the quality of the representations learned by Orchestra, we primarily use the standard linear probe protocol, where the model is frozen and a linear classifier is learned on top of the backbone (Chen et al., 2020). When comparing

Table 5. We analyze Orchestra in the cross-device and cross-silo setting on CIFAR-10/CIFAR-100 datasets. For cross-device settings, due to a lack of baselines, we use FL-extensions of several centralized techniques; for cross-silo settings, we follow implementations of these techniques proposed by recent works that use stateful clients and divergence-aware predictor updates (Zhuang et al., 2021). We evaluate models using the popular linear probe technique (Chen et al., 2020) and semi-supervised fine-tuning with 1% and 10% labelled data.

Dataset	CIFAR-10					
Setting	Cross-Device (K = 100)			Cross-Silo (K=10)		
	Linear	1%	10%	Linear	1%	10%
f-SimCLR	58.36 ± 0.19	41.95 ± 0.85	44.64 ± 0.71	69.29 ± 0.28	57.76 ± 0.33	68.27 ± 0.67
f-SimSiam	61.61 ± 0.68	49.99 ± 0.26	56.43 ± 0.52	75.12 ± 0.38	64.04 ± 0.41	72.25 ± 0.37
f-SpecLoss	66.51 ± 0.53	55.66 ± 0.80	62.09 ± 0.67	80.71 ± 0.31	70.88 ± 0.53	77.96 ± 0.55
f-BYOL	65.85 ± 0.19	56.05 ± 0.31	64.15 ± 0.30	76.08 ± 0.30	65.55 ± 0.34	73.18 ± 0.40
Orchestra	**71.58 ± 0.53**	**60.33 ± 0.63**	**66.20 ± 0.71**	**82.14 ± 0.38**	**71.30 ± 0.27**	**79.51 ± 0.51**
RotPred (pred)	44.44 ± 0.93	34.71 ± 0.69	46.15 ± 0.65	55.68 ± 0.38	45.84 ± 0.53	51.32 ± 0.55
FedAvg (sup)	80.85 ± 0.37	82.76 ± 0.71	80.34 ± 0.61	79.22 ± 0.38	86.81 ± 0.53	87.12 ± 0.77

Dataset	CIFAR-100					
Setting	Cross-Device (K = 100)			Cross-Silo (K=10)		
	Linear	1%	10%	Linear	1%	10%
f-SimCLR	34.52 ± 0.34	45.47 ± 0.32	51.88 ± 0.56	44.33 ± 0.33	57.61 ± 0.27	67.84 ± 0.15
f-SimSiam	34.96 ± 0.43	47.17 ± 0.73	55.13 ± 0.41	43.16 ± 0.32	53.38 ± 0.66	63.19 ± 0.81
f-SpecLoss	37.60 ± 0.37	47.11 ± 0.56	50.91 ± 0.15	**56.59 ± 0.45**	62.15 ± 0.67	72.09 ± 0.56
f-BYOL	38.47 ± 0.34	52.89 ± 0.62	58.56 ± 0.92	49.64 ± 0.44	57.34 ± 0.45	66.13 ± 0.63
Orchestra	**40.37 ± 0.30**	**54.01 ± 0.41**	**59.07 ± 0.69**	55.89 ± 0.49	**63.73 ± 0.28**	**73.06 ± 0.67**
RotPred (pred)	16.85 ± 0.74	15.79 ± 0.64	19.52 ± 0.73	25.0 ± 0.39	27.64 ± 0.95	28.81 ± 0.82
FedAvg (sup)	58.71 ± 0.42	59.07 ± 0.50	64.01 ± 0.53	65.59 ± 0.38	62.48 ± 0.42	71.59 ± 0.79

rounds to accuracy, we also use kNN accuracy probe (Chen & He, 2021). Finally, during semi-supervised evaluation, we fine-tune the entire model under limited labelled data (1% or 10% labels) where is held-out during the unsupervised training stage.

D. Additional Results

D.1. Linear and Semi-Supervised Evaluation Results (with standard deviations)

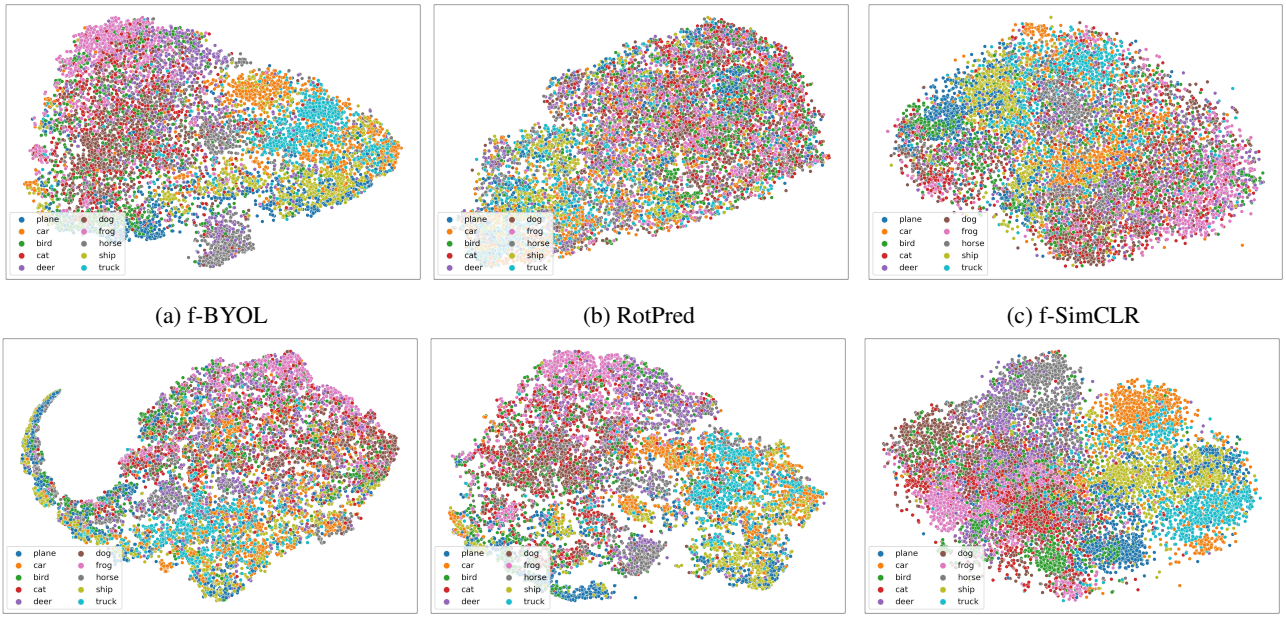
In Table 5, we report the mean and standard deviation of accuracies obtained using linear and semi-supervised Evaluation. Please note that this Table 5 is an extension of Table 1 presented in the main paper, with standard deviation values added to it. Each experiment was run three times with different seeds to obtain the mean and standard deviation scores.

D.2. Ablations

To help avoid degenerate solutions and ensure performant cluster assignments, Orchestra uses two important operations in its local training: degeneracy regularization and an EMA-based target model (see §4). We now provide ablative results for these operations in Table 6. As can be seen, by removing either of the operations, Orchestra can lose substantial performance. As

	Base Setting	No Degeneracy Regularization	No Target Model
CIFAR-10	71.58	56.62	68.63
CIFAR-100	40.37	28.86	36.8

Table 6. Ablation results for Orchestra's target model and degeneracy regularization. These results show both solutions, degeneracy regularization and use of target model are important. However, noisy assignments can be overcome with time and hence sensitivity to use of target model is lower than the use of degeneracy regularization, which helps prevent representational collapse. Experiment settings are the same as Table 1.



(a) f-BYOL (b) RotPred (c) f-SimCLR

(d) f-SimSiam (e) f-SpecLoss (f) **Orchestra**

Figure 12. TSNE visualizations obtained on the CIFAR-10 test set with models trained using different federated unsupervised learning approaches. No labeled data was used to fine-tune the models and the plot shows the quality of representations learned only using unlabeled data on the clients.

expected, noisy assignments can be overcome with time and hence Orchestra is less sensitive to the use of target model. On the other hand, degeneracy regularization is critical for Orchestra, as it helps prevent collapse of the model from the get-go.

D.3. T-SNE Visualizations

In Figure 12, we present the T-SNE visualizations obtained on the CIFAR-10 test set with models trained using different federated unsupervised learning approaches. These plots show the quality of representations learned only using unlabeled data on the clients. We can see that Orchestra provides better class separation than other methods.

D.4. Orchestra and Privacy

Orchestra, by design, does not share any raw data or representations between clients and the server. Only the local centroids computed on each client are shared with the server for the purpose of global clustering. Below we discuss how the design of Orchestra aligns well with the idea of K-anonymous clustering, and how we can further improve Orchestra's privacy guarantees using local differentially private clustering.

K-anonymity: We first focus on the K-anonymous clustering perspective. Formally, a K-anonymity guarantee ensures that for any randomly selected entry in a set, there are at least $K-1$ other entries with the same attributes. While in the discrete setting quasi-identifiers can be used to attack $K-$anonymity guarantees, these mechanisms provably fail in the continuous setting with high dimensional variables (Aggarwal, 2005), making them particularly useful for our settings due to their good utility. K-anonymous clustering methods thus focus on finding clusters with at least K members, providing non-uniform privacy to different samples. Our solution enforces an equal-size constraint on all L local clusters using the sinkhorn-knopp based clustering algorithm (Genevay et al., 2019). This enables uniform, N/L-anonymity across all N samples present on a client. Further, as we showed in Table 2, Orchestra is robust to the number of local clusters and can work with small values of L, which increases the anonymity guarantees of the algorithm.

Local Differential Privacy: Differentially private (DP) algorithms (Dwork & Roth, 2014) seek to design randomized mechanisms or algorithms with stochastic outputs by adding noise to the result of the mechanism. This guarantees that a given result could have been generated from multiple viable datasets. Formally, let $\mathcal{A}$ be a randomized mechanism that takes in a dataset $\mathcal{D}$ as input and whose image is denoted by the set $\mathcal{S}$. Assume $\mathcal{D}_1$ and $\mathcal{D}_2$ are two neighboring datasets, i.e., their

entries differ in only one point. Then, $\mathcal{A}$ is (ε, δ) differentially private if:

$$\Pr\left[\mathcal{A}(\mathcal{D}_1) \in \mathcal{S}\right] \leq \exp(\varepsilon) \cdot \Pr\left[\mathcal{A}(\mathcal{D}_2) \in \mathcal{S}\right] + \delta \tag{6}$$

In the situation a dataset is distributed across multiple participants, DP algorithms assume an honest server will conduct the algorithm by acquiring necessary information from the participants. This can be problematic in the situation where a server is only semi-honest and may seek to leak some sensitive information. To avoid this, *local*-DP guarantees have been developed. In this case, one applies the DP definition for individual participants. For example, if x_1 and x_2 denote two participants, then local DP is defined as:

$$\Pr\left[\mathcal{A}(x_1) \in \mathcal{S}\right] \leq \exp(\varepsilon) \cdot \Pr\left[\mathcal{A}(x_2) \in \mathcal{S}\right] + \delta \tag{7}$$

Recently, local DP has been used for designing private clustering algorithms for distributed settings (Balcan et al., 2017; Stemmer, 2020; Chang et al., 2021). These algorithms follow a standard route generally: (i) find a coreset that is representative of the dataset structure of a participant, but only approximately depends on it; (ii) use this coreset as an approximate notion of clusters from a participant; and (iii) find centroids that partition the coresets. The caveat with this approach is that since DP works on an aggregate scale, if a participant has few samples, the amount of noise that needs to be added to guarantee strong privacy can be huge, as shown by Cohen et al. (2021). Thus, even though one seeks to ensure $\varepsilon < 1.0$ and $\delta \sim \mathcal{O}\left(\frac{1}{\text{number of samples}}\right)$, production systems generally use ε values of order 10.0 (Bureau, 2020).

In Table 7, we provide linear probe results on CIFAR-10 for Orchestra with local-DP based clustering using the current state-of-the-art algorithm (Chang et al., 2021), which uses locality sensitive hashing for finding coresets in a participant's dataset. As can be seen, for practically useful values of ε, Orchestra witnesses only a small performance drop w.r.t. K-anonymity and is still outperforming alternative federated unsupervised methods (see Table 1).

	No Privacy	K-anonymity	ε=0.8	ε=1.0	ε=10.0
Accuracy	71.95	71.58	69.46	69.60	69.63

Table 7. Accuracy obtained using the linear probe evaluation technique (Chen et al., 2020) in five different privacy settings. 'No Privacy' represents the idealized setting when local representations are shared with the server – this setting is prone to representation inversion attacks (Dosovitskiy & Brox, 2016; Nash et al., 2019). The K-anonymity result represents the performance of Orchestra with its 2-level clustering approach, however without any adding any DP protections. We can see that Orchestra's performance with its built-in K-anonymity is almost similar to the 'No Privacy' setting. Further, Orchestra works reasonably well with local-DP based clustering; for practically useful values of ε, Orchestra witnesses only a small performance drop w.r.t. the K-anonymity setting, and still outperforms the various baseline techniques shown in Table 1. For these experiments, we use δ=1e-3, Dirichlet α=0.1, 100 clients, 10 local epochs, and 100 communication rounds.

E. Deferred Proofs

We provide deferred proofs in this section. For better readability and reference, we first tabulate the notations used in the paper in Table 8.

E.1. Orchestra's pipeline reduces δ

In §4, we claimed that by sharing the same set of centroids across all clients, Orchestra's pipeline reduces inter-cluster mixing δ every round. We formalize this result below.

Proposition E.1. *If the same set of global centroids μ are used across all clients, minimizing a loss that brings a sample's assigned centroids and its representation closer will ensure Orchestra's pipeline reduces δ every round.*

Proof. It is easy to see Orchestra's pipeline is an Expectation-Maximization (EM) framework (McLachlan & Krishnan, 2008). Thus, a standard proof schematic for showing convergence of EM can be used. In particular, assume we compute set of G global centroids from the local centroids $\mu = \mathcal{C}(\{\mathcal{C}(R_{X^k}, L^{(k)}) : k \in [K]\}, G)$. Denote the set of samples assigned to global cluster g as π_g. We overload the notation and let $\pi(x)$ denote the assignment of sample x. Then, without loss of generality, we let $(g_m^t, n^t) = \arg\max_{g_m \in [G]} \arg\max_{x_n \in \{X - \pi_g\}} \mu_g^T f(x_n)^t$, where the superscript denotes round t. That is, $\delta^t = \mu_{g_m^t}^T f(x_n)^t$. During local training, if the similarity between x_n's assigned centroid and its representation is increased, we have $\mu_{g_\pi^t(x_n)}^T f(x_n)^{t+1} \geq \mu_{\pi^t(x_n)}^T f(x_n)^t$, and, consequently, $\mu_{g_m^t}^T f(x_n)^{t+1} \leq \mu_{g_m^t}^T f(x_n)^t = \delta^t$ due to the use of softmax

Notation	Definition
$X \sim \mathcal{X}$	Unlabeled samples X drawn from a distribution $\mathcal{X}$
K	Number of clients
M	Number of classes
N	Number of samples
$\mathcal{T} : \mathcal{X} \to \tilde{\mathcal{X}}$	A stochastic augmentation function that transforms its input x to the space of augmented samples by randomly selecting a transform from a set of predefined transformation functions that is finite, but can be large.
$\tilde{X} := \{\mathcal{T}(x) : x \in X\}$	The collective set of augmented samples
$f : \mathcal{X} \to \mathrm{R}^D$	A parametric representation function
$\mathcal{E}(f)$	Error of f under a linear probe computed as $\min_W \mathbb{E}_{x \in \mathcal{X}}[\arg\max(W^T f(x)) = y(x)]$
$R_S = \{f(s) : s \in S\}$	The set of representations on a set S
$\mathcal{C}(\mathcal{B}, G)$	A clustering algorithm that returns G clusters on its input $\mathcal{B}$.
$\mu \in \mathrm{R}^{D \times G}$	Centroids returned by the clustering algorithm $\mathcal{C}(\mathcal{B}, G)$.
μ_g	The centroid of cluster g
$s_f(x)$	Cosine similarity between centroids and representations ${}^{\mu^T f(x)}/_{\|\mu^T f(x)\|}$
$P_f(x)$	Cluster assignment probabilities computed as $\sigma(s_f(x))$ where $\sigma(.)$ denotes the softmax function
$\mathcal{H}(.,.)$	Cross-entropy between two discrete distributions
$\mathcal{L}_{\text{spec}}$	Spectral Contrastive Loss (HaoChen et al., 2021): $= -2\mathbb{E}_{x \in X, \tilde{x} \sim \mathcal{T}(x)}[s_f(x)^T s_f(\tilde{x})] + \mathbb{E}_{x, y \neq x \in X}[(s_f(x)^T s_f(y))^2]$
$\mathcal{F}$	Hypothesis class that has a global minimizer of the L_{spec}

Table 8. Notations used in the paper

for computing assignments. With a similar argument for the server's clustering algorithm, we find the cluster centroids are brought closer to model representations that belong to that cluster. That is, $\mu_{g_m^{t+1}}^T f(x_n)^{t+1} \leq \mu_{g_m^t}^T f(x_n)^{t+1} \leq \delta^t$. Now, if $\delta^{t+1} = \mu_{g_m^{t+1}}^T f(x_n)^{t+1}$, then we have $\delta^{t+1} \leq \delta^t$. If the sample and cluster for computing δ change, then without loss of generality, we assume $\delta^{t+1} = \mu_{g_{m'}^{t+1}}^T f(x_{n'})^{t+1} \leq \mu_{g_{m'}^t}^T f(x_{n'})^{t+1} \leq \mu_{g_{m'}^t}^T f(x_{n'})^t \leq \delta$. Here, the last inequality follows from definition of δ. Second last inequality follows from the fact that local training brings representation of $f(x_{n'})$ closer to its assigned centroid and pushes it away from other centroids including $\mu_{g_{m'}}$. This is the step where we used the fact that global centroids are the same across all clients, due to which the above statement could be made on a dataset level. The first inequality follows from the global clustering step pushing $\mu_{g_{m'}}$ closer to representations of its assigned samples, which do not include $x_{n'}$. □

Note an assumption in the above argument is that all centroids are different from each other, else bringing a sample's representation closer to a centroid will not push it farther from the other centroids. In the case the model representations collapse to a single vector, as often observed in centralized clustering plus representation learning methods (Yang et al., 2017), all centroids become equivalent and hence our assumption is contradicted. Our use of a degeneracy regularization via predictive SSL was specifically motivated to enable this assumption, as it ensures representations do not collapse to a single vector.

The above reasoning also explains why clustering-based SSL solutions (Caron et al., 2020; 2019; 2018; Li et al., 2021a) from centralized settings cannot be directly used in federated settings. Note that SwAV and related methods avoid degenerate solutions via *per-iteration*, partition-based clustering. Consequently, using SwAV in FL would require communicating with the server every iteration. Since communication is very expensive in FL, SwAV becomes an infeasible baseline: for even 10 clients of CIFAR10, *SwAV has 3100x higher communications costs than Orchestra's!* Upon using local training only (no per-iteration operations), we indeed found SwAV reaches degenerate solutions.

E.2. Proofs for Propositions 3.2 and 3.3

Our results are based on the analysis by HaoChen et al. (2021). Therein, the authors derive a general result that shows a minimizer of the loss $-2\mathbb{E}_{x \in X, \tilde{x} \sim \mathcal{T}(x)}[f(x)^T f(\tilde{x})] + \mathbb{E}_{x, y \in X}[(f(x)^T f(y))^2]$ will necessarily have small generalization

error under a linear probe. This result arises out of an analysis of spectral clustering algorithms. As noted in the paper, our focus is partition-based clustering due to its straightforward application to distributed settings. Given the relationship between spectral clustering and partition-based clustering is well established (Dhillon et al., 2004), we can use the result by HaoChen et al. (2021) for our purposes by showing Orchestra implicitly minimizes the objective above, consequently achieving small generalization error if it belongs to a sufficiently complex hypothesis class that can minimize SpecLoss. We first discuss the main concepts underlying those works.

E.2.1. BACKGROUND

The works above include an important assumption called "recoverability", which is itself related to two important concepts called "expansion" and "separation". The notion of expansion and separation was originally proposed for manifolds by Balcan et al. (2004), recently verified empirically by Wei et al. (2021), and converted to "recoverability" or graph connectivity properties by HaoChen et al. (2021). Specifically, the authors assume that there exists a latent graph instantiating the data distribution such that the neighborhood of any low probability set of connected nodes has a higher probability than the set itself. Here a node represents a sample and edges represent joint probability of occurrence of two samples. Under expansion, any given community of nodes on the latent graph is guaranteed to be sufficiently well connected such that if a node with high enough probability is assigned a label, then via a loss promoting consistency to input perturbations, the label will propagate throughout the community. If communities are assumed to be well separated, these labels will have higher diffusion within the same community than between different communities. This induces sets of abstract classes with semantically related nodes (e.g., community of dogs). Using standard pairwise clustering tools, one can compute node representations that enable discrimination between these communities and, if a sufficiently expressive parametric model (e.g., a neural network) is trained to match these representations, it can be guaranteed the model will have small generalization error on unseen data. The approach outlined above was recently used by HaoChen et al. (2021) to design *SpecLoss*, an SSL technique with provable generalization guarantees. However, since pairwise clustering requires computation of representation similarity between different datapoints, it is not amenable to use in decentralized, privacy-sensitive settings. In our work, we circumvented this issue by designing a partition-based, federated clustering framework that first computes centroids capable of partitioning the federation's data into discriminable clusters and then asks clients to allocate their data into these clusters.

Assumption E.2. Recoverability. Let $\tilde{x} \in \mathcal{T}(x)$, where $x \sim \mathcal{X}$, and assume its ground truth label is $y(\tilde{x})$. We assume there exists a classifier U such that $g(x) = y(\tilde{x})$ with probability at least $1 - \phi$.

Let $\mathcal{G}$ denote a graph whose vertices v_x represent samples from distribution $\mathcal{X}$ and whose edges denote joint probability of occurrence of the graph nodes, i.e., $w(x, \tilde{x}) = \mathbb{E}_{\tilde{x}_1, \tilde{x}_2 \in \mathcal{A}(x)} [\Pr(\tilde{x}_1, \tilde{x}_2)]$

Definition E.3. Sparsest m-partition. For an integer $m \in [2, |\mathcal{X}|]$, the sparsest m-partition is defined as $\rho_m := \min_{S_1, S_2, \ldots, S_m} \max\{\kappa_{\mathcal{G}}(S_1), \kappa_{\mathcal{G}}(S_2), \ldots, \kappa_{\mathcal{G}}(S_m)\}$, where $\{S_1, S_2, \ldots, S_m\}$ denotes non-empty sets that form a partition of $\mathcal{X}$ and $\kappa_{\mathcal{G}}(S_i) := \frac{\sum_{x \in S_i, x \notin S_i} w(xx')}{\sum_{x \in S_i} w(x)}$ denotes the Dirichlet Conductance.

Under these definitions, we have the following result.

Theorem E.4. *(Theorem 4.2 from HaoChen et al. (2021)). Let Assumption E.2 hold, $f_{pop} \in \mathcal{F}$ is the population minimizer of L_{spec}, and assume $G \geq 4M + 2$,. Define $\zeta_{\mathcal{X}} = \frac{\phi}{\rho_{G/2}} \cdot \log(G)$. Then for an empirical minimizer function f of SpecLoss such that $\mathcal{L}_{spec}(f) < \mathcal{L}_{spec}(f_{pop}) + \epsilon$, we have:*

$$\mathcal{E}(f) < \zeta_{\mathcal{X}} + \mathcal{O}(\epsilon). \tag{8}$$

Here, $\zeta_{\mathcal{X}}$ is a property of the data distribution $\mathcal{X}$ and $\mathcal{O}$ hides constants related to Rademacher complexity of the function class. Essentially, if the classes of augmentations of a sample can be predicted from the sample, then ϕ and hence $\zeta_{\mathcal{X}}$ are small. This happens if the latent variables instantiating the data generating distribution are similar so that the label of an augmented sample can be predicted from the original sample itself. Further, the denominator $\rho_{G/2}$ depends on the *average* probability of a partition. If we have $G < 2M$, $\rho_{G/2}$ will be zero and hence the error can be arbitrarily large. If $G > 2M$, even though it *can* get larger with G, it will essentially remain constant since one starts inducing subpartitions of abstract classes at this point.

The above theorem is our primary tool. Our idea is to show that instead of the pairwise clustering algorithm (spectral clustering) used by HaoChen et al.(2021), one can use a partition-based clustering algorithm and exploit the result above to

understand if good generalization is feasible in a more practical manner for federated settings. To this end, we will compute the loss achieved by a representation function that yields consistent representations over augmentations and can partition the set of representations into G clusters with small inter-cluster mixing.

E.2.2. PROPOSITION 3.2

We now provide proof for Proposition 3.2, restated below for convenience.

Proposition E.5. *Assume $f \in \mathcal{F}$. Compute $\mathcal{G} > 4M + 2$ clusters $\mu = \mathcal{C}(\{R_{X^k} : k \in [K]\}, G)$ s.t. all clusters are equally sized. Then, if f minimizes $\mathcal{L} := \mathbb{E}_{k\in[K]}\left[\mathbb{E}_{x\in\mathcal{X}_k,\tilde{x}\sim\mathcal{T}(x)}\left[\mathcal{H}\left(P_f(x), P_f(\tilde{x})\right)\right]\right]$, we have*

$$\mathcal{E}(f) < \zeta_{\mathcal{X}} + \mathcal{O}\left(2\delta + (G-2)\delta^2\right). \tag{9}$$

Proof. Assume the hypothesis class is expressive enough to guarantee a zero error population minimizer exists for SpecLoss. Then, we need only bound the error of the empirical minimizer. We thus decompose $L_{\text{spec}} = L^+ + L^-$, where we define the following terms $L^+ := -\frac{2}{N|\mathcal{T}(x)|}\sum_{x\in X,\tilde{x}\in\mathcal{T}(x)}(s_f(x)^T s_f(\tilde{x}))$ and $L^- := \frac{1}{N(N-1)}\sum_{x\in X}\sum_{y\neq x, y\in X}(s_f(x)^T s_f(y))^2$. Let Δ_i denote a vector whose i^{th} term is 1 and rest of the terms are δ, the inter-cluster mixing. Then, we have the following:

$$\begin{aligned}
L^- &= \frac{1}{N(N-1)}\sum_{x\in X}\sum_{y\neq x, y\in X}(s_f(x)^T s_f(y))^2 \\
&\leq \frac{1}{N(N-1)}\sum_{x\in X}\sum_{y\in X}(s_f(x)^T s_f(y)) \\
&= \frac{1}{N(N-1)}\left(N/G\right)^2\sum_{x\in X}\sum_{y\in X}\left(\frac{s_f(x)^T s_f(y)}{\left(N/G\right)^2}\right) \\
&= \frac{N}{G^2(N-1)}\sum_{g_i}\sum_{g_j}\left(\sum_{x\in g_i}\frac{s_f(x)}{(N/G)}\right)^T\left(\sum_{y\in g_j}\frac{s_f(y)}{(N/G)}\right) \\
&= \frac{N}{G^2(N-1)}\sum_{g_i}\sum_{g_j}\left(\mu^T\mu_{g_i}\right)^T\left(\mu^T\mu_{g_j}\right) \\
&\leq \frac{N}{G^2(N-1)}\sum_{g_i}\sum_{g_j}\left(\Delta_i\right)^T\left(\Delta_j\right) \\
&= \frac{N}{G(N-1)}\left[\left(1+(G-1)\delta^2\right) + (G-1)\left(2\delta + (G-2)\delta^2\right)\right] \\
&= \frac{N}{(N-1)}\left[\left(\frac{1}{G}\right) + \left(1-\frac{1}{G}\right)\left(2\delta + (G-1)\delta^2\right)\right] \\
&= \mathcal{O}\left(2\delta + (G-1)\delta^2\right)
\end{aligned} \tag{10}$$

In the above, the first inequality follows from two facts: one, $s(.)$ is always unit norm, and hence the inner products are bound to be less than one, allowing us to ignore squares; two, self-interaction terms are positive, i.e., $s_f(x)^T s_f(x) > 0$. The second inequality follows from the definition of inter-cluster mixing. Note that $P_f(x) = P_f(\tilde{x})$ for $x, \tilde{x} \in \mathcal{T}(x)$ since f is a minimizer of $\mathcal{L}$. Correspondingly, we have $s_f(x) = s_f(\tilde{x})$ since $s(.)$ is scale invariant and hence we have $L^+ = 0$. Adding L^+ and L^- provides us the value of ϵ which can be directly substituted into Equation 8 to complete the proof. □

Note that we hide two constants in the final expression: a constant additive factor $\frac{N}{G(N-1)}$ and a multiplicative factor $\frac{N}{N-1}\left(1-\frac{1}{G}\right)$. The former will be close to 0 and the latter close to 1 for even moderately sized values of N and G.

E.2.3. PROPOSITION 3.3

We now provide proof for Proposition 3.3, restated below for convenience.

Proposition E.6. *Assume $f \in \mathcal{F}$. Denote the set of local centroids as $\mu^L = \{\mathcal{C}(R_{X^k}, L^{(k)}) : k \in [K]\}$ and compute new global centroids $\mu^G = \mathcal{C}(\mu^L, G)$ that are equally sized. Assume at least a fraction c samples are*

"consistently" assigned, i.e., they match their assignments from the idealized setting. Then, if f minimizes the loss $\mathcal{L} := \mathbb{E}_{k\in[K]}\left[\mathbb{E}_{x\in X_k,\tilde{x}\sim\mathcal{T}(x)}\left[\mathcal{H}\left(P_f(x), P_f(\tilde{x})\right)\right]\right]$,

$$\mathcal{E}(f) < \zeta_{\mathcal{X}} + \mathcal{O}\left(\gamma(1-c^2) + (2\delta + (G-1)\delta^2)\right). \tag{11}$$

Proof. The proof follows essentially the same route as subsubsection E.2.2. The part of the argument that changes is the computation of L^-, where we now have to account for the inconsistent assignments $c' = 1 - c$. First, we have the following $((\mu^G)^T f(x))^T((\mu^G)^T f(y)) \leq \Delta_{\text{same}} = 1 + (G-1)\delta^2$ if x and y belong to the same global cluster in the idealized setting. Similarly, $((\mu^G)^T f(x))^T((\mu^G)^T f(y)) \leq \Delta_{\text{diff}} = 2\delta + (G-2)\delta^2$. We also define $n_G = {}^N\!/_G$ as the number of datapoints assigned to a cluster. This implies the number of samples leaving a cluster are at least $c' n_G$ and can consider equality for worst-case analysis. Define the term D_{ij} as samples that were originally in global cluster i in the idealized setting but have now been moved to global cluster j due to the use of local clustering. Note that for any global cluster g, we have $\sum_i D_{gi} \leq c' n_G$, i.e., number of samples coming from different clusters must equal the number of samples that have left the given cluster. Again, for worst-case analysis, we can assume equality. Overall, we get the following.

$$\begin{aligned}
L^- &\leq \frac{1}{N(N-1)}\left(\sum_{i\in[G]}\left(((1-c')n_G)(c'n_G) + G\sum_{j\neq i} D_{ij}\cdot(n_G - D_{ij})\right)\Delta_{\text{same}}\right) \\
&\quad + \frac{1}{N(N-1)}\left(\sum_{i\in[G]}\left(((1-c')n_G)(N - c'n_G - n_G) + \sum_{j\neq i} D_{ij}(N - 2*n_G + D_{ij})\right)\Delta_{\text{diff}}\right) \\
&= \frac{1}{N(N-1)}\left(\left(G(1-c')c'n_G^2 + Gc'n_G^2 - (G-1)\left(\sum_{ij} D_{ij}^2\right)\right)\Delta_{\text{same}}\right) \\
&\quad + \frac{1}{N(N-1)}\left(G(1-c')n_G(N - (1+c')n_G) + Gc'n_G(N - 2n_G) + (G-1)\left(\sum_{ij} D_{ij}^2\right)\Delta_{\text{diff}}\right) \\
&\leq \frac{1}{N(N-1)}\left(Gc'(2-c')n_G^2\Delta_{\text{same}} + (NGn_G(1-c') - Gn_G^2(1-c'^2) + c'Gn_G(N - 2c'n_G))\Delta_{\text{diff}}\right).
\end{aligned} \tag{12}$$

Substituting expressions for variable terms and simplifying, we get,

$$\begin{aligned}
L^- &\leq \frac{N}{(N-1)}\left(\frac{1}{G}\left(1 - \frac{G^2}{N^2}\right) + \frac{1}{G}c'(2-c') + \left(\left(1 - \frac{c'}{G}\right)^2 - \frac{1}{G}\right)(2\delta + (G-1)\delta^2)\right) \\
&= \frac{N}{(N-1)}\left(\frac{1}{G}\left(1 - \frac{G^2}{N^2}\right) + \frac{1}{G}(1-c^2) + \left(\left(1 - \frac{1-c}{G}\right)^2 - \frac{1}{G}\right)(2\delta + (G-1)\delta^2)\right) \\
&= \mathcal{O}\left(\gamma(1-c^2) + (2\delta + (G-1)\delta^2)\right)
\end{aligned} \tag{13}$$

Here, $\gamma = \frac{1}{(G-1-c)^2 - G}$ is a constant that is < 1 for $G > 2$. Again adding and substituting L^+ and L^- in Equation 8 finishes the proof. □