

# Unsupervised Federated Optimization at the Edge: D2D-Enabled Learning without Labels

Satyavrat Wagle\*, Seyyedali Hosseinalipour†,

Naji Khosravan‡, and Christopher G. Brinton\* \*School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA †University at Buffalo-SUNY, Buffalo, NY, USA ‡Zillow Group, Seattle, WA, USA \*{wagles, cgb}@purdue.edu, †alipour@buffalo.edu, ‡najik@zillowgroup.com

**Abstract**—Federated learning (FL) is a popular solution for distributed machine learning (ML). While FL has traditionally been studied for supervised ML tasks, in many applications, it is impractical to assume availability of labeled data across devices. To this end, we develop Cooperative Federated unsupervised Contrastive Learning (CF-CL) to facilitate FL across edge devices with unlabeled datasets. CF-CL employs local device cooperation where either explicit (i.e., raw data) or implicit (i.e., embeddings) information is exchanged through device-to-device (D2D) communications to improve local diversity. Specifically, we introduce a *smart information push-pull* methodology for data/embedding exchange tailored to FL settings with either soft or strict data privacy restrictions. Information sharing is conducted through a probabilistic importance sampling technique at receivers leveraging a carefully crafted reserve dataset provided by transmitters. In the implicit case, embedding exchange is further integrated into the local ML training at the devices via a regularization term incorporated into the contrastive loss, augmented with a dynamic contrastive margin to adjust the volume of latent space explored. Numerical evaluations demonstrate that CF-CL leads to alignment of latent spaces learned across devices, results in faster and more efficient global model training, and is effective in extreme non-i.i.d. data distribution settings across devices.

## I. INTRODUCTION

Many emerging intelligence tasks require training machine learning (ML) models on a distributed dataset across a collection of wireless edge devices (e.g., smartphones, smart cars) [2]. Federated learning (FL) [3] utilizes the computational resources available at edge devices for data processing. Under conventional FL, model training consists of (i) a sequence of local iterations by devices on their individual datasets, and (ii) periodic global aggregations by a main server to generate a global model that is synchronized across devices to begin the next training round.

In this work, we are motivated by two fundamental challenges related to implementation of FL over real-world edge networks. First, device datasets are often non-independent and identically distributed (non-i.i.d.), causing local models to be biased and a significant degradation in global model performance. Second, data samples collected by each device (e.g., images, sensor measurements) are often unlabeled, which makes supervised ML model training impossible. We aim to jointly address these challenges with a novel methodology

for information sampling and exchange across devices while remaining sensitive to any data privacy restrictions.

### A. FL with Labeled vs. Unlabeled Data

Most works in FL assume labeled data across the devices [4]. Large amounts of unlabeled data can be useful to learn representations of a large scale distributed dataset with the objective of adapting to many downstream tasks, such as classification, segmentation, etc., through zero or few shot learning [5]. In the following, we provide a brief literature review to summarize the extensive line of work in supervised FL, and then detail the current state of the art in unsupervised FL.

1) *FL under labeled data*: Researchers have aimed to address the impact of non-i.i.d. device data distributions on FL performance. In [6], convergence analysis of FL via device gradient diversity-based metrics is conducted, and control algorithms for adapting system parameters (e.g., aggregation frequencies) to optimize convergence speeds are proposed. In [7], a reinforcement learning-based method for device selection is introduced, aiming to counteract local model biases caused by non-i.i.d. data. In [8], a clustering-based approach is developed, aiming to produce a hierarchy of local models that captures their diversity. In [9], the authors tune the global model aggregation procedure to reduce local model divergence using a theoretical upper bound.

2) *FL under unlabeled data*: A few works have recently considered unsupervised FL. In [10], a local pretraining methodology was introduced to generate unsupervised device model representations for downstream machine learning tasks. The authors in [11] proposed addressing the inconsistency of local representations arising from non-i.i.d. unlabelled datasets through a dictionary-based method implemented at the server. In [12], the dataset imbalance problem is addressed with weight aggregation at the server being defined according to inferred sample densities. In [13], unsupervised FL is considered for the case where unlabeled device data has been subdivided into sets that can be treated as surrogate labels for training. The authors in [14] exploit similarities across locally trained unsupervised model representations to correct against biases by maximizing local agreement with the global models.

In this work, we consider *contrastive learning* (CL) as our framework for unsupervised ML, and extend it to the federated setting. CL is an ML technique which aims to learn embeddings of unlabeled datapoints that maximize the distance between

This paper is an extension of the conference paper [1] which appeared in the 2022 IEEE Global Communications Conference.

different data points and minimizes it for similar points in the latent space [15]. Our work adapts contrastive learning to a federated setting, while avoiding the additional work of assigning surrogate labels to datapoints such as in [13]. Our work enables contrastive learning to be performed on exclusively unlabelled data, as opposed to a combination of supervised and contrastive learning, such as in [14]. Along these lines, in [16], the authors propose a self-supervised learning framework for federated systems using contrastive loss, and propose a divergence-aware method for local model updates which adjusts local models towards global knowledge. In contrast, our work employs the contrastive learning framework to identify and share crucial information between devices, to promote alignment between local models during federated training.

### B. FL with Information Exchange Among Devices

One of the major challenges faced in the implementation of FL over real-world networks is data heterogeneity across devices, which as discussed in Sec. I-A, leads to local model bias. A line of solutions to this challenge for settings that are not completely privacy restricted is to enable some form of *information* exchange among the devices.

When information sharing is permitted in FL, there are many scenarios in which inter-device information exchange may be preferred to sharing the information with a server and training the model in a centralized manner. This includes settings where the server is far away from the devices (e.g., in large wireless cells [17], where there is a large hierarchy of nodes separating device and server (e.g., in hierarchical fog networks) [18], and/or where the server's computational capabilities beyond facilitating model aggregations are limited [19], [20]. In such settings, centralization may be prone to large energy consumption and/or time delays compared with periodically sharing small amounts of information over local device networks [21].

Information can be classified into two categories: (i) *explicit* or *raw information*, which contains the full set of features describing the datapoints (e.g., RGB colors of the pixels in a photo), and (ii) *implicit* or *encoded* information, which correspond to processed datapoint features represented in a lower dimensional space upon applying some form of filtration (e.g., embeddings or projections).

In a networked system, explicit information exchange exposes raw user information, which makes it suitable only for applications that are not subject to stringent restrictions of data privacy, such as computation offloading in (i) wireless sensor networks [22] (ii) self driving vehicles [23], and (iii) resource constrained IoT devices [24]. One of the main advantages of explicit information exchange in a FL setting is that it does not require any pre-processing of features of data, and thus makes it suitable for resource constrained edge devices to incorporate into training (e.g., sensors, and unmanned aerial vehicles). On the other hand, in applications with strict data privacy concerns (e.g. facial recognition [25], user activity recognition [26], and some text mining applications [27]), exchange of only implicit information is permitted, since it is retrieving raw data features

from the encoded information is extremely difficult [28]. That being said, obtaining implicit/encoded information requires pre-processing of data at the end users, which in turn can impose high latency and computation burden.

Information exchange among FL devices has been studied by a few recent works, most of which are concerned with supervised FL [11], [29]–[31]. In this work, we aim to design smart data (i.e., explicit information) and embedding (i.e., implicit information) exchange for cooperative FL systems via exploiting device-to-device (D2D) communications. Our method can be applied for scenarios which allow for inter-device communication, and is compatible with different levels of privacy restrictions. We employ explicit information exchange for scenarios which are not subject to stringent privacy restrictions, such as wireless sensor networks aiming to learn about an environment across different modalities of data [32], vehicle to vehicle (V2V) networks which benefit from exchanging traffic information among vehicles to optimally plan routes [33], and swarm robotics where agents aim to cooperate to achieve a common goal [34]. On the other hand, for scenarios where data privacy is a primary concern, such as distributed healthcare systems [35], we employ implicit information exchange, with embeddings that cannot be used to reconstruct source datapoints. We also aim for our method to facilitate information exchange in a fully distributed manner, i.e., without any added control overhead at the FL server.

In the following, we provide a review of works concerned with information exchange in FL, and highlight the current art in unsupervised FL.

1) *FL with explicit information exchange*: Recent in supervised FL have shown that, when permissible, even a small amount of raw data exchange among devices can substantially improve training [11], [29]–[31]. However, to the best of our knowledge, the impact of raw data sharing in unsupervised FL has not been studied. In contrastive learning, the model at each device can use exchanged information as negatives for anchors selected from local data, and as anchors where negatives are selected from local data to improve the overall training performance (see Sec. I-B for the definition of negative and anchor datapoints). To exploit D2D communications as a substrate for improving local model alignment, we design a probabilistic smart data push-pull strategy for unsupervised FL that enables devices to exchange raw datapoints that have the highest expected impact on the performance of the global model.

2) *FL with implicit information exchange*: Implicit information exchange has been recently studied by a few works for unsupervised and semi-supervised FL. The authors in [11] have attempted to address the alignment of representations in unsupervised FL by incorporating implicit information exchange. However, this method requires the availability of a *public dataset* at the server, datapoints of which are sampled from the devices. Availability of public data is a common practice in personalized FL, where a public dataset is used for knowledge distillation [36], [37] and transfer learning [38]. However, reliance on such a dataset renders the method of [11] not directly applicable to our implicit information exchange setting where raw data sharing is not allowed. Alternatively,

methods such as [39] attempt to align the local models of the devices with a classification model learned at the server. Such methods rely on availability of labelled data, as well as a trainable model at the server.

As compared to the above works, we study a different problem, which aims to conduct smart implicit information exchange among the devices via D2D communications for unsupervised FL. We aim to facilitate the information exchange process in a distributed manner, without introducing any control overhead for a main server or alternative aggregation point.

### C. Implicit and Explicit Information Exchange

We characterize explicit information exchange as an exchange of raw datapoints, and implicit information exchange as an exchange of embeddings of datapoints generated by local models. Properties unique to each of these exchange modalities lend themselves to certain system characteristics. Explicit information is ground truth, which is immutable, and hence does not suffer from “staleness.” Implicit information on the other hand, is transient, as it is a function of local model  $\phi_i^t$  of device  $i$  at time-step  $t$ . The fidelity of information received at time  $t$  reduces with time, thus necessitating more frequent information exchange. However, implicit information encodes the data by the local model  $\phi_i^t$ , making it less of an exposure as compared to explicit information. Implicit information exchange also provides users with flexibility in choosing their communication payload, as the dimensionality of the representation is a user defined parameter. The size of explicit information, on the other hand, is predetermined. In case of implicit information exchange, adjusting this dimension allows us to improve communication and computation requirements for resource-limited devices.

Thus, explicit information exchange is favored for systems working with low risk data, and where frequent exchange of information is not feasible, e.g., in the wireless sensor networks example mentioned above. On the other hand, implicit information exchange is favored for systems that deal with data in situations where privacy is one of the primary concerns, but where frequent communication with other devices in the network is possible from a time and resource consumption perspective, e.g., in the distributed healthcare system example.

### D. Importance Sampling vs. Information Exchange

The connection between our work and the notion of importance sampling in ML [40], [41] is worth mentioning. In the ML community, importance sampling techniques have been introduced to accelerate training through the choice of minibatch data samples [42]. In FL, by contrast, importance sampling has typically been employed to identify *devices* whose models provide the largest improvement to the global model, e.g., [29], [43]. Our work extends the literature of importance sampling to consider inter-device datapoint/embedding transfers in a federated setting, where devices exchange their local datapoints and embeddings to accelerate model training speed.

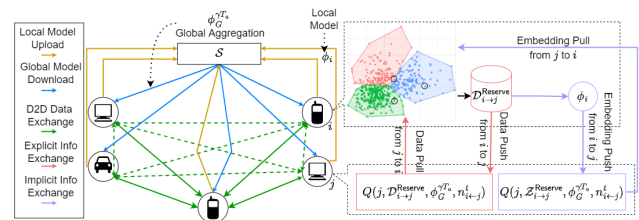


Fig. 1: CF-CL introduces smart push-pull information transfer to improve unsupervised FL based on importance information sampling.

### E. Summary of Contributions

- We develop CF-CL – Cooperative Federated unsupervised Contrastive Learning – a novel method for unsupervised FL. CF-CL improves training speed via smart D2D information exchange among the devices in a cooperative framework. CF-CL is a general plug-and-play technique that can be mounted on current unsupervised FL methods.
- We introduce a novel D2D-based information push-pull strategy based on probabilistic importance sampling in CF-CL. For explicit information exchange, we characterize the importance of a remote datapoint via a joint clustering and loss measurement technique, aiming to accelerate the convergence rate of FL in the presence of limited communication resources among the devices. We also develop a method for intelligent reserve data selection which identifies a set of datapoints that promote a balanced representation of the overall distribution to improve importance calculations.
- For implicit information exchange, we develop a probabilistic embedding sampling technique for CF-CL. To incorporate the exchanged embeddings among the devices in the training process, we introduce a regularization term into the conventional definition of contrastive loss which promotes model debiasing. We also introduce a dynamic contrastive margin for exchanged implicit information to adjust the volume of latent space explored for identifying hard negatives based on cluster size.
- Our numerical experiments demonstrate that CF-FL significantly improves FL training in terms of (i) alignment of unsupervised learned latent spaces, (ii) global ML model convergence, and (iii) communication resources and delay incurred to reach target accuracy levels. We find this for non-i.i.d. data distributions across devices under both explicit and implicit information sharing mechanisms.

An abridged conference version of this work appeared in [1]. Compared to [1], we make the following key additions: (i) We extend CF-CL to be applicable to embeddings (implicit information) in addition to datapoints (explicit information) as presented in the conference version. (ii) We adapt the *smart push-pull* of information to identify important embeddings using a distance based probabilistic selection mechanism. (iii) We address the inherent drawbacks of implicit information, namely staleness and consequently time-varying importance values by introducing a time-variant scaling operation which aims to reduce the influence of obsolete implicit information. (iv) We significantly expand upon our numerical section (Sec. IV) by conducting experiments on an additional dataset as well as analyzing the alignment of the embeddings produced by

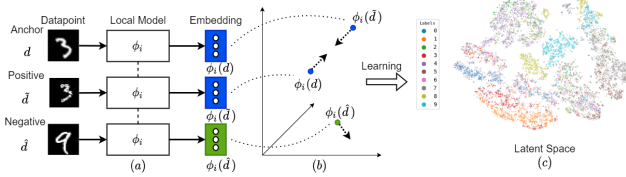


Fig. 2: (a) A datapoint (*anchor*), its augmentation (*positive*) and a distinct datapoint (*negative*), are passed through models to obtain embeddings (b). Training maximizes distance between anchor and negative, while minimizing distance between anchor and positive (c).

CF-CL, the resource costs of CF-CL, the impact of device selection per aggregation and the effect of choosing local models for importance calculations.

## II. SYSTEM MODEL AND MACHINE LEARNING TASK

An overview of our method is illustrated in Fig. 1. In this section, we go over our network model (Sec. II-A) followed by the ML task for unsupervised FL (Sec. II-B).

### A. Network Model of FL with Information Exchange

We consider a network consisting of a server  $S$  and a set of devices given by the set  $\mathcal{C}$ . At each time-step  $t$ , each device  $i \in \mathcal{C}$  possesses a local ML model parametrized by  $\phi_i^t \in \mathbb{R}^p$ , where  $p$  is the number of model parameters. Let  $\mathcal{D}_i$  denote the *initial* local dataset at device  $i$ , i.e., before any information exchange. Each local model  $\phi_i^t$  is trained on  $\mathcal{D}_i$  as well as information received from neighboring devices. The server aims to maintain a global model  $\phi_G^t$  via aggregating  $\{\phi_i^t\}_{i \in \mathcal{C}}$ .

We represent the D2D communication graph between the devices via  $\mathcal{G} = (\mathcal{C}, \mathcal{E})$  with vertex set  $\mathcal{C}$  and edge set  $\mathcal{E}$ . The existence of an edge between two nodes  $i$  and  $j$  (i.e.,  $(i, j) \in \mathcal{E}$ ) implies a communication link between the corresponding devices. This graph will be determined by the specific D2D protocol in place among the devices, which may factor in transmit powers, distances, and channel conditions (e.g., see Sec. V of [44]), which we assume is specified at the wireless layer. Without loss of generality, we assume  $\mathcal{G}$  is undirected, i.e.  $(i, j) \in \mathcal{E}$  implies  $(j, i) \in \mathcal{E}$ ,  $\forall i, j$ , and is static over all time  $t$ . We further denote the neighbors of device  $i$  with  $\mathcal{N}_i = \{j : (i, j) \in \mathcal{E}\}$ .

We consider *cooperation* among the devices [29], [31], where devices push and pull information among themselves over the graph  $\mathcal{G}$ . In case of explicit information sharing, the pushed/pulled information is a subset of the raw datapoints at the transmitter, while for implicit information exchange, it is a set of embeddings generated at the transmitter.

### B. Unsupervised FL Formulation

We consider an unsupervised learning task whose output is a set of *embeddings* of datapoints (i.e., projections of datapoints onto a latent space). We adopt a contrastive learning (CL) framework given its popularity in centralized ML [15]. CL aims to learn embeddings by minimizing the *distance* between similar datapoints while maximizing it between dissimilar datapoints. Given an anchor datapoint  $d$ ; a similar datapoint (i.e., a positive) is obtained by applying a randomly sampled

augmentation function (e.g., image transformations, Gaussian blurs and/or noise) to it [15]. Any datapoint distinct from the anchor is considered a dissimilar datapoint (i.e., a negative).

CL aims to learn embeddings using the *triplet loss* function [42]. Formally, given an embedding model  $\phi$ , margin  $m$ , anchor  $d$ , augmented anchor (positive)  $\tilde{d} = F(d)$  and distinct datapoint (negative)  $\hat{d}$ , triplet loss  $L$  is defined for triplet  $\{d, \tilde{d}, \hat{d}\}$  as

$$L_\phi(d, \tilde{d}, \hat{d}) = \max\left\{0, \|\phi(d) - \phi(\tilde{d})\|_2^2 - \|\phi(d) - \phi(\hat{d})\|_2^2 + m\right\}, \quad (1)$$

where  $F$  is a random augmentation function selected from a set of predefined augmentation functions  $\mathcal{F}$  (i.e.,  $F \in \mathcal{F}$ ). Minimizing triplet loss promotes a latent space in which similar datapoints are closer to one another while dissimilar ones are further away in latent space by at least a margin of  $m$  as illustrated in Fig. 2.

In our distributed ML setting, we define the goal of unsupervised FL as identifying a global model  $\phi_G^*$  such that

$$\phi_G^* = \min_{\phi \in \mathbb{R}^p} \sum_{d \in \mathcal{D}} \sum_{F \in \mathcal{F}} \sum_{\tilde{d} \in \mathcal{D}, \tilde{d} \neq d} L_\phi(d, F(d), \tilde{d}), \quad (2)$$

where  $\mathcal{D} = \bigcup_{i \in \mathcal{C}} \mathcal{D}_i$  represents the global dataset. The optimal *global latent space* (e.g., subplot (c) in Fig. 2) is the one in which anchors and their positive samples are closer to each other while being further away from negative samples across the *global dataset* in the latent space.

In the federated setting, the local datasets  $\mathcal{D}_i$  will not be independent and identically distributed (non-i.i.d.). In order to speed up convergence in the presence of non-i.i.d., *alignment* between *local* latent spaces during training is necessary. We propose to accelerate this alignment across devices by smart data (i.e., explicit information) or embedding (i.e., implicit information) transfers. Intuitively, when the data across the devices is homogeneous (i.e., i.i.d.), the latent spaces learned locally will be aligned, under which the global model training will exhibit a fast convergence. Thus, given non-i.i.d. data distributions, we aim to select and share a set of *important* datapoints or embeddings across the devices, that result in the closest alignment of their local models.

## III. COOPERATIVE FEDERATED UNSUPERVISED CONTRASTIVE LEARNING (CF-CL)

In this section, we develop our unsupervised FL methodology CF-CL. We first give an overview of the training and information exchange processes of CF-CL in Sec. III-A. We then detail our cooperative data transfer mechanism, i.e., explicit data exchange in Sec. III-B. Finally, we detail our embedding exchange methodology, i.e., implicit data exchange in Sec. III-C.

### A. Training and Information Exchange Overview

In CF-CL, (2) is solved through a sequence of global model aggregations indexed by  $\gamma \in \mathbb{Z}^+$ . Let  $T_a \in \mathbb{Z}^+$  denote the aggregation interval length in time-steps, so that the aggregations occur at times  $t \in \{\gamma T_a\}_{\gamma \in \mathbb{Z}^+}$ . At each time-step  $t = 1, \dots, T$ , each device  $i \in \mathcal{C}$  conducts one mini-batch stochastic gradient descent (SGD) iteration.

Operationally, the information exchange process in CF-CL is a combination of

- 1) A *Push* of information from each transmitting device  $i \in \mathcal{C}$  to its neighbors  $j \in \mathcal{N}_i$ . The pushed set is denoted by  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}$  in the case of explicit information exchange and  $\mathcal{Z}_{i \rightarrow j}^{\text{Reserve}}$  in the case of implicit information exchange. The pushed information is used by device  $j$  for determining information importance across its own neighbors.
- 2) This is followed by a periodic *Pull* of information. Indexed by  $\tau \in \mathbb{Z}^+$ , pulls occur at time-steps  $t \in \{\tau T_p\}_{\tau \in \mathbb{Z}^+}$ , where  $T_p$  is the information pull period. At each  $t = \tau T_p$ , each receiving device  $i$  requests pulling  $n_{j \rightarrow i}^t$  information units (i.e., datapoints or embeddings) from device  $j \in \mathcal{N}_i$ . The resulting pulled information is denoted by  $\mathcal{D}_{j \rightarrow i}^t$  in case of explicit information exchange and  $\mathcal{Z}_{j \rightarrow i}^t$  in case of implicit information exchange.

Periodic push-pull of information enables CF-CL to select information that is most important for local model training based on the most recent model parameters. As we observe in (1), there is zero loss and thus no change in the model parameters when the difference between the anchor-negative and anchor-positive embedding distances is more than the margin  $m$ , where the anchor, positive and negative embeddings generated by a model are a function of the current model parameters  $\phi$ . As a consequence, importance calculations based on stale model parameters can lead to the exchange of information that does not produce any meaningful change in the local models, and thus is not an efficient use of communication resources.

In practice, the number of information units pulled  $\{n_{j \rightarrow i}^t\}_{i \in \mathcal{C}, j \in \mathcal{N}_i}$  are subject to a limited exchange budget, e.g. based on wireless resource availability. Similar to the D2D graph  $\mathcal{G}$ , we assume the values of  $n_{j \rightarrow i}^t$  are fixed by the wireless layer, and focus on smart information selection built on top of this. Additionally, we assume that each device has a buffer of limited size to store pulled information units. Hence, before pulling information at time  $t = \tau T_p$ , it will purge any information pulled in the previous iterations  $\tau'$ , where  $\tau' T_p < \tau T_p$ .

The pseudocode of the overall CF-CL procedure is given in Alg. 1. The push-pull algorithms will be detailed in the following subsections. In doing so, one of our main contributions will be developing the information sampling functions referenced in lines 12 and 15 of Alg. 1. The function for explicit information exchange will sample the most important datapoints for local model training (Alg. 2), while for implicit information exchange it will promote a transfer of embeddings leading to the most expected impact on local training (Alg. 3).

### B. Smart Push-Pull of Explicit Information

Here we develop our methodology for settings where D2D data sharing is permissible, i.e., in ML applications where privacy is not a major concern.

<sup>1</sup>Our method also applies to settings where devices have unlimited buffer sizes and accumulate the pulled information units.

### Algorithm 1: CF-CL Procedure at each Device $i \in \mathcal{C}$

```

1 Input:  $i, T_a, T_p, \alpha, \mathcal{D}_i, \{n_{j \rightarrow i}^t\}_{j \in \mathcal{N}_i}$ 
2 Device  $i$  receives the initial global model from the server  $\phi_G^0$ 
3 Device  $i$  performs K-means clustering on  $\mathcal{D}_i$  with
    $K = K_{i \rightarrow j}^{\text{Reserve}}$ 
4 Device  $i$  samples reserve data  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}$  by choosing  $K_{i \rightarrow j}^{\text{Reserve}}$ 
   datapoints closest to centroids.
5 if Explicit Information Sharing then
6   Device  $i$  pushes reserve data  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}$  given by (6) to each
   of its neighboring devices  $j, j \in \mathcal{N}_i$ 
7 for  $t = 1$  to  $T$  do
8   if  $t = \tau T_p, \tau \in \mathbb{Z}^+$  then
9     for  $j \in \mathcal{N}_i$  do
10      Device  $i$  requests pulling  $n_{j \rightarrow i}^t$  units of
11      information from device  $j$  if Explicit
12      Information Sharing then
13        Device  $j$  transmits datapoints to device  $i$ 
14        using Alg. 2
15      if Implicit Information Sharing then
16        Device  $i$  pushes the embeddings of reserved
17        datapoint  $\mathcal{Z}_{i \rightarrow j}^{\text{Reserve}}$  given by (13) to device  $j$ 
18        Device  $j$  transmits embeddings to device  $i$ 
19        using Alg. 3
20    if Explicit Information Sharing then
21      Device  $i$  updates its local model  $\phi_i^t$  according to (4)
22      using the triplet loss definition in (1)
23    if Implicit Information Sharing then
24      Device  $i$  updates its local model  $\phi_i^t$  according to (4)
25      using the regularized triplet loss definition in (23)
26    if  $t = \gamma T_a, \gamma \in \mathbb{Z}^+$  then
27      Device  $i$  sends its local model to the server, which
28      updates global model  $\phi_G^t$  according to (5)

```

1) *Model Training:* Consider the time-steps  $t$  between data pulls  $\tau$  and  $\tau + 1$ , i.e.,  $\tau T_p \leq t < (\tau + 1)T_p$ . The data stored at each device will be  $\mathcal{D}_i^t = \mathcal{D}_i^{\tau T_p} \triangleq \mathcal{D}_i \cup \tilde{\mathcal{D}}_i^{\tau T_p}$ , i.e., its initial datapoints  $\mathcal{D}_i$  and those pulled from neighboring devices  $\tilde{\mathcal{D}}_i^{\tau T_p} = \cup_{j \in \mathcal{N}_i} \mathcal{D}_{j \rightarrow i}^{\tau T_p}$ . Over the course of the period, device  $i$  aims to minimize its triplet loss via solving a local version of (2) as follows:

$$\phi_i^* = \min_{\phi} \sum_{d \in \mathcal{D}_i^t} \sum_{F \in \mathcal{F}} \sum_{\hat{d} \in \mathcal{D}_i^t, \hat{d} \neq d} L_{\phi}(d, F(d), \hat{d}). \quad (3)$$

To solve (3), each device undergoes local model updates via SGD. At time  $t$ , given local model  $\phi_i^t$  and a mini-batch of triplets  $\mathcal{B}_i^t = \{(d, \tilde{d}, \hat{d}) : d \in \mathcal{D}_i^t, \tilde{d} = F(d), F \in \mathcal{F}, \hat{d} \in \mathcal{D}_i^t\}$  sampled from its local dataset  $\mathcal{D}_i^t$ , device  $i$  updates its local model as

$$\phi_i^{t+1} = \phi_i^t - \alpha_t \sum_{(d, \tilde{d}, \hat{d}) \in \mathcal{B}_i^t} \nabla_{\phi_i^t} L_{\phi_i^t}(d, \tilde{d}, \hat{d}) / |\mathcal{B}_i^t|, \quad (4)$$

where  $\alpha_t$  is the learning rate at time  $t$ .

To solve (2), the server aggregates the local models  $\{\phi_i^{t+1}\}_{i \in \mathcal{C}}$ . Aggregation  $\gamma$  occurs at  $t = \gamma T_a$ , and generates a global model  $\phi_G^t$ . We consider an aggregation scheme proportional to the average cardinality of local datapoints at each device since the last aggregation round  $(\gamma - 1)$ .

---

**Algorithm 2:** Explicit Information Pull by Device  $i$  from Device  $j \in \mathcal{N}_i$

---

- 1: **function**  $Q(j, \mathcal{D}_{i \rightarrow j}^{\text{Reserve}}, \phi_G^{\gamma T_a}, n_{j \rightarrow i}^t)$
  - 2: Transmitter  $j$  approximates its local dataset as  $\mathcal{D}_j^{t, \text{Approx}}$  according to (7)
  - 3: Transmitter  $j$  performs Kmeans++ on  $\mathcal{D}_j^{\text{Reserve}}$  and  $\mathcal{D}_j^{t, \text{Approx}}$  ▷ *Macro Importance*
  - 4: Transmitter  $j$  obtains sampling probability  $[P_{j \rightarrow i}^{t, \text{Macro}}(\ell)]_{\ell \in \mathcal{L}_{j \rightarrow i}^t}$  via (8)
  - 5: Transmitter  $j$  calculates importance of each datapoint  $\hat{d}$  in cluster  $\ell \in \mathcal{L}_{j \rightarrow i}^t$  by (11) ▷ *Micro Importance*
  - 6: Transmitter  $j$  samples  $n_{j \rightarrow i}^t$  datapoints to obtain set  $\mathcal{D}_{j \rightarrow i}^t$  according to  $P_{j \rightarrow i}^t$  from (12) ▷ *Data sampling*
  - 7: Device  $j$  transmits  $\mathcal{D}_{j \rightarrow i}^t$  to device  $i$  ▷ *Data Transfer*
  - 8: **end function**
- 

Formally, letting  $D_i^{(\gamma-1)\gamma} = \sum_{t \in \{(\gamma-1)T_a+1 \dots \gamma T_a\}} D_i^t / T_a, \forall i$ , the aggregation is:

$$\phi_G^t = \frac{1}{\sum_{i \in \mathcal{C}} D_i^{(\gamma-1)\gamma}} \sum_{i \in \mathcal{C}} \phi_i^t D_i^{(\gamma-1)\gamma}, \quad t = \gamma T_a, \gamma \in \mathbb{Z}^+. \quad (5)$$

Global model  $\phi_G^t$  is then broadcast across all devices  $i \in \mathcal{C}$ , synchronizing all local models, i.e.,  $\phi_i^t \leftarrow \phi_G^t$ , when  $t = \gamma T_a$ . This model is then used for the next local training rounds as in (4).

2) *Smart Push of Explicit Information:* Explicit information exchange aims to share datapoints that best contribute to cross-device embedding alignment. First, each device  $i$  pushes a set of *representative* datapoints (called reserve data) to each of its neighbors  $j$  as  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}$  at  $j$ , i.e.,

$$\mathcal{D}_{i \rightarrow j}^{\text{Reserve}} = \{d : d \sim \mathcal{D}_i\}, |\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}| = K_{i \rightarrow j}^{\text{Reserve}}, \quad j \in \mathcal{N}_i, \quad (6)$$

where  $K_{i \rightarrow j}^{\text{Reserve}}$  is the number of reserve datapoints at  $j$  from  $i$ . To select explicit information to exchange, we use K-means++ [45] clustering, with  $K = K_{i \rightarrow j}^{\text{Reserve}}$  at each device  $i$  and include the centers of clusters/centroids in  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}$ ,  $j \in \mathcal{N}_i$ . Typically,  $K_{i \rightarrow j}^{\text{Reserve}}$  is constant across neighbours  $j$ , so K-means++ is executed only once. This increases performance significantly compared to random data sampling, especially if  $K_{i \rightarrow j}^{\text{Reserve}}$  is small, as we will see in Sec. IV.

3) *Smart Pull of Explicit Information:* We now describe the strategy for explicit information selection, which we denote by  $Q$ . The summary of this process is given in in Alg. 2.  $Q$  identifies locally important datapoints to be pulled by each device  $i \in \mathcal{C}$  from device  $j \in \mathcal{N}_i$ . At each global aggregation time-step  $t = \gamma T_a$ , device  $j$  approximates its local dataset by uniformly sampling a fixed number  $K_j^{\text{Approx}}$  of its local datapoints, i.e.,

$$\mathcal{D}_j^{t, \text{Approx}} = \{d : d \sim \mathcal{D}_j^t\}, |\mathcal{D}_j^{t, \text{Approx}}| = K_j^{\text{Approx}}, \quad t = \gamma T_a. \quad (7)$$

$\mathcal{D}_j^{t, \text{Approx}}$  constitutes the set of candidate datapoints at device  $j$  for transmission to neighboring devices. Uniformly sampling a subset of  $\mathcal{D}^t$  improves the efficiency of CF-CL for large local datasets while providing an unbiased estimate of the local data distribution at device  $j$ .

At each information pull instance  $\tau$ , which occurs between two global aggregation rounds  $\gamma$  and  $\gamma + 1$  (i.e.,  $\gamma T_a \leq \tau T_p <$

$(\gamma + 1)T_a$ ), the explicit information pull by device  $i$  from device  $j$  is denoted by  $\mathcal{D}_{j \rightarrow i}^{\tau T_p} = Q(j, \mathcal{D}_{i \rightarrow j}^{\text{Reserve}}, \phi_G^{\gamma T_a}, n_{j \rightarrow i}^t) \subseteq \mathcal{D}_j^{t, \text{Approx}}$ . We aim to design the selection strategy  $Q$  to promote faster convergence of global models  $\phi_G^t$  to the optimal model  $\phi_G^*$  by sampling and pulling information that is *important* in that, it accelerates local model convergence while avoiding local model bias. The global model  $\phi_G^{\gamma T_a}$  is used in  $Q$  to determine the most effective datapoints from device  $j$  to minimize device  $i$ 's bias to its local dataset.

Formally, to perform the data pull between each pair of devices  $(i, j)$ , we implement a *two-stage probabilistic importance sampling* procedure, consisting of macro and micro sampling steps.

(1) *Macro Sampling:* In *macro sampling*, the data at device  $j$  that is used for estimates pertaining to device  $i$ , which is  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}} \cup \mathcal{D}_j^{t, \text{Approx}}$ , is partitioned into clusters, which are assigned a cluster-level sampling probability. Specifically, device  $j$  obtains the embeddings of all datapoints in  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}$  and  $\mathcal{D}_j^{t, \text{Approx}}$  by feeding them through global model  $\phi_G^{\gamma T_a}$ , and performs K-means++ to partition them into clusters of embeddings, which we denote as a set of clusters  $\mathcal{L}_{j \rightarrow i}^t$ . Device  $j$  then assigns a sampling probability  $P_{j \rightarrow i}^{t, \text{Macro}}(\ell)$  to each of the clusters  $\ell \in \mathcal{L}_{j \rightarrow i}^t$  proportional to the importance of the cluster relative to device  $i$ ; we refer to this cluster level importance as the *macro* sampling probability. Formally, these probabilities are obtained as

$$P_{j \rightarrow i}^{t, \text{Macro}}(\ell) = \frac{X_{j \rightarrow i}^{t, \text{Macro}}(\ell)}{\sum_{\ell \in \mathcal{L}_{j \rightarrow i}^t} X_{j \rightarrow i}^{t, \text{Macro}}(\ell)}, \quad t = \tau T_p, \quad (8)$$

where

$$X_{j \rightarrow i}^{t, \text{Macro}}(\ell) \triangleq \frac{K_{j \rightarrow i}^{t, \text{Approx}}(\ell)}{K_j^{t, \text{Approx}}(\ell) + K_{i \rightarrow j}^{t, \text{Reserve}}(\ell)}. \quad (9)$$

In (9),  $K_{j \rightarrow i}^{t, \text{Approx}}(\ell)$  is the number of datapoints in  $\mathcal{D}_j^{t, \text{Approx}}$  located in cluster  $\ell$ , and  $K_{i \rightarrow j}^{t, \text{Reserve}}(\ell)$  is the number of datapoints in  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}$  located in cluster  $\ell$ .<sup>2</sup> Intuitively, using the sampling probability  $P_{j \rightarrow i}^{t, \text{Macro}}(\ell)$  defined in (8) promotes selection of clusters containing a higher fraction of the datapoints at device  $j$ , which are representative of  $j$ 's distribution (through  $K_j^{t, \text{Approx}}(\ell)$ ) but dissimilar from device  $i$  (through  $K_{i \rightarrow j}^{t, \text{Reserve}}(\ell)$ ), and thus selecting clusters which are more likely to improve device  $i$ 's local distribution. Such diversification of the local distribution at each device  $i$  results in the local dataset being closer to the global distribution that FL is aiming to optimize over. The benefit of local dataset diversification has been established analytically in existing works for supervised FL, e.g., [46], [47]. CF-CL builds upon this theoretical foundation, as the push-pull mechanism employed by each device assigns importance to datapoints/embeddings at their neighbors according to the dissimilarity from their local distribution.

(2) *Micro Sampling:* In *micro sampling* the probability of sampling individual datapoints are calculated to assign a data-level importance separate from the cluster assignments via (8).

<sup>2</sup>Note that we have  $\sum_{\ell \in \mathcal{L}_j^t} K_{j \rightarrow i}^{t, \text{Approx}}(\ell) = K_j^{\text{Approx}}, \forall t$  and  $\sum_{\ell \in \mathcal{L}_j^t} K_{i \rightarrow j}^{t, \text{Reserve}}(\ell) = K_{i \rightarrow j}^{\text{Reserve}}, \forall t$ .

We assign a probability  $P_{j \rightarrow i}^{t, \text{Micro}}(\hat{d})$  to data point  $\hat{d}$  according to the expected loss when it is used as a negative relative to datapoints in  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}$  used as anchors:

$$\mathbb{E}_{d \sim \mathcal{D}_{i \rightarrow j}^{\text{Reserve}}} \left[ L_{\phi_G^{\gamma T_a}}(d, F(d), \hat{d}) \right] = \frac{\sum_{d \in \mathcal{D}_{i \rightarrow j}^{\text{Reserve}}} L_{\phi_G^{\gamma T_a}}(d, \tilde{d}, \hat{d})}{K_{i \rightarrow j}^{\text{Reserve}}}. \quad (10)$$

The probability of selection of datapoint  $\hat{d} \in \ell$  is then computed as

$$P_{j \rightarrow i}^{t, \text{Micro}}(\hat{d}) = \frac{\exp \left( \lambda^t \cdot \mathbb{E}_{d \sim \mathcal{D}_{i \rightarrow j}^{\text{Reserve}}} \left[ L_{\phi_G^{\gamma T_a}}(d, \tilde{d}, \hat{d}) \right] \right)}{\sum_{\hat{d}' \in \ell} \exp \left( \lambda^t \cdot \mathbb{E}_{d \sim \mathcal{D}_{i \rightarrow j}^{\text{Reserve}}} \left[ L_{\phi_G^{\gamma T_a}}(d, \tilde{d}, \hat{d}') \right] \right)} \quad (11)$$

In (11),  $\lambda^t$  is the *selection temperature*, a user-defined hyperparameter which is used to make our selection algorithm robust against the loss values becoming more homogeneous. Considering (11), our selection algorithm aims to improve the model training performance by prioritizing the transmission of datapoints which produce a higher loss at the receiver (measured via their loss over  $\mathcal{D}_{i \rightarrow j}^{\text{Reserve}}$ ). Intuitively, the choice of  $\lambda^t$  defines the degree of greediness with which the algorithm selects important datapoints.

Combining *macro sampling* and *micro sampling*, when device  $i$  pulls  $n_{j \rightarrow i}^t$  datapoints from  $j$ , they are selected from device  $j$ 's candidate samples  $\mathcal{D}_j^{t, \text{Approx}}$ . The probability of sampling of each datapoint  $\hat{d} \in \mathcal{D}_j^{\text{Approx}}$  given by

$$P_{j \rightarrow i}^t(\hat{d}) = P_{j \rightarrow i}^{t, \text{Micro}}(\hat{d}) \times P_{j \rightarrow i}^{t, \text{Macro}}(\ell_{\hat{d}}), \hat{d} \in \mathcal{D}_j^{\text{Approx}}, \quad (12)$$

Where  $\ell_{\hat{d}}$  is the cluster that  $\hat{d}$  is assigned to via the K-means++ procedure that generates  $\mathcal{L}_{j \rightarrow i}$ .

### C. Smart Push-Pull of Implicit Information and Triplet Loss Regularization

In Sec. III-B, we allowed for explicit information exchange in the form of data. We now consider the exchange of embeddings or implicit information, which is smaller in size and also more desirable in privacy sensitive applications. These benefits come at the cost of performance compromises compared to data exchange, as we will see experimentally in Sec. IV.

When implicit information is exchanged between devices, the local ML training and global model aggregations of CF-CL are similar to those under explicit information exchange and follow (4) and (5) with two differences: (i) The datapoints used for training at each device are static (i.e.,  $\mathcal{D}_i^t = \mathcal{D}_i, \forall t$ ), since only the embeddings are exchanged, and (ii) a modified definition for the triplet loss function  $L_{\phi}(d, \tilde{d}, \hat{d})$  is used which includes a regularization term to incorporate exchanged implicit information, which is explained in this section.

Similar to explicit push-pull, our method consists of pushing embeddings from each device to its neighboring devices, and are used to identify local embeddings that contribute the most to model performance, followed by pulling important embeddings from the neighboring devices.

### Algorithm 3: Implicit Information Pull by Device $i$ from Device $j \in \mathcal{N}_i$

- 1: **function**  $Q'((j, \mathcal{Z}_{i \rightarrow j}^{\text{Reserve}}, \phi_G^{\gamma T_a}, n_{j \rightarrow i}^t))$
- 2: Transmitter  $j$  receives reserve embeddings  $\mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}$  from device  $i$  as (13)
- 3: Transmitter  $j$  approximates local dataset  $\mathcal{D}_j^{t, \text{Approx}}$  as (7)
- 4: Transmitter  $j$  obtains local embeddings  $\mathcal{Z}_j^{t, \text{Approx}}$  as (14)
- 5: Transmitter  $j$  performs KMeans++ to obtain clusters of local embeddings  $\mathcal{H}_j^t$   $\triangleright$  *Macro Importance*
- 6: Transmitter  $j$  obtains probability distribution for clusters according to (17)
- 7: Transmitter  $j$  obtains probability distribution for embeddings in each cluster according to (21) and (19)  $\triangleright$  *Micro Importance*
- 8: Transmitter  $j$  samples  $n_{j \rightarrow i}^t$  embeddings to obtain set  $\mathcal{Z}_{j \rightarrow i}^t$  according to probabilities obtained in (22)  $\triangleright$  *Embedding Sampling*
- 9: Transmitter  $j$  transmits  $\mathcal{Z}_{j \rightarrow i}^t$  to device  $i$   $\triangleright$  *Embedding Transfer*
- 10: **end function**

1) *Smart Push of Implicit Information:* At each global aggregation time-step  $t = \gamma T_a, \forall \gamma$ , each device first pushes reserve embeddings  $\mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}$  to its neighboring devices, providing them with fresh embeddings to use for importance measurements. To this end, each device  $i$  samples a set of representative local datapoints  $\mathcal{D}_i^{t, \text{Reserve}}$  according to (6) and obtains their embeddings  $\mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}$ , which are then pushed to device  $j$ , where<sup>3</sup>

$$\mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}} = \left\{ z : z = \phi_G^{\gamma T_a}(d), d \in \mathcal{D}_i^{t, \text{Reserve}} \right\}, t = \gamma T_a. \quad (13)$$

2) *Smart Pull of Implicit Information:* Similar to data sampling in Sec. III-B, our embedding selection method consists of: (i) identifying and sampling clusters of embeddings proportional to their importance; and (ii) the selection of high importance embeddings from the sampled clusters. A detailed description of these processes is given in Sec. III-C2.

In the case of implicit exchange, local embeddings are a function of the global model  $\phi_G^{\gamma T_a}$ , and thus change over the course of training. Hence, as with explicit information, it is necessary for the implicit information to be shared regularly over time so that it reflects the current global model.

At each embedding pull instance  $\tau$ , i.e.,  $t = \tau T_p$ , which occurs between two global aggregation rounds  $\gamma$  and  $\gamma + 1$  (i.e.,  $\gamma T_a \leq \tau T_p < (\gamma + 1)T_a$ ), the embeddings pulled by device  $i$  from device  $j$ , denoted by set  $\mathcal{Z}_{j \rightarrow i}^t$ , are obtained by execution of probabilistic function  $Q'$ , where  $\mathcal{Z}_{j \rightarrow i}^t = Q'(j, \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}, \phi_G^{\gamma T_a}, n_{j \rightarrow i}^t) \subseteq \{\phi_j^t(d)\}_{d \in \mathcal{D}_j}$ . We provide a summary of  $Q'$  in Alg. 3 and detail it below. After  $\mathcal{Z}_{j \rightarrow i}^t$  is obtained, device  $j$  first selects a set of representative local datapoints  $\mathcal{D}_j^{t, \text{Approx}}$  according to (7) and generates their embeddings  $\mathcal{Z}_j^{t, \text{Approx}}$  as follows:

$$\mathcal{Z}_j^{t, \text{Approx}} = \left\{ z : z = \phi_G^{\gamma T_a}(d), d \in \mathcal{D}_j^{t, \text{Approx}} \right\}, t = \tau T_p. \quad (14)$$

<sup>3</sup>As compared to explicit push-pull in Sec. III-B where data push is conducted only once at the beginning of training, in implicit push-pull we need to conduct the push procedure after every global aggregation since the embeddings are generated using the current global model.

Elements of  $\mathcal{Z}_j^{t, \text{Approx}}$  are candidate embeddings from which the most important are to be selected and transmitted to the neighboring devices. Note that for each neighboring device  $i \in \mathcal{N}_j$ , most important embeddings in  $\mathcal{Z}_j^{t, \text{Approx}}$  are dissimilar since the data distribution at the devices is non-i.i.d. We select the most important embeddings using a *two-stage probabilistic importance sampling* procedure, consisting of macro and micro sampling steps, similar to Sec. III-B3.

(1) *Macro Sampling*: partitions  $\mathcal{Z}_j^{t, \text{Approx}}$  using KMeans++ to  $Z^{\text{local}}$  clusters of local embeddings denoted by set  $\mathcal{H}_j^t$ . Similarly, KMeans++ is performed on  $\mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}$  to obtain  $Z^{\text{Reserve}}$  clusters.

For exchange from device  $j$  to neighboring device  $i$ , the sampling probability distribution for each cluster  $h$  in  $\mathcal{H}_j^t$ , is found by calculating the aggregate importance of the embeddings within that cluster using reserve embeddings  $\mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}$  via the score metric

$$S(h, \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}) = \frac{\sum_{z \in h} s(z, \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}})}{|\mathcal{H}_j^t[h]|}, \quad (15)$$

where  $|\mathcal{H}_j^t[h]|$  denotes the number of datapoints in cluster  $h$ . In (15),  $s(z, \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}})$  is the importance score of embedding  $z$  relative to the embeddings in  $\mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}$  computed as

$$s(z, \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}) = \max_{z \in h} (0, \|z - \mu_h\|_2^2) \sum_{z' \in \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}} (\|z' - z\|_2^2), \quad (16)$$

where  $\mu_h$  is the centroid of cluster  $h$  if embedding  $z$  belongs to cluster  $h$ .

Intuitively, this formulation ensures that embeddings that are closer to the centroid of the cluster are selected more frequently, and thus the set of selected embeddings is a better representative of its parent cluster. Our importance computation rule in (16) is tailored to the triplet loss computation rule (1) and assigns a higher importance score to those embeddings which are closer to the embeddings in  $\mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}$ , since such embeddings constitute the *hard negatives* (i.e., they contribute more to the local training). Using the cluster importance score (15), we obtain the sampling probability of each cluster  $h$  as follows:

$$P_{j \rightarrow i}^{t, \text{Macro}}(h) = \frac{S(h, \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}})}{\sum_{h' \in \mathcal{H}_j^t} S(h', \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}})}. \quad (17)$$

Transmitting embeddings from clusters which have a significant overlap with  $Z^{\text{Reserve}}$  should be avoided, as these clusters have a high probability of containing embeddings similar to  $\mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}}$ , which will negatively impact performance if used as regularizers. We calculate the overlap  $B(h)$  of local clusters  $h$  with centroid  $c^h$  as follows:

$$B(h) = \text{PDF}(b(h), \hat{\mu}, \hat{\sigma}), \quad (18)$$

$$b(h) = \frac{\frac{\sum_{i \in Z^{\text{Reserve}}} \|c^h - c^i\|_2^2}{|Z^{\text{Reserve}}|} - \frac{\sum_{i \in Z^{\text{Local}}} \|c^h - c^i\|_2^2}{|Z^{\text{Local}}| - 1}}{\frac{\sum_{i \in Z^{\text{Local}}} \|c^h - c^i\|_2^2}{|Z^{\text{Local}}| - 1}}, \quad (19)$$

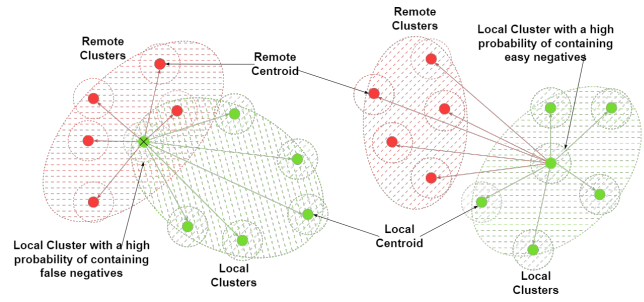


Fig. 3: Centroids of remote clusters (green) that are closer to local clusters (red) have a higher chance of being similar to local data (Left), while centroids that are further away are less likely (Right).

where PDF is the probability density function of a normal distribution.  $\hat{\mu}, \hat{\sigma}$ , the mean and standard deviation of the distribution respectively, are tunable parameters.

$B(h)$  functions as a scaling factor which is large for clusters that are equally close to remote and local clusters, indicating that there is a high likelihood of hard negatives being present in the cluster. Conversely, close proximity to remote clusters relative to local clusters indicate a higher possibility of false negatives and closer proximity to local clusters relative to remote clusters indicate the presence of easy negatives, and hence, the value  $B(h)$  for both cases is small, as shown in Fig. 3. We modify the probability of cluster  $h$  given in (17) as follows:

$$P_{j \rightarrow i}^{t, \text{Macro}}(h) = B(h) \cdot P_{j \rightarrow i}^{t, \text{Macro}}(h). \quad (20)$$

(2) *Micro Sampling*: In *micro sampling*, once  $P_{j \rightarrow i}^{t, \text{Macro}}$  is obtained, the sampling probability of each embedding  $z$  belonging to cluster  $h$  is calculated as follows:

$$P_{j \rightarrow i}^{t, \text{Micro}}(z) = \frac{s(z, \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}})}{\sum_{z' \in h} s(z', \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}})}, \quad z \in h. \quad (21)$$

Where  $s(z, \mathcal{Z}_{i \rightarrow j}^{t, \text{Reserve}})$  is obtained from (16).

Combining *macro sampling* and *micro sampling*, at each  $t = \tau T_p, \forall \tau$ , upon pulling  $n_{j \rightarrow i}^t$  embeddings by device  $i$  from  $j$ , the embeddings from  $\mathcal{D}_j^{t, \text{Approx}}$  are selected to obtain set  $\mathcal{Z}_{j \rightarrow i}^t$ , where  $n_{j \rightarrow i}^t = |\mathcal{Z}_{j \rightarrow i}^t|$ , with each embedding  $z$  sampled with probability

$$P_{j \rightarrow i}^t(z) = P_{j \rightarrow i}^{t, \text{Macro}}(h) \times P_{j \rightarrow i}^{t, \text{Micro}}(z), \quad z \in h, \quad h \in \mathcal{H}_j^t. \quad (22)$$

Note that in the above procedure, we obtain  $\mathcal{Z}_{j \rightarrow i}^t$  only at embedding exchange instances  $t = \tau T_p, \forall \tau$ . Nevertheless, we will define  $\mathcal{Z}_{j \rightarrow i}^t, 1 \leq t \leq T$  to facilitate our explanations in the following discussions. Since the previously received embeddings are kept at each device until receiving new embedding, we define  $\mathcal{Z}_{j \rightarrow i}^t \triangleq \mathcal{Z}_{j \rightarrow i}^t |_{t=\tau T_p}$ , where  $\tau T_p \leq t < (\tau + 1)T_p$ .

3) *Integrating Implicit Information into Local ML Training via Triplet Loss Regularization*: As opposed to explicit information exchange, implicit information cannot be directly used in the local ML training as they are generated according to the local model at each transmitting device. Hence, the received device cannot modify them according to its own local model during SGD iterations (4) since their corresponding input datapoints are not locally available.



In order to incorporate the information contained in the exchanged embeddings for local training at the receiver, we propose integrating the received embeddings at each device  $i$  into its local ML model training by introducing a regularization term into the triplet loss formulation (1). In particular, we redefine the triplet loss at each device  $i$  as follows for the duration of local learning  $\gamma T_a \leq t < (\gamma + 1)T_a$ :

$$L_{\phi, i, t}(d, \hat{d}, \bar{d}) = \underbrace{\max\left[0, \|\phi(d) - \phi(\hat{d})\|_2^2 - \|\phi(d) - \phi(\bar{d})\|_2^2 + m\right]}_{\text{Contrastive Loss}} + W_t \underbrace{\sum_{j \in \mathcal{N}_i, z \in \mathcal{Z}_j^t} \max\left[0, \|\phi(d) - \phi(\hat{d})\|_2^2 - \|\phi(d) - z\|_2^2 + m^{\text{Reg}}\right]}_{\text{Regularization Term}}. \quad (23)$$

In (23), the regularization margin  $m^{\text{Reg}}$  is different from the triplet loss margin  $m$ , and adapts to the current model as follows. For local clusters  $h \in \mathcal{Z}^{\text{Local}}$  with radii  $r^h$  we define  $m^{\text{Reg}}$  as

$$m^{\text{Reg}} = k \cdot \frac{\sum_{h \in \mathcal{Z}^{\text{Local}}} r^h}{|\mathcal{Z}^{\text{Local}}|}. \quad (24)$$

The regularization margin  $m^{\text{Reg}}$  ensures that for any anchor embedding in cluster  $h$ , all remote embeddings that lie within  $h$  are considered hard negatives. The scaling factor  $k$  indicates the volume of latent space that the model considers when looking for hard negatives. The objective of the regularization term is to incorporate the hard negatives identified via (16) into the local ML training. Since the received embeddings at each device  $i$ , i.e.,  $\cup_{j \in \mathcal{N}_i} \mathcal{Z}_{j \rightarrow i}^t$ , are computed at the instance of global model reception, they become obsolete as the local ML model training proceeds during the period  $\gamma T_a \leq t < (\gamma + 1)T_a$ . Hence, the regularization coefficient  $W_t$  is introduced in (23), which is used to put a higher weight on the received embeddings while they are fresh (i.e., at the beginning of local ML training) and gradually diminishes as the local model changes and the embeddings become obsolete. Subsequently, to capture the effect of the staleness of received embeddings, we propose the following policy for the regularization coefficient:

$$W_t = \lambda \cdot \left( \exp\left(\frac{-t \bmod T_a}{T_a - 1}\right) + \exp\left(\frac{t}{T} - \varrho \zeta_t\right) \right). \quad (25)$$

The first term in  $W_t$  forms a sawtooth shape which takes its maximum value at the time-step of global aggregation  $t = \gamma T_a$ ,  $\gamma \in \mathbb{Z}^+$ . This term naturally puts a higher weight on the regularization when the received embeddings are fresh. The second term reflects the fact that as the models are trained, model embedding spaces become less transient, and the effect of staleness reduces.

It is worth mentioning that regularization terms have been previously explored in FL research via the well known work [48] and its subsequent literature in a different context. Implementations such as [48] consider proximal regularization terms imposed on the model parameters (i.e., neural network parameters) to enhance the convergence of FL over non-i.i.d data. The focus of our work and in turn our regularization term exploits the information exchange in unsupervised FL, which is fundamentally different from previous literature focusing on model parameter weights. Hence, the regularization term in [48] can be further mounted on top of our technique.

## IV. NUMERICAL EXPERIMENTS

### A. Simulation Setup

In our numerical experiments, we use the Fashion MNIST dataset [49], the USPS Handwritten Digits dataset [50] and the Street View House Numbers (SVHN) dataset [51]. The Fashion MNIST dataset and SVHN dataset consist of 60K images with 10 classes and the USPS dataset consists of 7291 images with 10 classes. We consider a network of  $|\mathcal{C}| = 10$  devices and emulate non-i.i.d. data across devices as discussed below.

For Fashion MNIST, each device is allocated 6K datapoints from only three out of 10 classes. For this dataset, we use the Alexnet architecture [52] with an output size of 16. The Adam optimizer is used with an initial learning rate of  $10^{-4}$  and devices' models are trained for  $T = 2000$  local SGD iterations. Data augmentation consists of random resized crops, random horizontal flips, and Gaussian blurs [53]. Unless otherwise stated, we set  $K_{i \rightarrow j}^{\text{Push}} = 20$ , and  $K_j^{\text{Approx}} = 100$ , and local K-means employs 20 clusters.

For USPS, each device is allocated  $\sim 730$  datapoints from only three out of 10 classes. For this dataset, we use a CNN with a single convolutional layer with 8 kernels, each of size  $3 \times 3$ , followed by 3 linear layers of sizes 1024, 256 and 16. The Adam optimizer is used with an initial learning rate of  $10^{-3}$  and models of devices are trained for  $T = 1500$  local SGD iterations. Data augmentation consists of random rotations and random perspective transformations. We set  $K_{i \rightarrow j}^{\text{Push}} = 10$ , and  $K_j^{\text{Approx}} = 100$ , and local K-means employs 10 clusters.

For SVHN, each device is allocated 6K datapoints from only three out of 10 classes. For this dataset, we use the Resnet-18 architecture [54] with an output size of 256. The Adam optimizer is used with an initial learning rate of  $10^{-4}$  and devices' models are trained for  $T = 4000$  local SGD iterations. Data augmentation consists of random resized crops, random horizontal flips, and Gaussian blurs. Unless otherwise stated, we set  $K_{i \rightarrow j}^{\text{Push}} = 25$ , and  $K_j^{\text{Approx}} = 200$ , and local K-means employs 25 clusters.

We conduct simulations on a server with 48GB Tesla-P100 GPU with 128GB RAM. All hyperparameters are identical for implicit and explicit data exchange algorithms.

To obtain the accuracy of predictions, we adopt the linear evaluation [15], and use  $\phi_G^t, \forall t$ , to train a linear layer  $\theta$  in a supervised manner on top of  $\phi_G^t$  to perform a classification at the server. The linear layer is trained over 1000 SGD iterations. As mentioned in Sec. I, smart data transfer has not been studied in the context of unsupervised federated learning, and literature [29], [30], [11] have only considered uniform data transfer across the network. Thus, we compare the performance of CF-CL against four baselines: (i) *uniform sampling*, where data points transferred are sampled uniformly at random from the local datasets; (ii) *bulk sampling*, where importance sampling and data sharing is done only at the beginning of training, and the amount of information shared is equivalent to what CF-CL would share over the course of training; (iii) *K-Means exchange*, where the information to be shared is selected by the transmitter using K-Means clustering (i.e., in place of lines 3 to 5 of Algorithm 2 and lines 2 to 7 of Algorithm 3), with information that is closest to the centroid of the clusters selected to be

shared; and (iv) classic federated learning (FedAvg), which does not conduct any data transfer across devices.

The communication graph  $\mathcal{G}$  is assumed to be random geometric graph (RGG), which is a common model used for wireless peer-to-peer networks. We follow the same procedure as in [18] to create RGG with average node degree 7. For Fashion MNIST, devices conduct  $T_a = 25$  local SGD iterations and exchange data after  $T_p = 25$  iterations, and for USPS,  $T_a = 10$  and  $T_p = 25$ .

## B. Comparison against Baselines

**Embedding Alignment:** In Fig. 4, we consider the embeddings generated by conventional FL (Fig. 4(a)), explicit CF-CL (Fig. 4(b)) and implicit CF-CL (Fig. 4(c)) and calculate the Euclidian distance between all pairs of embeddings from all combinations of labels. We observe that smart data transfer in CF-CL in both implicit and explicit cases leads to a latent space with more separated embeddings as compared to conventional FL, i.e., datapoints with distinct labels are further from each other as compared to those that have identical labels which appear on the diagonal of the heatmaps. While explicit CF-CL can distinguish embeddings to a larger extent, the separation produced by both explicit and implicit cases results in a significant improvement in performance, as we will show in Fig. 6.

**Training Convergence Performance:** In Fig. 5(a) to 5(f), we study the model training performance of explicit CF-CL, implicit CF-CL and baseline methods. Both explicit and implicit CF-CL outperform respective baselines in terms of achieved model accuracy over training iterations, due to their ability to select information crucial to the alignment of local models. More specifically, the improvement over baselines validates a few aspects of CF-CL's design: (a) exchanging information that is both representative of the transmitter's local dataset and important to the receiver (i.e., compared with K-means and Uniform), (b) selecting reserve information which accurately represents the local data/embedding distributions, and (c) adapting to the changes in local embedding distributions over the training duration (i.e., compared to Bulk). We observe that in all cases, explicit CF-CL converges faster than implicit CF-CL. However, we will now show that this comes at the cost of significantly more information exchanged over the network.

In Fig. 6(a) to 6(f), we measure the information overhead required to achieve performance milestones, i.e., to reach a particular testing accuracy percentage. In the explicit case of CF-CL, information exchanged are datapoints of size 784 bytes for Fashion MNIST, 3072 bytes for SVHN and 256 bytes for USPS, considering each datapoint to be of the format specified above. For FMNIST and USPS in the implicit case of CF-CL, the size of exchanged information is 64 bytes per embeddings, as embeddings of size 16 are used, with each element being a floating point value of size 4 bytes. For SVHN, the embeddings are of size 256, making the size of each embedding 1024 bytes. We observe that both explicit CF-CL and implicit CF-CL utilize less information to meet target performance thresholds than the respective baselines, particularly as the threshold increases. This indicates that

(a) despite the initial information overhead, the reserve data selection method of CF-CL leads to more efficient model training by efficiently selecting representative information, and (b) periodically exchanging information according to the updated model training state is more efficient in improving performance. We also observe that the information overhead required by implicit CF-CL is significantly smaller than explicit CF-CL, owing to the smaller size of the exchanged information in implicit CF-CL.

To further reveal the impact of faster convergence of CF-CL on network resource savings, we focus on the latency of model training as a performance metric in 6(g) to 6(l). We assume that transmission rate in D2D and uplink are 1Mbits/sec with 32 bits quantization applied on the model parameters (45433 for Fashion-MNIST, 26704 for USPS and 1142352 for SVHN) and 8 on datapoints, which results in an uplink transmission delay of  $45433 \times 32/10^6 \approx 1.45s$  per model parameter exchange for Fashion MNIST,  $26704 \times 32/10^6 \approx 0.855s$  for USPS and  $1142352 \times 32/10^6 \approx 36.566s$  for SVHN. Fashion MNIST has a D2D delay of  $28 \times 28 \times 8/10^6 \approx 6.2ms$  per data point exchange (each data point is a  $28 \times 28$  gray-scale image with each pixel taking 256 values), USPS has a D2D delay of  $16 \times 16 \times 8/10^6 \approx 2ms$  per data point exchange (each data point is a  $16 \times 16$  gray-scale image with each pixel taking 256 values) and SVHN has a D2D delay of  $3 \times 32 \times 8/10^6 \approx 24.5ms$  per data point exchange, in case of the explicit exchange (each data point is a  $32 \times 32$  image with 3 channels and each pixel taking 256 values). Our models produce embeddings of size 16 for both Fashion MNIST and USPS, thus resulting in a delay of  $16 \times 32/10^6 \approx 0.512ms$  per embedding in the implicit case (each embedding has a dimension 16, each being a floating point number). Similarly, for implicit exchange using SVHN and embedding size 256 results in a delay of  $256 \times 32/10^6 \approx 8ms$ . We also compute the extra computation time of CF-CL (i.e., the K-means and importance calculations) and that of uniform sampling and incorporate that into delay computations. In the last two rows of Fig. 6, we observe that both variants of CF-CL achieve performance milestones significantly faster than their respective baselines, indicating that (a) intelligent selection of reserve information and D2D information is crucial to delay reduction, (b) the delays incurred by D2D communication in CF-CL are compensated for by the savings in device-to-server communication delays, and (c) CF-CL cuts down on the upstream communication energy consumption required by reducing the number of global aggregations. We also observe explicit CF-CL being more efficient than implicit CF-CL in terms of convergence speed for both datasets.

The above results indicate a tradeoff between latency and amount of information exchanged between explicit and implicit CF-CL. We can conclude that implicit CF-CL is best for bandwidth limited applications where minimal information exchange between devices is crucial, while explicit CF-CL is best for latency critical applications.

**Importance of Exchanged Information:** In Fig. 7, we calculate the average distance between the embeddings of received information  $Z_{j \rightarrow i}^t$  and the approximate local latent space occupied by  $\{\phi_G^{T_a}(x)\} \forall x \in \mathcal{D}$ . The latent space

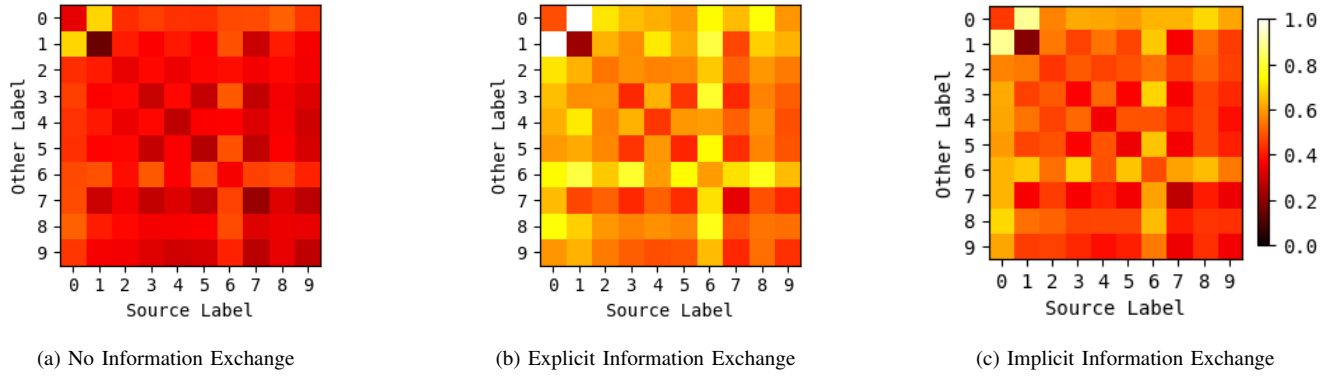


Fig. 4: Distance between embeddings of pairwise combinations of labels. Information exchange results in dissimilar embeddings being further apart from each other.

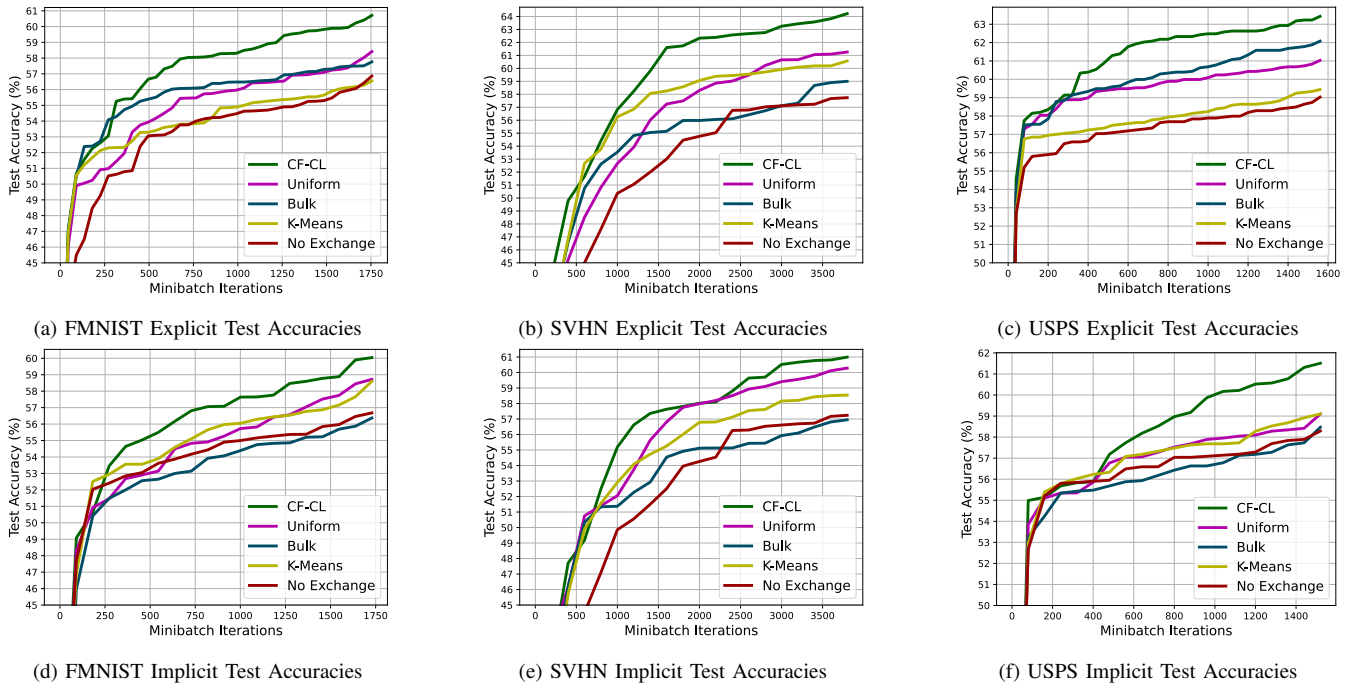


Fig. 5: Training performance comparison of CF-CL against baselines over training iterations, for each dataset and information exchange regime. We see that CF-CL has superior performance in all cases, validating the benefit of its information sharing mechanism for local model alignment.

is characterized by the centroids of  $\{\phi_G^{\gamma T_a}(x)\} \forall x \in \mathcal{D}$  after performing K-means. Intuitively, the closer the selected information is to local centroids in representation space, the more important it is. We construct a histogram of the average distance of each received embedding in  $\mathcal{Z}_{j \rightarrow i}^t$  to observe the distribution of average distance over received information. We observe that CF-CL, both implicit and explicit forms, can select information which is more important to the receiving device, resulting in the peaks of the histograms for CF-CL occurring at a distance closer to local data as compared to baseline methods.

**Number of Devices Selected for Aggregation:** In Fig. 8, we observe the effect of changing the number of devices selected per aggregation on the performance of CF-CL and compare it to baselines. In this experiment, local models are aggregated only from a subset of  $n$  devices uniformly at random

every  $T_a$  time-steps, which is then broadcast to all devices. Both implicit and explicit data exchange methods for CF-CL consistently outperform baselines. This indicates the ability of CF-CL adapt to conditions where only a subset of devices can communicate with the server for aggregation. This indicates that CF-CL is more resilient to a reduction in the number of local models being aggregated. This characteristic can be leveraged to conserve resources by reducing the frequency of communication of each device with the server.

### C. Impact of Variation in System Parameters

**Reserve Data Selection Methods:** In Fig. 9, we investigate the effect of using random sampling of  $K_{i \rightarrow j}^{\text{Push}}, \forall i, j$ , data points as reserved datapoints vs. K-means based selection (Sec. III-B2), in which device  $i$  selects reserve information by running a K-means algorithm on local data with  $K_{i \rightarrow j}^{\text{Push}}$

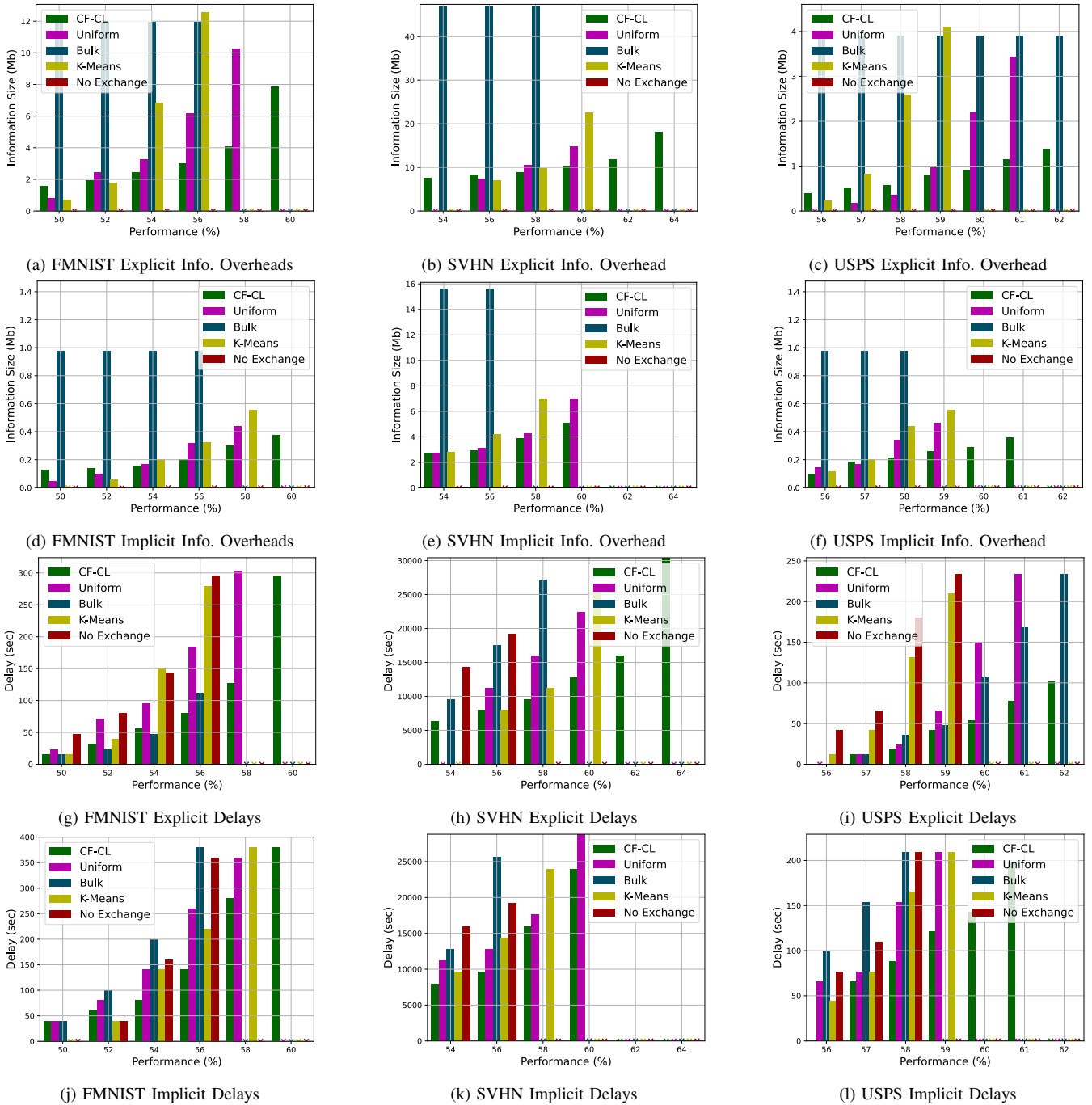


Fig. 6: CF-CL incurs a smaller information overhead (6(a) to 6(f)) and is quicker (6(g) to 6(l)) to achieve performance benchmarks compared to baselines. Explicit exchange achieves benchmarks faster than implicit exchange at the cost of larger information overhead. (× indicates that the method was unable to reach the performance threshold.)

clusters, under varying  $K_{i \rightarrow j}^{\text{Push}}$ . From Fig. 9, performance of both methods of CF-CL improve with selection of reserve data using K-Means. This is because K-means selects datapoints that best approximate the data distribution at each device. The effect of which is more significant in extreme cases, e.g.,  $K_{i \rightarrow j}^{\text{Push}} = 50, \forall i, j$ , and diminishes as the allowable number of pushed data increases. This is because, as the number of reserve data increases, data selected at device  $i$  by random sampling becomes more similar to the actual data distribution at device  $i$ .

**Local vs. Global Models for Importance Calculation:** Fig. 10 illustrates the impact of using either local or global models for importance calculations at the transmitter and receiver. For a constant exchange interval  $T_p = 25$ , when aggregation and exchange are asynchronous, the performance gap between using local model  $\phi_i^{\tau T_p}$  and using latest global model  $\phi_G^{\gamma T_a}$  where  $\gamma T_a \leq \tau T_p < (\gamma + 1)T_a$  for importance calculation increases as the aggregation interval  $T_a$  increases, indicating that CF-CL obtains performance gains even if models used for importance calculations are different. This

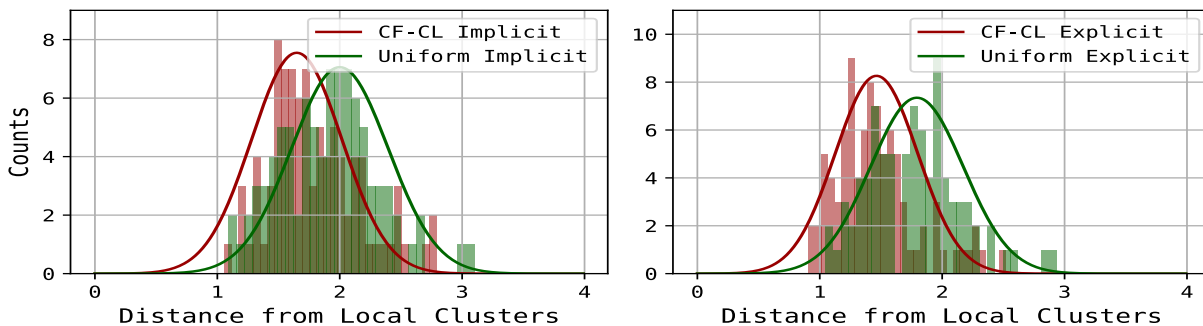


Fig. 7: Proximity of received embeddings to local embeddings. In both Implicit (left) and Explicit (right) cases, CF-CL selects information that is harder to distinguish as negatives in the latent space compared to information selected by the uniform baseline, and are hence more important.

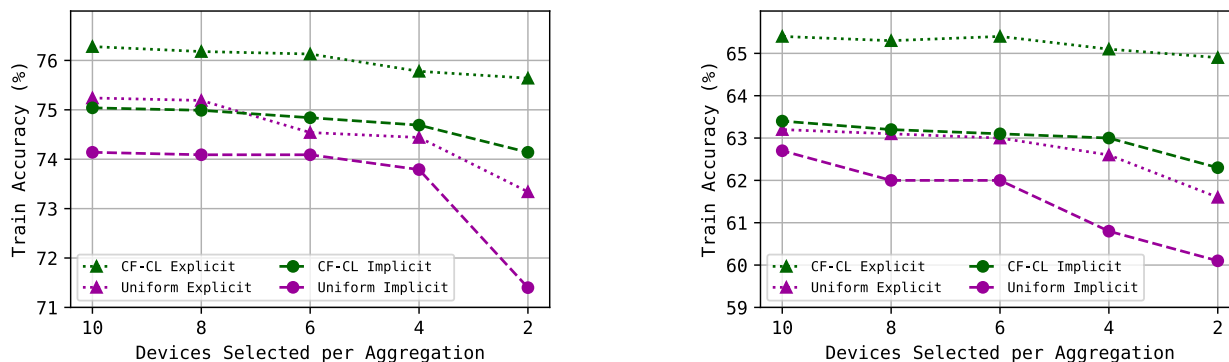


Fig. 8: Change in number of devices selected per iteration on performance of CF-CL against baselines for both datasets. CF-CL is more resilient to a reduction in the number of local models being aggregated.

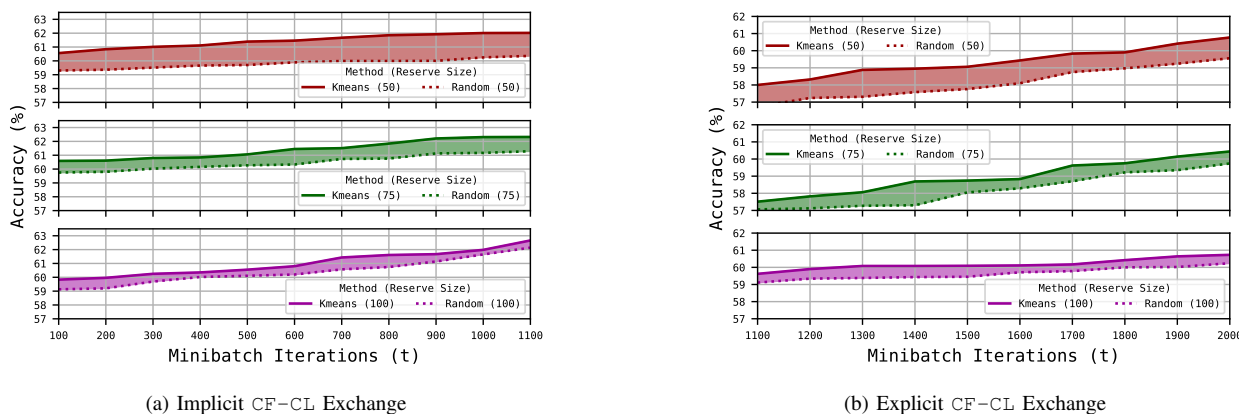


Fig. 9: Effects of using K-means and random sampling for reserve datapoints selection. Using K-means to select reserve datapoints results in better performance than selecting reserve data uniformly at random.

shows that for CF-CL, for the purposes of importance calculation, synchronization of local models at every exchange interval is not necessary for importance calculations, i.e. global aggregations are not needed at exchange intervals. This implies global knowledge is not necessary for importance calculation, which is desirable. We also observe that the explicit version of CF-CL is more resilient to local model differences, this is consistent with the fact that the information exchanged is not a function of the model at the transmitter.

**Local Data Availability and Device Connectivity:** Fig. 11

shows different scenarios of non-i.i.d data with varying connectivity between devices. We consider cases where the average node degree in the communication graph  $\mathcal{G}$  is 2 for a sparsely connected graph and 8 for a densely connected graph. We vary the number of labels in each device's local dataset and show that as the local data distributions become more non-i.i.d. (i.e., fewer labels per device), the speed of convergence of the methods drop due to more biased local models. In such cases, higher connectivity significantly improves the performance, as a higher connectivity allows for the exposure of local datasets to a more

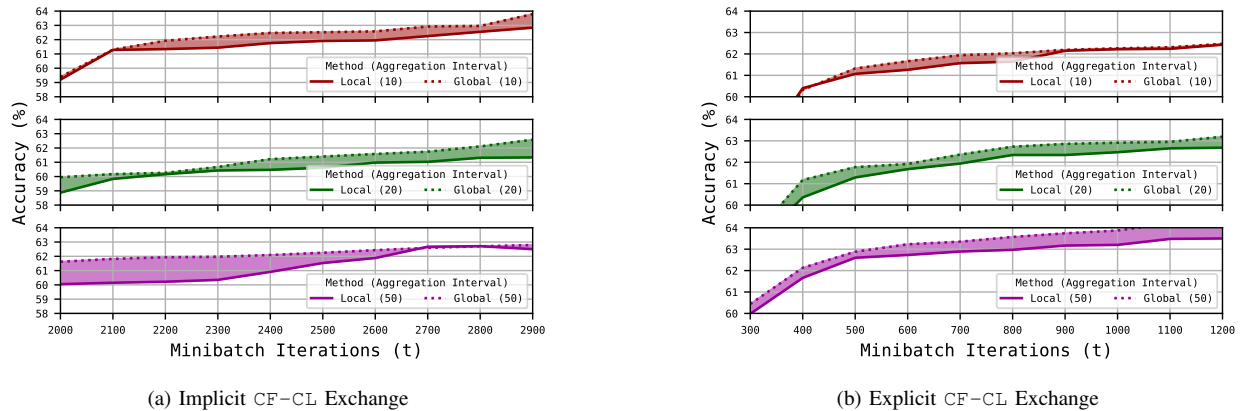


Fig. 10: Variation in performance of CF-CL when local and global models are used for importance calculations for both information exchange methods. CF-CL is resilient to differences in local models at transmitter and receiver.

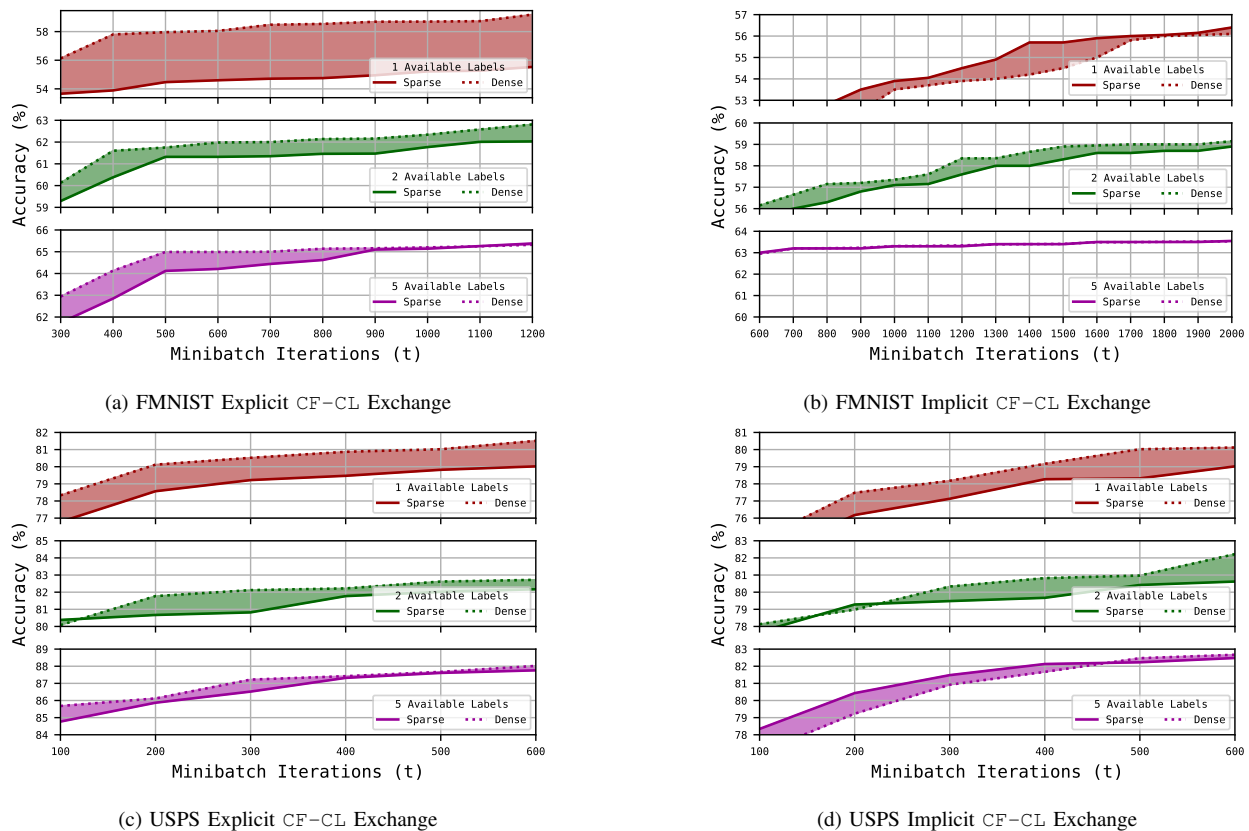


Fig. 11: Effect of connectivity of devices under varying conditions of non-i.i.d. data. Higher connectivity improves the performance of CF-CL in conditions where locally available data is not diverse.

diverse set of information resulting in less biased local models. We also find that in Fig. 11(a) and Fig. 11(c), the performance on the FMNIST dataset improves significantly more than on the USPS dataset, as classification using FMNIST is a more complex learning task, and thus affected by non-i.i.d. to a larger extent. We also observe that in the case of implicit information exchange, once the number of labels per device become large enough, the performance of our method for sparse and densely connected graphs is either equivalent as in Fig. 11(b) or our method performs better in a sparsely connected graph as in Fig. 11(d). This is because the implicit form of CF-CL employs

regularization to incorporate implicit information into training, as described in (23). A densely connected graph of devices, where each device has local access to a large number of labels may result in the exchange of redundant information, resulting in a slower convergence rate, especially in the earlier stages of training.

## V. CONCLUSION AND FUTURE WORK

In this paper, we proposed Cooperative Federated unsupervised Contrastive Learning (CF-CL). In CF-CL, devices learn local representations of unlabeled data and engage in cooperative

smart explicit or implicit information push-pull to eliminate the local model bias. In case of explicit information exchange, we proposed an efficient randomized data importance estimation and subsequently developed a two-staged probabilistic data sampling scheme across the devices. In case of implicit information exchange, we designed a probabilistic embedding exchange scheme and modified the definition of triplet loss function to use the exchanged embeddings while taking into account for their freshness. Through numerical simulations, we studied the model training behavior of CF-CL and showed that it outperforms the baseline methods in terms of accuracy and efficiency. Future work can extend CF-CL from single-hop information sharing to multi-hop information sharing and D2D link formation under network constraints. Such a system will require a more nuanced method of information importance estimation to take into account first hop neighbors, as well as higher order neighbours in the D2D graph.

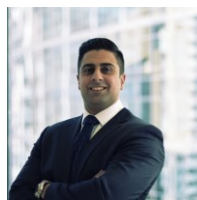
## REFERENCES

- [1] S. Wagle, S. Hosseinalipour, N. Khosravan, M. Chiang, and C. G. Brinton, "Embedding alignment for unsupervised federated learning via smart data exchange," in *IEEE Global Communications Conference*, 2022, pp. 492–497.
- [2] F. Zantalis, G. Koulouras, S. Karabetsos, and D. Kandris, "A review of machine learning and IoT in smart transportation," *Future Internet*, vol. 11, no. 4, 2019.
- [3] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, 2020.
- [4] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artif. Intell. Stat.*, 2017, pp. 1273–1282.
- [5] C. Fan and J. Huang, "Federated few-shot learning with adversarial learning," in *Int. Symp. Model. Opt. Mobile, Ad hoc, Wireless Netw. (WiOpt)*, 2021, pp. 1–8.
- [6] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, 2019.
- [7] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-iid data with reinforcement learning," in *IEEE Conf. Comput. Commun. (INFOCOM)*, 2020, pp. 1698–1707.
- [8] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-IID data," in *Int. Joint Conf. Neural Netw. (IJCNN)*, 2020, pp. 1–9.
- [9] Z. Zhao, C. Feng, W. Hong, J. Jiang, C. Jia, T. Q. Quek, and M. Peng, "Federated learning with non-IID data in wireless networks," *IEEE Trans. Wireless Commun.*, 2021.
- [10] B. van Berlo, A. Saeed, and T. Ozcelebi, "Towards federated unsupervised representation learning," in *ACM Int. WKSHP Edge Syst. Anal. Netw.*, 2020, pp. 31–36.
- [11] F. Zhang, K. Kuang, L. Chen, Z. You, T. Shen, J. Xiao, Y. Zhang, C. Wu, F. Wu, Y. Zhuang *et al.*, "Federated unsupervised representation learning," *Frontiers of Information Technology & Electronic Engineering*, vol. 24, no. 8, pp. 1181–1193, 2023.
- [12] M. Servetnyk, C. C. Fung, and Z. Han, "Unsupervised federated learning for unbalanced data," in *IEEE Global Commun. Conf.*, 2020, pp. 1–6.
- [13] N. Lu, Z. Wang, X. Li, G. Niu, Q. Dou, and M. Sugiyama, "Federated learning from only unlabeled data with class-conditional-sharing clients," in *Int. Conf. Learn. Represent. (ICLR)*, 2022.
- [14] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *IEEE/CVF Conf. Comput. Vision Pattern Rec. (CVPR)*, 2021, pp. 10 713–10 722.
- [15] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Int. Conf. Machine Learn. (ICML)*, 2020, pp. 1597–1607.
- [16] W. Zhuang, Y. Wen, and S. Zhang, "Divergence-aware federated self-supervised learning," in *Int. Conf. on Learning Repr.*, 2022.
- [17] M. Adil, M. K. Khan, M. Jamjoom, and A. Farouk, "Mhadbor: Ai-enabled administrative-distance-based opportunistic load balancing scheme for an agriculture internet of things network," *IEEE Micro*, vol. 42, no. 1, pp. 41–50, 2022.
- [18] S. Hosseinalipour, S. S. Azam, C. G. Brinton, N. Michelusi, V. Aggarwal, D. J. Love, and H. Dai, "Multi-stage hybrid federated learning over large-scale D2D-enabled fog networks," *IEEE/ACM Trans. Netw.*, pp. 1–16, 2022.
- [19] B. Omoniwa, R. Hussain, M. Adil, A. Shakeel, A. K. Tahir, Q. U. Hasan, and S. A. Malik, "An optimal relay scheme for outage minimization in fog-based internet-of-things (iot) networks," *IEEE IoT J.*, vol. 6, no. 2, pp. 3044–3054, 2019.
- [20] L. Chen, L. Fan, X. Lei, T. Q. Duong, A. Nallanathan, and G. K. Karagiannidis, "Relay-assisted federated edge learning: Performance analysis and system optimization," *IEEE Trans. on Commun.*, vol. 71, no. 6, pp. 3387–3401, 2023.
- [21] S. Wang, S. Hosseinalipour, V. Aggarwal, C. G. Brinton, D. J. Love, W. Su, and M. Chiang, "Towards cooperative federated learning over heterogeneous edge/fog networks," *IEEE Communications Magazine*, 2023.
- [22] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Netw. App.*, vol. 18, no. 1, pp. 129–140, 2013.
- [23] S. Wan, X. Li, Y. Xue, W. Lin, and X. Xu, "Efficient computation offloading for internet of vehicles in edge computing-assisted 5G networks," *J. Supercomputing*, vol. 76, no. 4, pp. 2518–2547, 2020.
- [24] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, "Learning-based computation offloading for iot devices with energy harvesting," *IEEE Trans. Veh. Tech.*, vol. 68, no. 2, pp. 1930–1941, 2019.
- [25] C. I. Bercea, B. Wiestler, D. Rueckert, and S. Albarqouni, "Feddis: Disentangled federated learning for unsupervised brain pathology segmentation," *arXiv:2103.03705*, 2021.
- [26] D. Bui, K. Malik, J. Goetz, H. Liu, S. Moon, A. Kumar, and K. G. Shin, "Federated user representation learning," *arXiv:1909.12535*, 2019.
- [27] M. Liu, S. Ho, M. Wang, L. Gao, Y. Jin, and H. Zhang, "Federated learning meets natural language processing: A survey," *arXiv preprint arXiv:2107.12603*, 2021.
- [28] S. S. Azam, T. Kim, S. Hosseinalipour, C. Joe-Wong, S. Bagchi, and C. Brinton, "Can we generalize and distribute private representation learning?" in *Int. Conf. Artif. Intell. Stat.*, 2022, pp. 11 320–11 340.
- [29] S. Wang, M. Lee, S. Hosseinalipour, R. Morabito, M. Chiang, and C. G. Brinton, "Device sampling for heterogeneous federated learning: Theory, algorithms, and implementation," in *IEEE Conf. Comput. Commun. (INFOCOM)*, 2021, pp. 1–10.
- [30] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.
- [31] S. Hosseinalipour, C. G. Brinton, V. Aggarwal, H. Dai, and M. Chiang, "From federated to fog learning: Distributed machine learning over heterogeneous wireless networks," *IEEE Commun. Mag.*, vol. 58, no. 12, pp. 41–47, 2020.
- [32] P. Corke, T. Wark, R. Jurdak, W. Hu, P. Valencia, and D. Moore, "Environmental wireless sensor networks," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1903–1917, 2010.
- [33] Z. Ji, L. Chen, N. Zhao, Y. Chen, G. Wei, and F. R. Yu, "Computation offloading for edge-assisted federated learning," *IEEE Trans. on Veh. Tech.*, vol. 70, no. 9, pp. 9330–9344, 2021.
- [34] Y. Khaluf, E. Mathews, and F. J. Rammig, "Self-organized cooperation in swarm robotics," in *2011 14th IEEE Int. Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, 2011, pp. 217–226.
- [35] M. Ali, F. Naeem, M. Tariq, and G. Kaddoum, "Federated learning for privacy preservation in smart healthcare systems: A comprehensive survey," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 2, pp. 778–789, 2023.
- [36] I. Bistrütz, A. J. Mann, and N. Bambos, "Distributed distillation for on-device learning," in *Int. Conf. Neural Inf. Process. Syst.*, 2020.
- [37] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, 2020.
- [38] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation," *arXiv:1910.03581*, 2019.
- [39] Z. Long, J. Wang, Y. Wang, H. Xiao, and F. Ma, "Fedcon: A contrastive framework for federated semi-supervised learning," *arXiv:2109.04533*, 2021.
- [40] D. Csiba and P. Richtárik, "Importance sampling for minibatches," *J. Machine Learn. Res.*, vol. 19, no. 1, pp. 962–982, 2018.

- [41] P. Zhao and T. Zhang, "Stochastic optimization with importance sampling for regularized loss minimization," in *Int. Conf. on machine learning*, PMLR, 2015, pp. 1–9.
- [42] X. Dong and J. Shen, "Triplet loss in siamese network for object tracking," in *Eur. Conf. Comput. Vision (ECCV)*, 2018, pp. 459–474.
- [43] E. Rizk, S. Vlaski, and A. H. Sayed, "Optimal importance sampling for federated learning," in *IEEE Int. Conf. Acous. Speech Signal Proc. (ICASSP)*, 2021, pp. 3095–3099.
- [44] F. P.-C. Lin, S. Hosseinalipour, S. S. Azam, C. G. Brinton, and N. Michelusi, "Semi-decentralized federated learning with cooperative D2D local model aggregations," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3851–3869, 2021.
- [45] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *ACM-SIAM Symp. Disc. Alg.*, 2007, pp. 1027–1035.
- [46] S. Wang, M. Lee, S. Hosseinalipour, R. Morabito, M. Chiang, and C. G. Brinton, "Device sampling for heterogeneous federated learning: Theory, algorithms, and implementation," in *IEEE Conf. on Computer Communications*, 2021, pp. 1–10.
- [47] S. Hosseinalipour, S. Wang, N. Michelusi, V. Aggarwal, C. Brinton, D. Love, and M. Chiang, "Parallel successive learning for dynamic distributed model training over heterogeneous wireless networks," *IEEE/ACM Transactions on Networking*, pp. 1–16, 2023, publisher Copyright: IEEE.
- [48] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Machine Learn. Sys. (MLSys)*, vol. 2, pp. 429–450, 2020.
- [49] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms." [Online]. Available: <http://arxiv.org/abs/1708.07747>
- [50] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. on Pattern Analysis and Machine Intell.*, vol. 16, no. 5, pp. 550–554, 1994.
- [51] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," 2011.
- [52] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.
- [53] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. of big data*, vol. 6, no. 1, pp. 1–48, 2019.
- [54] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conf. on computer vision and pattern recognition*, 2016, pp. 770–778.



**Satyavrat Wagle** Satyavrat Wagle received his MS Degree from Carnegie Mellon University in 2018 and is currently pursuing the PhD Degree from Purdue University. His research interests include unsupervised distributed learning and machine learning for networks and communication systems.



**Seyyedali Hosseinalipour** Seyyedali Hosseinalipour (Member, IEEE) received the B.S. degree in electrical engineering from Amirkabir University of Technology, Tehran, Iran, in 2015 with high honor and top-rank recognition. He then received the M.S. and Ph.D. degrees in electrical engineering from North Carolina State University, NC, USA, in 2017 and 2020, respectively. He is currently an assistant professor at the Department of Electrical Engineering at the University at Buffalo (SUNY). He has been the recipient of the ECE Doctoral Scholar of the Year Award (2020) and ECE Distinguished Dissertation Award (2021) at North Carolina State University; and Students' Choice Teaching Excellence Award at University at Buffalo (SUNY). He has served as the TPC Co-Chair of workshops related to distributed/federated learning and edge computing held in conjunction with IEEE INFOCOM, IEEE GLOBECOM, IEEE ICC, IEEE/CVF CVPR, IEEE MSN, and IEEE VTC. His research interests include the analysis of modern wireless networks, synergies between machine learning methods and fog computing systems, distributed/federated learning, and network optimization.



**Naji Khosravan** Naji Khosravan (IEEE/CVF member) received his B.Sc. in Electrical Engineering from Amirkabir University of Technology (Tehran Polytechnique), Tehran, Iran, in 2015. He received his M.Sc. and Ph.D. in Computer Science from the Center for Research In Computer Vision (CRCV) at the University of Central Florida, FL, USA in 2019. His research focus spans Human Centered AI, Computer Vision, Deep Learning and Medical Image Analysis. He has been an active contributor to IEEE CVF conferences in the form of workshop organizer and reviewer. He has been publishing his work in different venues such as CVPR, WACV, MICCAI, IEEE GLOBECOM, IEEE EMBC and ACM. He is currently an Applied Science manager at Zillow group where his teams are focusing developing cutting edge AI solutions.



**Christopher G. Brinton** Christopher G. Brinton (S'08, M'16, SM'20) is the Elmore Rising Star Assistant Professor of Electrical and Computer Engineering (ECE) at Purdue University. His research interest is at the intersection of networking, communications, and machine learning, specifically in fog/edge network intelligence, distributed machine learning, and AI/ML-inspired wireless network optimization. Dr. Brinton is a recipient of the National Science Foundation (NSF) CAREER Award, Office of Naval Research (ONR) Young Investigator Program (YIP) Award, Defense Advanced Research Projects Agency (DARPA) Young Faculty Award (YFA), Air Force Office of Scientific Research (AFOSR) YIP Award, Intel Rising Star Faculty Award, and roughly \$16M in sponsored research projects as a PI or co-PI. He has also been awarded Purdue College of Engineering Faculty Excellence Awards in Early Career Research, Early Career Teaching, and Online Learning. He currently serves as an Associate Editor for IEEE/ACM Transactions on Networking, and previously was an Associate Editor for IEEE Transactions on Wireless Communications. Prior to joining Purdue, Dr. Brinton was the Associate Director of the EDGE Lab and a Lecturer of Electrical Engineering at Princeton University. He also co-founded Zoomi Inc., a big data startup company that has provided learning optimization to more than one million users worldwide and holds US Patents in machine learning for education. His book *The Power of Networks: 6 Principles That Connect our Lives* and associated Massive Open Online Courses (MOOCs) have reached over 400,000 students. Dr. Brinton received the PhD (with honors) and MS Degrees from Princeton in 2016 and 2013, respectively, both in Electrical Engineering.