ClusterFL: A Clustering-based Federated Learning System for Human Activity Recognition

XIAOMIN OUYANG, The Chinese University of Hong Kong, Hong Kong
ZHIYUAN XIE, The Chinese University of Hong Kong, Hong Kong
JIAYU ZHOU, Michigan State University, USA
JIANWEI HUANG, The Chinese University of Hong Kong, Shenzhen and Shenzhen Institute of Artificial Intelligence and Robotics for Society, China
GUOLIANG XING, The Chinese University of Hong Kong, Hong Kong

Federated Learning (FL) has recently received significant interests thanks to its capability of protecting data privacy. However, existing FL paradigms yield unsatisfactory performance for a wide class of human activity recognition (HAR) applications since they are oblivious to the intrinsic relationship between data of different users. We propose ClusterFL, a clustering-based federated learning system that can provide high model accuracy and low communication overhead for HAR applications. ClusterFL features a novel clustered multi-task federated learning framework that minimizes the empirical training loss of multiple learned models while automatically capturing the intrinsic clustering relationship among the nodes. We theoretically prove the convergence of proposed FL framework for non-convex and strongly convex models, and provide the guidance on selection of hyper-parameters for achieving such convergence. Based on the learned cluster relationship, ClusterFL can efficiently drop the nodes that converge slower or have little correlations with others in each cluster, significantly speeding up the convergence while maintaining the accuracy performance. We evaluate the performance of ClusterFL on an NVIDIA edge testbed using four new HAR datasets collected from 145 users. The results show that, ClusterFL outperforms several state-of-the-art FL paradigms in terms of overall accuracy, and can save more than 50% communication overhead.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing systems and tools**; • **Computing methodologies → Learning paradigms**.

Additional Key Words and Phrases: Activity recognition, federated learning, clustering, multi-task learning

ACM Reference Format:
Xiaomin Ouyang, Zhiyuan Xie, Jiayu Zhou, Jianwei Huang, and Guoliang Xing. 2022. ClusterFL: A Clustering-based Federated Learning System for Human Activity Recognition. *ACM Trans. Sensor Netw.* 18, 4, Article 1 (August 2022), 32 pages. https://doi.org/10.1145/1122445.1122456

This work is extended from a conference paper published at MobiSys 2021 [31].
This work is supported by the Research Grants Council (RGC) of Hong Kong, China, under GRF Grants No. 14203420 and No. 14209619, the Alzheimer's Drug Discovery Foundation, under Grant RDADB-201906-2019049, the Shenzhen Science and Technology Program (Project JCYJ20210324120011032), Guangdong Basic and Applied Basic Research Foundation (Project 2021B1515120008), and the Shenzhen Institute of Artificial Intelligence and Robotics for Society.
Authors' address: X. Ouyang, Z. Xie and G. Xing (corresponding author), Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong SAR 999077, China; email: {xmouyang,xavier_ie, glxing}@cuhk.edu.hk; J. Zhou, Department of Computer Science, Michigan State University, East Lansing, MI 48824 USA; email: jiayuz@egr.msu.edu; J. Huang (co-corresponding author), School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, Shenzhen 518172, China, and Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen 518129, China; email: jianweihuang@cuhk.edu.cn.

1550-4859/2022/8-ART1 $15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Recently deep learning has been increasingly adopted in human activity recognition [18, 36, 45]. Most of the previous work is focused on the *centralized learning* approach that needs to be trained centrally using all the data collected from users. However, collecting sensor data centrally imposes significant privacy concern for applications like longitudinal chronic condition monitoring [41, 45]. Moreover, the data of different users often exhibits a high level of heterogeneity due to diverse human biological features, physical environments and even sensor biases, which leads to poor model accuracy [18].

Federated learning (FL) [19] is a distributed machine learning approach, which only requires the nodes to upload model weights to avoid exposing users' raw data during the learning process. A typical federated learning approach (e.g., FedAvg) [28] aggregates model weights from all nodes iteratively until converging to a global model. However, such a single model learning paradigm suffers poor performance in practical HAR applications [35] due to the heterogeneity of different users' data. Recently, federated transfer learning [6] was proposed to improve the accuracy of the global FL model through personalizing the model on-device. However, there has been no theoretical analysis on how to balance local personalization and learning from others [42] for such post-training personalization FL method. Another method is *federated multi-task learning* [34], which optimizes multiple models simultaneously in a distributed setting. However, current federated multi-task learning approaches [43] only work for convex models such as support vector machine, which limits their applications in challenging HAR tasks such as image recognition.

In this paper, we propose ClusterFL, a clustering-based federated learning system that can achieve high system accuracy and low communication overhead. The design of ClusterFL is motivated by the key observation that, in spite of the heterogeneity, data distributions of different users' activities may share spatial-temporal similarity [36]. We then show that such similarity can be captured by cluster structures in a wide range of HAR applications, based on the analysis of two public and four new datasets consisting of data of total 184 users.

Motivated by this observation, we propose ClusterFL, a new federated learning system that exploits the clustering relationship of users' data to simultaneously improve the model accuracy and communication efficiency. First, ClusterFL features a novel *clustered multi-task federated learning* formulation by introducing a cluster indicator matrix indicating the similarity of users, which minimizes the empirical loss of learned models while automatically capturing the intrinsic cluster structure among the data of different users. To solve the formulated problem, we propose a new distributed optimization framework based on the Alternating Direction Method of Multipliers (ADMM) [5], which updates the model weights and the cluster structure alternatively until convergence. Moreover, to adapt the ADMM approach for the FL setting, we decompose the learning process into updates of local model weights to keep the data locality of nodes. Compared with previous personalized FL work, our learning framework enables collaborative learning among similar nodes and is applicable for general non-convex machine learning models (e.g., DNN and CNN models). Besides, through the convergence analysis on the proposed similarity-aware federated framework, we provide the guidance on how to choose hyper-parameters for achieving the convergence with general non-convex and strongly convex local models. Second, leveraging the learned cluster structure, ClusterFL integrates two new mechanisms, namely *cluster-wise straggler dropout* and *correlation-based node selection*, to reduce the communication overhead. Specifically, the server will drop two types of nodes in each cluster, including the *stragglers*[1] who converge slower and the nodes that are less related to others, which is demonstrated to reduce the overall communication overhead while maintaining the overall accuracy performance.

[1] *stragglers* in our paper refer to the nodes who converge slower than others, i.e., need more iterations to converge.

We evaluate the performance of ClusterFL on a hardware tesbed of 10 NVIDIA edge devices using four new datasets collected by ourselves, including a large-scale HAR dataset consisting of data of 121 subjects collected using a smartphone application in a crowdsourcing manner, and three in-lab small-scale HAR datasets of different applications. Our evaluation shows that, by capturing the cluster relationship in heterogeneous data of different user activities, ClusterFL outperforms several existing machine learning paradigms significantly in terms of system accuracy and the proposed communication optimization mechanisms can reduce more than 50% communication overhead. Moreover, the performance ClusterFL is robust under dynamic network conditions with unexpected disconnections between the server and nodes as well as various system settings.

Our key contributions can be summarized as follows:

- To understand the impact of clusterability that is central to ClusterFL, we analyze two public HAR datasets and four new datasets consisting of data of total 184 users. We find out that the clustering relationships are widely exhibited in users' HAR data and can be leveraged to improve the model accuracy of federated learning.
- We proposed ClusterFL, a clustering-based federated learning system, to achieve a high model accuracy by enabling collaborative learning among similar nodes. Compared with existing approaches, ClusterFL also helps reduce overall communication overhead through *cluster-wise straggler dropout* and *correlation-based node selection*, based on the learned cluster structure.
- We theoretically prove the convergence of the proposed clustering-based federated learning framework for non-convex and strongly convex local models. Moreover, we also provide the guidance on how to choose the hyper-parameters for achieving such convergence, which are in line with our experimental results on real-world datasets.
- We collect four new HAR datasets involving total 145 subjects with significant dynamics and conduct extensive experiments on our NVIDIA edge testbed using the four new datasets. We demonstrate the superior performance of ClusterFL compared with several state-of-the-art baselines under dynamic system configurations, different datasets and various system settings.

The rest of paper is organized as follows. Section 2 reviews the related work. Section 3 presents the motivation of our approach. Section 4-6 present the overview and design of ClusterFL. Section 7 introduces the new collected datasets and Section 8 presents the evaluation results. Section 9 discusses the future work and Section 10 concludes the paper.

2 RELATED WORK

Human Activity Recognition (HAR). Machine Learning has been increasingly adopted in the area of human activity recognition [36]. Various algorithms based on handcrafted features [3] and deep neural networks [18] have been developed to classify different human activities. Most work in this space is focused on the centralized approach that needs to train the algorithms at a central server, which imposes significant privacy concern due to the need to share raw user data. Recently, significant advances have been made on running deep learning models on mobile devices [23]. However, the labeled data on each end device is usually insufficient for training a good model.

Federated Learning (FL) [19] is a distributed machine learning approach that enables training on a large corpus of decentralized data residing on devices. Several existing FL approaches [19, 27, 28, 32] aim to learn a single model for all users by averaging the model weights. For example, Yogi [32] uses adaptive optimizers on the server to improve the convergence performance of federated learning. FedProx [27] adds a a regression term to the local subproblem of nodes to effectively limit the impact of changing local updates. However, the single learned model usually has limited

generality, making it poorly suited for heterogeneous user data in HAR applications. In ClusterFL, different nodes will train different personalized models during collaborative learning from similar nodes in federated learning. However, the single learned model usually has limited generality, making it poorly suited for heterogeneous user data in HAR applications [35]. To deal with this issue, federated transfer learning (FTL) is proposed [6, 9] to improve the accuracy of the global learning model by personalizing it for local data. However, there is a lack of theoretical analysis for such a post-training personalized FL approach to balance local personalization and learning from others [42]. Moreover, Smith et al. [34] propose to learn multiple models simultaneously under the FL settings. However, convergence is only guaranteed for convex models such as SVM in their method. ClusterFL, in contrast, is applicable to general non-convex models (e.g., DNN and CNN models). The guidance on selecting hyperparameters to achieve the convergence of ClusterFL is also provided for non-convex and strongly convex local models, respectively. For communication-effective FL, Oort [22] proposes to optimize the time-to-accuracy of FL training processing via specially designed participant selection during the aggregation between the central server and users. Such an approach can be incorporated with the node selection scheme in ClusterFL to improve system efficiency when the nodes train on heterogeneous devices. Other previous studies either choose stragglers (the nodes who converge slower) from all nodes [26] or focus on the quantization techniques [20], which are oblivious to the relationship between the data of different nodes. ClusterFL leverages the inherent cluster relationship learned in the FL settings to reduce the communication overhead while maintaining the overall accuracy performance.

HAR with Federated Learning. Federated learning has been recently applied to HAR to protect user's data privacy [6, 35, 37]. Specifically, Sozinov et al. [35] utilize FedAvg for HAR but achieve worse accuracy compared to centralized models. Chen et al. [6] propose a federated transfer learning framework for Parkinson's disease auxiliary diagnosis, which first performs FedAvg and then builds relatively personalized models on-device. Feng et al. [9] apply FL to human mobility prediction and propose a fine-tuned personal adaptor to improve the prediction performance. However, these post-training personalized FL methods are based on the FedAvg model without exploiting the relationships among nodes. Yu et al. [43] adopt federated multi-task learning to learn access control policies for smart home applications, which is only applicable for convex models such as SVM. Compared with existing FL work on HAR, ClusterFL exploits the similarity of users to not only improve accuracy of FL models, but also reduce the overall system overhead. Moreover, it can be applied to general deep learning models for various HAR applications.

3 MOTIVATION

In this section, we analyze several real-world HAR datasets to motivate the approach of ClusterFL. We first investigate the clusterability of real-world HAR datasets, including two public datasets and four new datasets collected by ourselves (see Section 7). Then we show the key advantages of learning among similar users based on the clusterability of their data.

3.1 Clusterability of HAR Data

Activities of different users often exhibit certain level of similarity, which may result from the subjects' biological features (e.g., gender, height, weight, etc.), the physical environment (e.g., where the subjects move about), or even sensor biases [36, 46]. We investigate data similarity of six real-word HAR datasets listed in Table 1, including a public smartphone-based human activity recognition (SHAR) dataset [1], a public heterogeneous HAR (HHAR) dataset [36]; and four new datasets (Depth, IMU, UWB, HARBox dataset) collected by ourselves (see Section 7).

Specifically, we show the clusterability of these HAR datasets using the Hopkins statistic [12] of the data from different users, which is a statistical metric between 0 and 1 and quantifies the

clustering tendency of data. A higher Hopkins statistic means stronger clusterability of the data. Table 1 shows the Hopkins statistic of each dataset which is calculated for data of same activity and then averaged across different activities. It shows that the Hopkins statistics of all six HAR datasets exceed 0.5, which means that they exhibit clustering relationships among different subjects' data. Particularly, the SHAR, HHAR, Depth and HARBox datasets have a stronger cluster tendency with the Hopkins statistics more than 0.7.

We further visualize the data distribution by plotting the data of "walking" in the HHAR dataset after reducing the dimension of features to 2D using Principal Component Analysis. As shown in Fig. 1, there exists a clear clustering relationship among different subjects' data, where the data from the same model of smartphone is grouped to the same cluster.

Dataset	SHAR	HHAR	Depth	IMU	UWB	HARBox
# of subjects/activities	30/6	9/6	9/3	7/2	8/3	121/5
Hopkins statistic	0.8813	0.7951	0.8699	0.6966	0.5742	0.8946

Table 1. Hopkins statistics of 6 different HAR datasets. A higher Hopkins statistic means stronger clusterability.

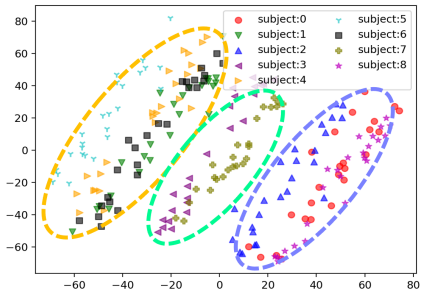


Fig. 1. The data of "walking" from the HHAR dataset after reducing dimension to 2D using PCA. There exists a clear cluster relationship among different subjects' data.

3.2 Impact of Clusterability on Learning

In this section, to motivate the approach of utilizing cluster relationship of users, we design a method referred to as "Centralized-cluster", where we cluster the subjects using K-means in a centralized manner based on their training data and then train a model for each cluster using data of all subjects who belong to the cluster. We compare the accuracy of "Centralized-cluster" with four typical machine learning paradigms: local learning, centralized single model learning ("Centralized-single"), federated average (FedAvg) and federated transfer learning (FTL) using the HHAR dataset. In local learning, each node trains a model using its own data, which may suffer overfitting due to limited local data. In centralized single model learning, the server collects data from all nodes and learns a single global model, which imposes a significant privacy concern by sharing the raw user data. FedAvg is a classical FL method proposed by Google [19], where the nodes only upload model weights to the server to generate one model by averaging their model weights. Federated transfer learning [6, 9] is a state-of-art FL approach that aims to improve the performance of FedAvg by personalizing the learned single FL model for different users based on their own data.

We evaluated the above methods using nine subjects' data from the HHAR dataset, which is collected using three models of smartphones. The task is to classify six kinds of human activities

using accelerometer and gyroscope data. Table 2 summarizes the mean accuracy using a 4-layer neural network.

Method	Local	FedAvg	FTL	Centralized-single	Centralized-cluster
Mean Accuracy (%)	53.67	33.61	54.61	72.22	**76.39**
STD (%)	9.74	12.812	10.19	6.06	**3.61**

Table 2. Accuracy comparison of different paradigms.

First, we observe that FedAvg fails to converge to the centralized model and performs even worse than local learning. It is not surprising since FedAvg is essentially a distributed approximation of centralized learning, which proves to suffer poor performance when nodes' data is heterogeneous [6, 35]. Moreover, although federated transfer learning (FTL) aims to customize different models for heterogeneous users, the accuracy improvement (0.94%) is still limited, as it does not explicitly consider the similarity of some nodes. The centralized single model learning performs better than the above three methods as it trains on the largest amount of data that is collected from all subjects. Finally, the "Centralized-cluster" method can achieve 76.39% mean accuracy, which performs even better than "Centralized-single". Although this method requires access to all the data and is not practical in distributed settings, it demonstrates the benefit of leveraging clustering relationships of nodes to improve accuracy.

The case study also suggests two main insights. First, if the cluster relationship of nodes could be captured in a distributed manner, it naturally entails a highly efficient FL paradigm in which only the nodes sharing data similarity will collaborate in learning, mitigating the impact of noise/outliers from other nodes. Another key advantage is that the cluster relationship provides opportunities for reducing communication overhead as the outliers who are less related to others can be dropped out in advance to avoid redundant communications during the distributed learning process.

4 APPROACH OVERVIEW

We now introduce ClusterFL, a practical federated learning system that aims to improve both model accuracy and communication efficiency for human activity recognition, using the intrinsic similarity among some nodes. We first briefly discuss the application scenarios of ClusterFL and then describe the system architecture.

Application Scenarios. ClusterFL is designed for a wide class of applications where user activities are tracked in a continuous and longitudinal manner. For example, in an Alzheimer's patient monitoring scenario [24], wearable and ambient sensors continuously track a patient's daily activities such as indoor/outdoor time, sleeping, etc, which are important digital biomarkers [10] for early Alzheimer's diagnosis. Other representative applications include fitness tracking [45], family daily routine monitoring [3] and social distancing detection [4]. In these applications, personal devices can accumulate data for a certain period of time and use it to train machine learning models for activity recognition. For each collaborative distributed training session in such scenarios, the cloud will communicate with ClusterFL on the devices to learn personalized local models. As the data distribution and characteristics of user activities may change over time, ClusterFL can run periodically (e.g., daily) to update the local models using recently accumulated data.

System Architecture. Firstly, ClusterFL features a novel clustering-based federated learning framework that minimizes the empirical training loss of learned models while automatically capturing the intrinsic cluster structure among the data of different nodes[2]. Specifically, we formulate

[2]In our context, a node refers to a device or a set of devices carried by the user, which runs a machine learning model to recognize the user's activities.

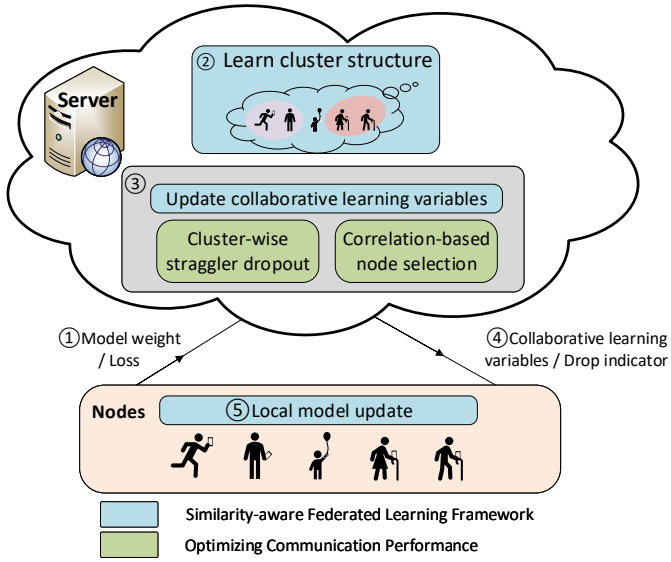


Fig. 2. System Architecture of ClusterFL

a new *clustered multi-task federated learning* problem by introducing a cluster indicator matrix which denotes the similarity of nodes and present a distributed solution to iteratively update nodes' model weights and the cluster indicator matrix using *alternating optimization* techniques. Through this framework, nodes in the same cluster will collaboratively improve performance by maximizing model correlations, and the server will be able to learn the cluster relationship among nodes in a small number of iterations. Second, based on the learned cluster structure, ClusterFL will utilize two new mechanisms, *cluster-wise straggler dropout* and *correlation-based node selection*, to reduce the communication overhead while maintaining local model accuracy. Specifically, the server will drop *stragglers* who converge slower than other nodes within each cluster to reduce the overall communication rounds. In addition, the server will drop nodes that are less related to others in the same cluster. In this case, the number of nodes interacting with the server will be further reduced so that the communication time for one round can be reduced while "more important" nodes are kept to perform ClusterFL.

Fig. 2 shows the overall system architecture of ClusterFL. Specifically, each communication round consists of the following steps: (1) The nodes will upload their current model updates and training loss to the server. (2) The server will quantify the relationship of model weights using a cluster indicator matrix through an optimization framework. As a result, the server can dynamically group the nodes into clusters with joint data distributions during the early phase of federated learning. (3) Based on the learned cluster indicator matrix, the server updates the collaborative learning variables which will be used by nodes to update their models, and drops stragglers that converge slower and the nodes that are less related to others within each cluster to reduce the communication overhead. (4) The collaborative learning variables and a drop indicator will be sent back to each node. (5) The nodes will update their models using the received collaborative learning variables, and decide whether to continue the learning process in the next round according to the drop indicator.

The above steps will run iteratively until convergence, i.e., the objective function of the *clustered multi-task federated learning* problem sees little changes. As discussed earlier, to adapt to dynamic

variations of user activities, this distributed training process can be repeated periodically (e.g., daily) using recently collected data.

5 CLUSTERING-BASED FEDERATED LEARNING FRAMEWORK

The design of ClusterFL is based on the key observation that data of many applications in human activity recognition exhibits inherent cluster relationships due to the subjects' biological features, the physical environment or even sensor biases, which can be leveraged to improve the overall model accuracy. Therefore, our goal is to capture the cluster relationship among nodes and aggregate model weights for nodes in the same cluster to improve the accuracy.

5.1 Problem Formulation

ClusterFL features a novel federated learning framework that maximizes the accuracy of learned models while automatically capturing the inherent similarity among the data of different nodes by introducing a cluster indicator matrix. Specifically, We formulate a clustered multi-task federated learning problem as follows:

$$\min_{\mathbf{W},\mathbf{F}} \sum_{i=1}^{M} \frac{1}{N_i} \sum_{r=1}^{N_i} l(\mathbf{w}_i^T \mathbf{x}_i^r, y_i^r) + \alpha tr(\mathbf{W}\mathbf{W}^T) - \beta tr(\mathbf{F}^T \mathbf{W}\mathbf{W}^T \mathbf{F}) \tag{1}$$

- M is the number of total involved nodes, N_i is number of training data samples in node i. α and β are the hyperparameters and $\alpha \geq \beta > 0$.
- $(\mathbf{x}_i^r, y_i^r) \in \mathbb{R}^D \times \mathbb{R}$ is the r-th training pair of i-th node; $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_M]^T \in \mathbb{R}^{M \times D}$ is the weight matrix to be estimated where each local model is an activity classifier; l is the loss function of local model.
- $\mathbf{F} \in \mathbb{R}^{M \times K}$ is an orthogonal cluster indicator matrix, with $F_{i,j} = \frac{1}{\sqrt{C_j}}$ if node i belongs to j-th cluster and $F_{i,j} = 0$ otherwise [47]. Here K is the number of clusters and C_j denotes the number of nodes in j-th cluster. We emphasize that we do not need to know K in our proposed optimization framework as the discrete constraints of $\mathbf{F}$ will be relaxed to obtain its continuous solutions in Section 5.3.

In the above formulation, the first term is the sum of empirical errors of activity recognition across all nodes; the second and third term can be rewritten as $(\alpha - \beta) \sum_{i=1}^{M} ||\mathbf{w}_i||_2^2 + \beta \sum_{j=1}^{K} \sum_{v \in \mathcal{S}_j} ||\mathbf{w}_v - \overline{\mathbf{w}_j}||_2^2$, which consist of the L2-norm regularization of model weights to prevent over-fitting and the K-means clustering to minimize the overall intra-cluster distances of model weights. Compared to FedAvg, which provides only one model for all nodes with heterogeneous data, our formulation will customize a model for each node while preserving the similarity of model weights of nodes in the same cluster. In this case, each node's model will be updated based on its local data and referring to other nodes' models, which will significantly improve individual performance by collaboratively learning within clusters.

Alternating optimization. Here we have two variables ($\mathbf{W}$ and $\mathbf{F}$) to be solved in problem (1) under federated learning setting. It is easy to see that (1) is not jointly convex w.r.t $\mathbf{W}$ and $\mathbf{F}$. Moreover, it would be very difficult to solve them simultaneously. To address this challenge, we propose to use the alternating optimization approach [2] to solve (1). In this case, we will fix $\mathbf{W}$ or $\mathbf{F}$ for each outer iteration of optimization and update the other variable, alternating between optimizations of these two variables until convergence. Algorithm 1 shows a centralized view of the alternating optimization between nodes and the server. In Section 5.2 and Section 5.3, we will present how to distributedly optimize nodes' model weights $\mathbf{W}$ and how to learn their cluster structure $\mathbf{F}$ in a federated setting, respectively.

Algorithm 1: Alternating Optimization of ClusterFL

Initialization: set $\mathbf{W} = \mathbf{0}^{M\times D}$, $\mathbf{F} = \mathbf{0}^{M\times M}$
for *h = 0* ***to*** H **do**
  1. Optimization of $\mathbf{W}$ with $\mathbf{F}$ fixed.
  for *t = 0* ***to*** T **do**
    node update: parallelly update $\mathbf{w}_i$(i = 1,...,M)
      $\mathbf{w}_i^{t+1} \leftarrow$ **Local SGD**$(\mathbf{w}_i^t, (\mathbf{x}_i, \mathbf{y}_i), \lambda_i, \mathbf{z}_i^t)$
    server update: update $(\lambda_i, \mathbf{z}_i^t)$ for all nodes.
      a) Update the auxiliary variables $\Omega, \mathbf{U}$.
      b) Update the collaborative variables $(\lambda_i, \mathbf{z}_i^t)$ using $\Omega, \mathbf{U}, \mathbf{W}, \mathbf{F}$, for node i=1, ..., M.
  end
  2. Optimization of $\mathbf{F}$ with $\mathbf{W}$ fixed.
    server update: Update $\mathbf{F}$ based on Section 5.3.
end

5.2 Optimize Model Weights

When $\mathbf{F}$ is fixed, problem (1) w.r.t $\mathbf{W}$ is equivalent to:

$$\min_{\mathbf{W}} \sum_{i=1}^{M} \frac{1}{N_i} \sum_{r=1}^{N_i} l(\mathbf{w}_i^T \mathbf{x}_i^r, y_i^r) + \alpha tr(\mathbf{W}\mathbf{W}^T) - \beta tr(\mathbf{F}^T \mathbf{W}\mathbf{W}^T \mathbf{F}) \tag{2}$$

Here, we propose to use the Alternating Direction Method of Multipliers (ADMM) approach [5] to update $\mathbf{w}_i$(i = 1, ..., M) distributedly across nodes without sharing information of data sample. The idea of ADMM is to fix two variables in the augmented Lagrangian $\mathbf{L}_\rho$ and update the remaining variable, which will run in an alternating or sequential fashion. We note that although ADMM is widely used for distributed optimization of statistical learning problems, applying it to optimize the model weights $\mathbf{W}$ in a federated setting is not trivial.

In particular, we need first to formulate a new problem where two decision variables are subject to linear constraints, to be consistent with the standard ADMM formulation. Therefore, we define $\Omega = \mathbf{F}^T\mathbf{W} \in \mathbb{R}^{K\times D}$ and reformulate problem (2) as follows:

$$\min_{\mathbf{W},\Omega} f(\mathbf{W}) + g(\Omega)$$

$$s.t.\ \mathbf{F}^T\mathbf{W} - \Omega = 0$$

$$\text{where: } f(\mathbf{W}) = \sum_{i=1}^{M} \frac{1}{N_i} \sum_{r=1}^{N_i} l(\mathbf{w}_i^T \mathbf{x}_i^r, y_i^r) + \alpha tr(\mathbf{W}\mathbf{W}^T), \quad g(\Omega) = -\beta tr(\Omega\Omega^T). \tag{3}$$

Moreover, in the federated learning setting, the empirical loss function embedded in $f_i(\mathbf{w}_i)$ needs to be calculated locally according to different nodes' data $(\mathbf{x}_i, \mathbf{y}_i)$. Therefore, we have to decompose the minimization step of $\mathbf{W}$ to a combination of updates of local model weight $\mathbf{w}_i$(i = 1, ..., M), so as to keep the locality of data. Then we obtain the augmented Lagrangian of the variable splitting ADMM formulation as follows:

$$L_\rho(\mathbf{W}, \Omega, \mathbf{U}) = \sum_{i=1}^{M} f_i(\mathbf{w}_i) + \sum_{j=1}^{K} g(\Omega_j) + \sum_{j=1}^{K} (\mathbf{F}_j^T\mathbf{W} - \Omega_j) \cdot \mathbf{U}_j^T + \sum_{j=1}^{K} \frac{\rho}{2} ||\mathbf{F}_j^T\mathbf{W} - \Omega_j||_2^2, \tag{4}$$

where $\mathbf{U}_j \in \mathbb{R}^{1\times D} (j = 1, ..., K)$ is the dual variable and ρ is the penalty parameter introduced in the augmented Lagrangian.

According to the above augmented Lagrangian, we obtain the following centralized ADMM update at iteration t+1:

$$\mathbf{W}^{t+1} = \arg\min_{\mathbf{W}}(\sum_{i=1}^{M} f_i(\mathbf{w}_i) + \sum_{j=1}^{K}(\mathbf{F}_j^T\mathbf{W})\cdot(\mathbf{U}_j^t)^T + \frac{\rho}{2}\sum_{j=1}^{K}||\mathbf{F}_j^T\mathbf{W} - \Omega_j||_2^2)$$

$$\Omega_j^{t+1} = \arg\min_{\Omega_j}(g(\Omega_j) - \Omega_j\cdot(\mathbf{U}_j^t)^T + \frac{\rho}{2}||\Omega_j - \mathbf{F}_j^T\mathbf{W}^{t+1}||_2^2) \quad \text{j = 1, ..., K}$$

$$\mathbf{U}_j^{t+1} = \mathbf{U}_j^t + \rho(\mathbf{F}_j^T\mathbf{W}^{t+1} - \Omega_j^{t+1}) \quad \text{j = 1, ..., K}$$

The Ω_j (j = 1, ..., K) update is easy to have a closed form solution. When setting the derivative of the objective function equals to 0, we have:

$$\Omega_j^{t+1} = \frac{\rho(\mathbf{F}_j^T\mathbf{W}^{t+1}) + \mathbf{U}_j^t}{\rho - 2\beta} \quad (\rho \neq 2\beta)$$

Distributed Update of W. As the model weight of each node ($\mathbf{w}_i$, i=1,...,M) needs to be updated on device to keep the locality of data under the federated learning settings, we then decompose the update of $\mathbf{W}$ to the summation of $\mathbf{w}_i$ (i=1,...,M). The $\mathbf{W}$ update can be written as:

$$\begin{aligned}
\mathbf{W}^{t+1} &= \arg\min_{\mathbf{W}}(\sum_{i=1}^{M} f_i(\mathbf{w}_i) + \sum_{j=1}^{K}(\mathbf{F}_j^T\mathbf{W})\cdot(\mathbf{U}_j^t)^T + \frac{\rho}{2}\sum_{j=1}^{K}||\mathbf{F}_j^T\mathbf{W} - \Omega_j||_2^2) \\
&\iff \arg\min_{\mathbf{W}}(\sum_{i=1}^{M} f_i(\mathbf{w}_i) + \frac{\rho}{2}\sum_{j=1}^{K}||\mathbf{F}_j^T\mathbf{W}||_2^2 + \sum_{j=1}^{K}(\mathbf{F}_j^T\mathbf{W})\cdot(\mathbf{U}_j^t - \rho\Omega_j)^T) \\
&\iff \arg\min_{\mathbf{W}}(\sum_{i=1}^{M} f_i(\mathbf{w}_i) + \frac{\rho}{2}\sum_{j=1}^{K}||\sum_{i=1}^{M}F_{ij}\mathbf{w}_i||_2^2 + \sum_{j=1}^{K}(\sum_{i=1}^{M}F_{ij}\mathbf{w}_i)\cdot(\mathbf{U}_j^t - \rho\Omega_j)^T) \qquad (5)
\end{aligned}$$

Then the $\mathbf{W}$ update can be distributed to the update of $\mathbf{w}_i$ (i=1,...,M):

$$\mathbf{w}_i = arg\min_{\mathbf{w}_i}(f_i(\mathbf{w}_i) + \frac{\rho}{2}\sum_{j=1}^{K}F_{ij}^2||\mathbf{w}_i||_2^2 + \mathbf{w}_i\cdot(\rho\sum_{j=1}^{K}F_{ij}\sum_{q=1,q\neq i}^{M}F_{iq}\mathbf{w}_q + \sum_{j=1}^{K}F_{ij}(\mathbf{U}_j^t - \rho\Omega_j)^T))$$

Or equivalently write as:

$$\mathbf{w}_i^{t+1} = \arg\min_{\mathbf{w}_i}(f_i(\mathbf{w}_i) + \lambda_i||\mathbf{w}_i - \frac{1}{2\lambda_i}\mathbf{z}_i^t||_2^2)$$

$$\text{where: } \lambda_i = \frac{\rho}{2}\sum_{j=1}^{K}F_{ij}^2, \mathbf{z}_i^t = \sum_{j=1}^{K}F_{ij}(\rho\Omega_j^t - \mathbf{U}_j^t - \rho\sum_{q=1,q\neq i}^{M}F_{iq}\mathbf{w}_q^t)^T$$

Finally, an iteration t+1 of federated ADMM update consists of the following steps:

- **Node Update** (line 4-5 in Algorithm 1): Each node will parallelly optimize (e.g., using gradient descent methods) its model weight $\mathbf{w}_i$ based on its local data $(\mathbf{x}_i, \mathbf{y}_i)$ and the collaborative learning variables $(\lambda_i, \mathbf{z}_i^t) \in \mathbb{R}^{D+1}$ from the server.

$$\mathbf{w}_i^{t+1} = \arg\min_{\mathbf{w}_i}(f_i(\mathbf{w}_i, \mathbf{x}_i, \mathbf{y}_i) + \lambda_i||\mathbf{w}_i - \frac{1}{2\lambda_i}\mathbf{z}_i^t||_2^2) \qquad (6)$$

- **Server Update** (line 6-8 in Algorithm 1): The server will further utilize the newly-updated model weights from nodes $\mathbf{W}$ and the cluster structure $\mathbf{F}$ (optimized in Section 5.3) to update:

a) the auxiliary variables $\mathbf{\Omega}, \mathbf{U}$ introduced in ADMM, for j=1,...,K:

$$\mathbf{\Omega}_j^{t+1} = \frac{\rho(\mathbf{F}_j^T\mathbf{W}^{t+1}) + \mathbf{U}_j^t}{\rho - 2\beta} \quad (\rho \neq 2\beta)$$
$$\mathbf{U}_j^{t+1} = \mathbf{U}_j^t + \rho(\mathbf{F}_j^T\mathbf{W}^{t+1} - \mathbf{\Omega}_j^{t+1})$$

b) the collaborative learning variables $(\lambda_i, \mathbf{z}_i^t)$ transmitted to each node, for i=1,...,M:

$$\lambda_i = \frac{\rho}{2}\sum_{j=1}^{K} F_{ij}^2, \quad \mathbf{z}_i^t = \sum_{j=1}^{K} F_{ij}(\rho\mathbf{\Omega}_j^t - \mathbf{U}_j^t - \rho\sum_{q=1,q\neq i}^{M} F_{iq})^T$$

Summary of Federated ADMM update. For the distributed ADMM update in the federated learning settings, instead of directly transferring $\mathbf{\Omega} \in \mathbb{R}^{M\times D}$ and $\mathbf{U} \in \mathbb{R}^{M\times D}$ that are high-dimensional, the server will calculate $(\lambda_i, \mathbf{z}_i^t) \in \mathbb{R}^{D+1}$ for each node (i = 1, ..., M) using the updated $\mathbf{\Omega}$ and $\mathbf{U}$ and send back to the nodes. Therefore, $(\lambda_i, \mathbf{z}_i^t)$ can be regraded as the collaborative learning variables sent to node i and $(\mathbf{\Omega}, \mathbf{U})$ can be seen as the intermediate auxiliary variables introduced in the ADMM update.

Therefore, in a communication round between nodes and the server, the nodes need to upload their updated model weights $\mathbf{w}_i$ to the server, and the server will send the collaborative learning variables $(\lambda_i, \mathbf{z}_i^t) \in \mathbb{R}^{D+1}$ to node i after aggregating the weights. Referring to the expression of $\mathbf{\Omega}$, $\mathbf{U}$ and problem formulation in (1), here $\mathbf{z}_i^t \in \mathbb{R}^D$ is a combination of model weights $(\mathbf{w}_1, ..., \mathbf{w}_M)$ and serves as the weight center of the cluster that node i belongs to. Therefore, for the local model update of node i, it will minimize the empirical loss on local data and the distance between its model weight and the cluster center, so as to learn from similar nodes in the federated learning system. It's noted that, the communication overhead between the server and nodes in each iteration of ClusterFL is almost the same as Fedavg ($\mathbf{w}_i^t \in \mathbb{R}^D$). The details for the derivation of the above federated ADMM update are given in Appendix **??**.

5.3 Learn Cluster Structure

When $\mathbf{W}$ is fixed, problem (1) w.r.t $\mathbf{F}$ can be seen as a K-means clustering problem (in a matrix representation) on the nodes' model weights $\mathbf{W}$. There are two challenges when learning the cluster relationship of nodes using their model weights at the server: how to quantify the similarity of model weights and how to optimize cluster structure dynamically without knowing the number of clusters K.

Similarity of model weights. As demonstrated in previous work [11], distance-based clustering methods have severe limitation in modeling the similarity of machine learning models since they are only applicable to models with convex loss functions. Therefore, for general DNN models with non-convex loss functions, we choose to use the Kullback–Leibler divergence (KLD) to measure the similarity of nodes' models. KLD [21] is used to measure how one probability distribution is different from the other and is widely used in knowledge distillation [15], model adaptation [42] and similarity measurement[17]. The KLD of two DNN models $(\mathbf{w}_i, \mathbf{w}_j)$ can be expressed as:

$$D_{KL}(\mathbf{w}_i, \mathbf{w}_j) = \frac{1}{N_O}\sum_{r=1}^{N_O} \delta(\mathbf{w}_i, \mathbf{x}_o^r) log\frac{\delta(\mathbf{w}_i, \mathbf{x}_o^r)}{\delta(\mathbf{w}_j, \mathbf{x}_o^r)}$$
$$\delta(\mathbf{w}_i, \mathbf{x}_o^r) = softmax(\frac{\Phi(\mathbf{w}_i, \mathbf{x}_o^r)}{\tau})$$

where $\Phi(\mathbf{w}_i, \mathbf{x}_o^r)$ denotes the pre-softmax output of model $\mathbf{w}_i$ on the input data $\mathbf{x}_o^r$. Smaller KL divergence denotes closer relationship of models for node i and j. Then we calculate the KL

divergence between any two models and therefore obtain a similarity matrix $\mathbf{D} \in \mathbb{R}^{M\times M}$, where $\mathbf{D}_{i,j} = D_{KL}(\mathbf{w}_i, \mathbf{w}_j)$ measures the similarity between the i_{th} and j_{th} node.

Optimize cluster relationship. Next, the server will learn cluster indicator matrix $\mathbf{F}$ with the model similarity matrix $\mathbf{D} \in \mathbb{R}^{M\times M}$ of nodes. As the number of clusters among nodes (i.e., K) is typically not available at the server in federated learning, in this paper, we use spectral clustering [38, 44] to obtain an approximation of the cluster relationship $\mathbf{P} \in \mathbb{R}^{M\times M}$ using the similarity matrix of nodes $\mathbf{D}$. Then we post-process $\mathbf{P}$ to satisfy the constraints of $\mathbf{F} \in \mathbb{R}^{M\times M}$ defined in the problem formulation in Section 5.1, so that we can solve the problem without knowing the number of clusters K. Therefore, the dimension M of the relaxed cluster indicator matrix $\mathbf{F}$, has lost the physical meaning of cluster number after spectral relaxation [44]. It means that M is no longer the cluster number a prior, but rather a subspace dimension describing the cluster structure, and is commonly treated as a hyperparameter.

According to [8], principal components of the similarity matrix are the continuous solutions to the discrete cluster membership indicators for K-means clustering. Therefore, $\mathbf{P} = \mathbf{Q}_{M-1}\mathbf{Q}_{M-1}^T \in \mathbb{R}^{M\times M}$ will be the continuous solution of $\mathbf{F}$, where $\mathbf{Q}_{M-1} = (\mathbf{v}_1, ..., \mathbf{v}_{M-1}) \in \mathbb{R}^{M\times(M-1)}$ collects the $M-1$ principal components of $\mathbf{D}$ using principal components analysis (PCA).

If the data of nodes has a cluster structure, $\mathbf{P}$ is expected to yield a similar diagonal block structure after permutation clusters together. However, as $\mathbf{P}$ is an approximation of the discrete valued indicators, there may be outliers in $\mathbf{P}$. We recover $\mathbf{F}$ more accurately as follows: 1) we set $P_{ij} = 0$ if $P_{ij} < 0$ as $\mathbf{P}$ could contain negative elements; 2) we scale each element of $\mathbf{P}$ as $F_{ij} = \frac{P_{ij}}{\sqrt{\sum_{i=1}^{M} P_{ij}}}$ to obtain the final normalized cluster indicator matrix $\mathbf{F}$. Note that here we obtain a continuous approximation of $\mathbf{F}$ for the optimization steps in line 12-13 of Algorithm 1 without requiring to know the number of cluster K in the optimiztion process.

5.4 Convergence Analysis

Referring to Algorithm 1, we use an alternating optimization process for updating $\mathbf{W}$ and $\mathbf{F}$, which is guaranteed to converge to a stationary point [2] of problem (1) if $\mathbf{W}$ is centrally optimized. In this section, we will focus on the convergence of ADMM update (including $\mathbf{W}$, $\mathbf{\Omega}$, $\mathbf{U}$) in the federated learning setting. First, we prove the convergence of federated ADMM update when the nodes train general non-convex models (deep learning models such as CNN and DNN) with proper choices of the penalty parameter ρ. Moreover, if the nodes' local models are strongly convex (linear models such as linear regression and SVM), better properties of the convergence can be given, including a more concrete and practical guidance for the parameter chosen to achieve a linear convergence rate of update. We will present the key ideas of convergence analysis and several important results in the paper. The details of convergence proof can be found in Appendix A and Appendix B.

5.4.1 Convergence for non-convex models. In this subsection, we provide the convergence analysis of federated ADMM update when the local models of nodes are non-convex (general deep learning models such as CNN and DNN), under the standard assumptions of the objective function (Assumption 1 and 2) and proper choices of the penalty parameters ρ (Assumption 3).

Assumption 1 (Lipschitz Continous Gradient). *For the loss function f_i of each node i (i=1,...,M), there exists a positive constant $L_{f_i} > 0$ such that: $||\nabla f_i(\mathbf{u}) - \nabla f_i(\mathbf{v})||_2^2 \leq L_{f_i}||\mathbf{u} - \mathbf{v}||_2^2$, for $\forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^D$.*

Assumption 2 (Bounded from Below). *The objective value of problem (2), $P(\mathbf{W})$, is bounded below over the set $\mathcal{W}$, that is: $\underline{P} := \min_{\mathbf{W}\in\mathcal{W}} P(\mathbf{W}) > -\infty$.*

Assumption 3 (Proper Parameter Chosen). *The penalty parameter of ADMM update ρ is chosen large enough such that:*

- *The subproblem of* $\mathbf{w}_i$ *update in (6) (*$\mathbf{w}_i = \arg\min_{\mathbf{w}} P_i(\mathbf{w}), \forall i = 1, ..., M$*) is strongly convex with modulus* $\gamma_i(\rho)$*, i.e.,* $P_i(\mathbf{u}) - P_i(\mathbf{v}) \geq P_i(\mathbf{v})(\mathbf{u} - \mathbf{v}) + \frac{\gamma_i(\rho)}{2}||\mathbf{u} - \mathbf{v}||_2^2$.
- $\rho\gamma(\rho) > \frac{2L_f}{\sigma_{max}^2}$*, where* $\gamma(\rho) = \max_i \gamma_i(\rho)$*,* $L_f = \max_i L_{f_i} (i = 1, ..., M)$*,* σ_{max} *is the largest singular value of* $\mathbf{F}$.

Remark 1. *On the one hand, in Assumption 3, as* ρ *increases, the subproblem of* $\mathbf{w}_i$ *update (*$\forall i = 1, ..., M$*) will be eventually strongly convex with respect to* $\mathbf{w}_i$ *[16]. Moreover, the corresponding strong convexity modulus* $\gamma_i(\rho)$ *is a monotonic increasing function of* ρ*. On the other hand, note that* ρ *acts like the learning rate hyper-parameter in the update of* $\mathbf{U}$ *(see Section 5.2), a large* ρ *enlarges the fluctuations of* $\mathbf{U}$ *and slows down its convergence. Therefore, the penalty parameter* ρ *should not be set too large once Assumption 3 is satisfied and the convergence is achieved.*

When the local models of nodes is lipschitz continous (Assumption 1) and bounded from below (Assumption 2), and if ρ is chosen large enough such that the subproblem of $\mathbf{w}_i$ update is strongly convex (Assumption 3), Theorem 1 shows that Algorithm 1 will converge to at least a stationary point satisfying the KKT conditions.

Theorem 1. *When Assumption 1,2,3 are satisfied, we have the following:*

- *The ADMM update converges:*

$$\lim_{t\to\infty} ||\mathbf{w}_i^{t+1} - \mathbf{w}_i^t||_2^2 = 0, \forall i = 1, ..., M \tag{4a}$$

$$\lim_{t\to\infty} ||\mathbf{U}^{t+1} - \mathbf{U}^t||_2^2 = 0 \tag{4b}$$

$$\lim_{t\to\infty} ||\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1}||_2^2 = 0 \tag{4c}$$

- *Let* $\{\mathbf{W}^*, \Omega^*, \mathbf{U}^*\}$ *denote any limit point of* $\{\mathbf{W}^{t+1}, \Omega^{t+1}, \mathbf{U}^{t+1}\}$*, then* $\{\mathbf{W}^*, \Omega^*, \mathbf{U}^*\}$ *is a stationary solution of Problem (1) and satisfies the KKT conditions:*

$$\nabla f(\mathbf{W}^*) + \mathbf{F}\mathbf{U}^* = 0 \tag{5a}$$

$$\nabla g(\Omega^*) - \mathbf{U}^* = 0 \tag{5b}$$

$$\mathbf{F}^T\mathbf{W}^* - \Omega^* = \mathbf{0} \tag{5c}$$

- *When the set* $\mathcal{W}$ *for* $\mathbf{W}$ *is compact, the sequence of iterates generated by Algorithm 1 converges to stationary points.*

where the first term shows the convergence of the variables in ADMM update; the second term means that any limit point of Problem (1) is a stationary point that satisfies the KKT conditions, which is proved based on the first term and the optimal condition of $\mathbf{W}$ update and Ω update; for the third term, if the set of $\mathcal{W}$ is compact, $\mathbf{W}$ will converge to at least one limit point of the sequence (the convergence in the first term is not necessary to a feasible solution), then based on the second term , the sequence $\{\mathbf{W}^{t+1}, \Omega^{t+1}, \mathbf{U}^{t+1}\}$ generated by Algorithm 1 will converge to the set of stationary points satisfying KKT conditions. The detail proof of Theorem 1 can be found in Appendix A.

5.4.2 Linear convergence rate for strongly convex models. To avoid overfitting on the limited training data or save energy consumptions on embedded or mobile devices, the nodes may also adopt linear models with strongly convex loss functions, e.g., linear regression and support vector machines. If we further assume the strongly convexity of local models (Assumption 4), we can provide a more concrete guidance for the choices of ρ to achieve a linear convergence rate of update, as shown in Theorem 2.

Assumption 4 (Strongly Convexity). *The function f_i for each node i (i = 1, 2, ..., M) is strongly convex with parameter m_{f_i} ($m_{f_i} > 0$). That is, for $\forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^D$, we have: $\langle \nabla f_i(\mathbf{u}) - \nabla f_i(\mathbf{v}), \mathbf{u} - \mathbf{v} \rangle \geq m_{f_i} ||\mathbf{u} - \mathbf{v}||_2^2$.*

Definition 1 (Q-linearly Convergence). *We say that a sequence $\mathbf{x}_t$, where the superscript t stands for time index, Q-linearly converges to a point $\mathbf{x}^*$ if there exists a number $q \in (0, 1)$ such that $lim_{t\to\infty} \frac{||\mathbf{x}_{t+1} - \mathbf{x}^*||}{||\mathbf{x}_t - \mathbf{x}^*||} = q$ with $||\cdot||$ being a vector norm [7].*

Definition 2. *Define the matrix $\mathbf{V} = (\Omega, \mathbf{U})$, $\mathbf{G} = \begin{pmatrix} \rho \mathbf{I}_{KD} & \mathbf{0} \\ \mathbf{0} & \frac{1}{\rho}\mathbf{I}_{KD} \end{pmatrix}$, then the G-norm of $\mathbf{V}$ will be $||\mathbf{V}||_{\mathbf{G}}^2 = \mathbf{V}^T \mathbf{G} \mathbf{V} = \rho ||\Omega||_2^2 + \frac{1}{\rho} ||\mathbf{U}||_2^2$.*

Theorem 2. *When Assumption 4 is satisfied, the matrix $\mathbf{V}^t = (\Omega^t, \mathbf{U}^t)$ is Q-linearly convergent to its optimal $\mathbf{V}^* = (\Omega^*, \mathbf{U}^*)$ with respect to the G-norm:*

$$||\mathbf{V}^{t+1} - \mathbf{V}^*||_{\mathbf{G}}^2 \leq \frac{1}{1+\delta} ||\mathbf{V}^t - \mathbf{V}^*||_{\mathbf{G}}^2 \tag{9}$$

$$and \quad \delta = min\{\frac{\rho^2 - 4\beta\rho}{4\beta^2}, \frac{\rho(\frac{2m_f}{\sigma_{max}^2} - 4\beta)}{\rho^2 - 4\beta^2}\} > 0,$$

where σ_{max} is the largest singular value of the cluster indicator matrix $\mathbf{F}$.

Theorem 2 shows that when the nodes train strong convex models (Assumption 4), $\rho > 4\beta$ and $\beta < \frac{m_f}{2\sigma_{max}^2}$ is needed to achieve a Q-linear convergence of the ADMM update. The detail proof of Theorem 2 can be found in Appendix B.

6 OPTIMIZING COMMUNICATION PERFORMANCE

In traditional FL systems, a large number of update iterations between nodes and the server are required, which makes communication overhead the bottleneck of the learning process[28]. To make the FL system more communication-effective, some previous work focuses on the model compression and quantization techniques [20], which does not always reduce the total communication delay when the number of communication rounds is large. Other approaches chooses *stragglers* (the nodes who converge slower) from all nodes and drop them simultaneously [26], which does not consider the discrepancy among the stragglers. In this work, we aim to leverage the inherent cluster relationship learned by ClusterFL to dynamically drop nodes during the FL process, while maintaining the overall accuracy performance.

Our key idea here is that the server can utilize the learned cluster structure to drop some nodes for reducing the communication delay. In particular, the cluster structure in our framework will usually be learned early, e.g., in several communication rounds. Fig. 3 shows the update of the cluster indicator matrix $\mathbf{F}$, where we use its L1-norm to show how $\mathbf{F}$ changes over the number of times that $\mathbf{F}$ is updated. Here we use the depth images dataset we collected from three environments (see Section 7) and the task is to recognize five gestures. We can see that $\mathbf{F}$ remains almost unchanged after updated for 8 times even though the ClusterFL updates $\mathbf{F}$ 25 times in total. It clearly shows that, through $\mathbf{F}$ in the 8th update, the server already captures the cluster relationships among nodes and thus is able to use this information to optimize communication performance.

Motivated by this key observation, we propose two novel mechanisms to intelligently drop two types of nodes during the federated learning process. First, using the clustering relationship and loss update of nodes, the server will identify *stragglers* within clusters and drop them early, which is shown to reduce more communication overhead without hurting the accuracy of learned models. Second, to further reduce the communication cost of nodes for a large-scale FL system, for each

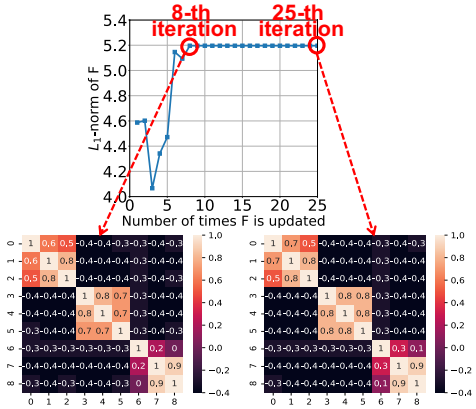
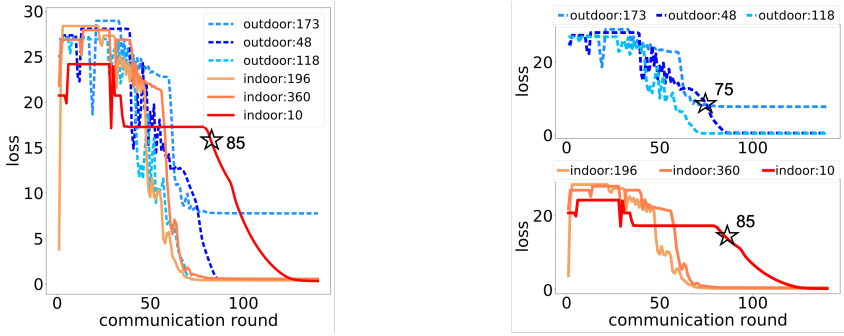


Fig. 3. The cluster structure of nodes (**F**) can be captured after updated several times.

node, the server will compute its mean correlation with other nodes in the same cluster and drop nodes whose data is less related to others.

6.1 Cluster-wise Straggler Dropout

We first use an example to illustrate the key advantage of dropping stragglers within clusters. Fig. 4 shows the loss update of six nodes from the environment “outdoor" and “indoor" of the depth dataset. As shown in Fig. 4a, only one node in the cluster “indoor" is dropped if the server identifies stragglers from all nodes. However, our *cluster-wise straggler dropout* in Fig. 4b can dynamically identify stragglers within each cluster (i.e., drop one node from the cluster “outdoor" at round 75 and another from the cluster “indoor" at round 85). Compared with identifying stragglers among all nodes, our *cluster-wise straggler droppout* can reduce more communication overhead (i.e., 122.9s) of transferring model weights while maintaining the accuracy performance.



(a) Among all nodes. Communication time: 842.5s; Accuracy: 73.01

(b) Within clusters. Communication time: 719.6s; Accuracy: 72.12

Fig. 4. Identifying stragglers among all nodes (a) and within clusters (b). Our cluster-wise straggler dropout can dynamically identify and drop stragglers within each cluster, thus reducing more communication overhead (i.e., 122.9s) while maintaining the accuracy performance. Here “indoor:196" means the node has 196 data samples from the environment “indoor".

Next, we discuss how to measure the convergence performance of nodes to identify stragglers for a specific cluster. For cluster j (j = 1,...,K) which contains C_j nodes, we define the averaged

convergence rate γ_q for node q (q = 1,...,C_j) in the cluster during the latest T_c iterations as follows:

$$\gamma_q = \frac{1}{T_c}\sum_{t=1}^{T_c}\varepsilon_q^t \quad for\ q = 1, 2, ..., C_j$$

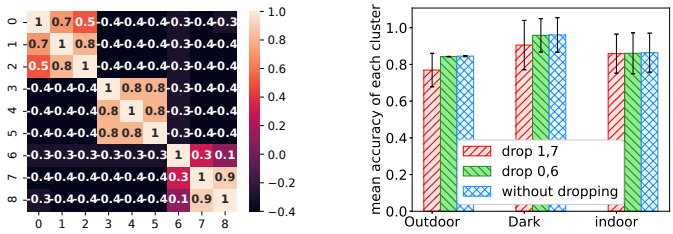
$$\varepsilon_q^t = \frac{|loss_q(t) - loss_q(t-1)|}{\sum_{q=1}^{C_j}|loss_q(t) - loss_q(t-1)|}$$

Here ε_q^t is the normalized loss update for node q compared to other nodes in cluster j at t-th iteration; γ_q is the mean value of ε_q^t in the latest T_c iterations; T_c is the threshold of iteration for determining stragglers, where a larger T_c means obtaining a more smooth loss update to filter out interferences while also putting off the iteration to recognize the stragglers.

If a node is dropped by the server, it will stop communicating with the server, but train locally on its own data and the last updated model until local convergence. As the remaining nodes have nearly converged, their model updates will not be affected substantially by dropping the stragglers.

6.2 Correlation-based Node Selection

In this section, we propose a new approach based on the observation that if a subset of nodes in a cluster are more correlated with each other, they will benefit more through collaborative learning. As Fig. 5a shows, node 0 in cluster “outdoor" and node 6 in cluster “indoor" have weaker correlations with other nodes in their clusters. If we drop them in the intermediate phase, as Fig. 5b shows, the mean accuracy of nodes in the same cluster only suffers minor degradation. However, if we choose to drop node 1 and node 7 at the same iteration, the mean accuracy will drop substantially from 89.00% to 84.44%. Therefore, the server will choose to drop nodes based on their correlations with other nodes in the cluster.



(a) Correlation matrix of nodes (b) Accuracy of each cluster

Fig. 5. Effectiveness of correlation-based dropout. Left: The three block denotes the cluster “outdoor”, “dark”, “indoor”, respectively. Compared to node 1 and node 7, node 0 and node 6 are less related to other nodes in the clusters. Right: The accuracy performance after dropping the less related nodes suffers minor degradation.

Specifically, the server will compute an importance vector σ for all nodes using the correlation matrix of $\mathbf{F}$ to measure their degree of correlation with other nodes in each cluster. Suppose node q is in the cluster j which contains C_j nodes, we define the importance metric σ_q for node q as follows:

$$\sigma_q = \frac{1}{C_j}\sum_{p=1}^{C_j}\mathbf{R}_{pq} \quad \text{for node } q \text{ in cluster } j.$$

As shown in Fig. 5, a node with a larger σ_q (e.g., node 1 with $\sigma_1 = 0.833$ while node 0 with $\sigma_0 = 0.733$) means it has a higher level of correlation with other nodes in the same cluster. The server will then order σ_q of all clusters from large to small and drop the last M_d nodes at iteration T_{thresh}. In this way, the total number of nodes communicating with the server will be reduced to $M - M_d$.

Application	Task	Data Dimension	Number of Subjects	Number of Data Records	Sensor
Human Movement Detection using UWB	with/without Human Movement	55	8	663	Decawave DWM1000 UWB
Walking Activity Recognition using IMU	walking on corridors/ upstairs/downstairs	900	7	1369	LPMS-B2 IMU
Gesture Recognition using Depth Camera	good/ok/ victory/stop/fist	1296	9	7422	PicoZense DCAM710
HARBox: ADL Recognition using Smartphones	walking/hopping/ phone calls/waving/typing	900	121	32935	77 different smartphone models

Table 3. Four new HAR datasets (UWB, IMU, Depth, HARBox) collected in real-world experiments

The dropped-out nodes will then locally train their models based on the last update and do not communicate with the server. Finally, the total communication time will be reduced since the number of nodes interacting with the server is smaller, while relatively more important nodes are kept in the learning process to improve the overall performance.

Here both the number of dropped nodes M_d and the dropping iteration T_{thresh} will be decided based on the desired tradeoff between overall accuracy and communication performance. The smaller M_d means fewer nodes to be dropped out, resulting in less communication reduction and accuracy loss. The smaller T_{thresh} means dropping nodes earlier, resulting in more communication reduction and accuracy loss.

7 DATASETS AND PREPROCESSING

We collect four new human activity datasets (Table 3) in real-world settings. The first dataset is a large-scale dataset collected using an Android App in a crowdsourcing manner. The other three are collected in indoor environments.[3]

The reasons we use self-collected datasets are as follows. First, most of existing HAR datasets lack the subject ID and mix all subjects' data, which is not suitable for the FL settings. Second, current HAR datasets are mainly collected in controlled (sometimes the same) environments with little dynamics, which is not consistent with the real-world applications where the users, devices, and environments are highly diverse. To address this issue, we collected four new HAR datasets in different environments with significant dynamics. Finally, there is currently no large-scale public HAR dataset collected in real-world settings. Using a new smartphone App, we collect data of total 121 users with 77 different models of smartphones. We will release this large-scale high-quality dataset to the research community.

Large-scale HARBox Dataset: Smartphones are increasingly used to monitor people's daily activities or health conditions [14]. We developed and released an Android App named "HARBox" to collect HAR data using users' own smartphones in a crowdsourcing manner. The App collects 9-axis IMU data of users' smartphones when the user conducts five activities of daily life (ADL), including walking, hopping, phone calls, waving and typing. The users label the activities themselves by clicking the "start" and "end" buttons shown in the app before and after performing each activity. We finally obtain valid data submissions from 121 users (17-55 years old) with 77 different smartphone models. We resample the original IMU data at 50Hz, with a sliding time window of 2s, and generate a 900-dimension feature for each data sample. This dataset is larger and more heterogeneous, which can be used to evaluate the scalability and robustness of different methods.

[3]All the data collection was approved by IRB of the authors' institution. The datasets are available at https://github.com/xmouyang/FL-Datasets-for-HAR.

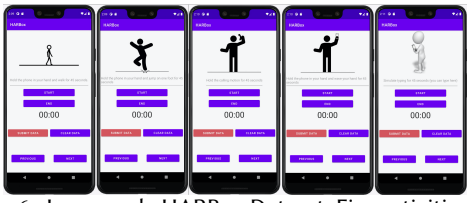


Fig. 6. Large-scale HARBox Dataset: Five activities of daily life (ADL) are recorded using an Android App. We received data record from 121 users with 77 different models of smartphones in total.

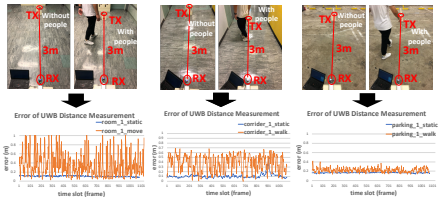


Fig. 7. UWB-based Human Movement Detection. The UWB devices are placed 3m away from each other (in parking lots/corridors/rooms), with or without a person walking between them.

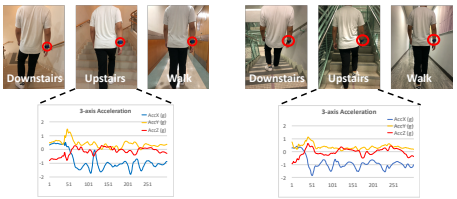


Fig. 8. IMU-based Walking Activity Recognition. Seven participants are recruited to conduct three walking activities in two buildings using an IMU module.

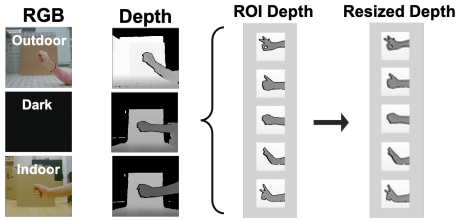


Fig. 9. Depth Camera based Gesture Recognition. Five gestures of two subjects are recorded using a depth camera in outdoor, dark and indoor environments.

Human Movement Detection using UWB: Human movement can influence the multi-path effect of UWB (Ultra Wide Band) signals, thus introducing significant errors to distance measurement between two UWB nodes [25], which can be used to detect human movement. As Fig.7 shows, we conduct experiments in 3 different environments (parking lot/corridor/room) using two UWB nodes that measure distance by two-way ranging (TWR). The two nodes are placed 3m away from each other, with or without a person walking between them, with a sampling rate of 5Hz. Each data record contains a 10-second long recording of normalized errors (50 dimensions), and five statistical attributes (maximum, minimum, mean, standard deviation, and the number of values equal to the mean). This dataset has a small number of data records, with a low dimension of each record. Moreover, the task of human movement detection using UWB is a simple 2-class classification task.

Walking Activity Recognition using IMU: As Fig. 8 shows, we record three walking activities using an off-the-shelf Inertial Measurement Unit (IMU) module. We recruited 7 participants (4 males and 3 females) to conduct three walking activities (walking in corridor/ upstairs/ downstairs) in two buildings. The sampling rate of IMU is 50 Hz, and each frame of data contains 9-axis IMU data. The time window we choose is 2 seconds, which makes each recording a 900-dimensional vector. This dataset is heterogeneous given different subjects and environments involved in the experiments, making it challenging to capture the intrinsic relationship of the subjects' data.

Gesture Recognition using Depth Camera: Unlike RGB camera, depth camera can preserve user's privacy and is increasingly used for activity monitoring and gesture control [33]. Fig. 9 shows the experiment setting of collecting depth data. We record five types of gestures (good/ok/victory/stop/fist) that are conducted by two subjects using a depth camera in three environments (outdoor, dark, and indoor, respectively). We first obtain the ROI (region of interest) of the depth gesture, then normalize the depth value to 0-1 and resize the obtained depth image to 36*36 pixels. This dataset has a large number of data records and the dimension of each data record is relatively high, thus increasing the difficulty of activity recognition.

8 IMPLEMENTATION AND EVALUATION

We design and implement a ClusterFL prototype on a PC (as the server), seven NVIDIA Jetson TX2 and three Jetson AGX Xavier (as nodes). The hardware setup is shown in Fig. 10. The PC (Intel Core i7-9700 CPU 3.0GHz × 8) runs Ubuntu 18.04.5, while the TX2 (6-core ARM CPU) and Xavier (8-core ARM CPU) run Ubuntu 18.04. The server and nodes are connected via a TP-link TL-SG2016K switch. The codes are implemented using Python3.

Our evaluation focuses on three aspects of ClusterFL, including dynamic system performance, performance on different datasets and different models depths. For the experiments on the three small-scale datasets, each node will only train on one edge device (TX2 or Xavier) to measure their energy consumptions. The power are measured using the on-board power monitoring sensor TIINA3221x [30], and only the CPU power is counted. For the large-scale HARBox dataset, we let each CPU run multiple nodes to incorporate up to 120 nodes.

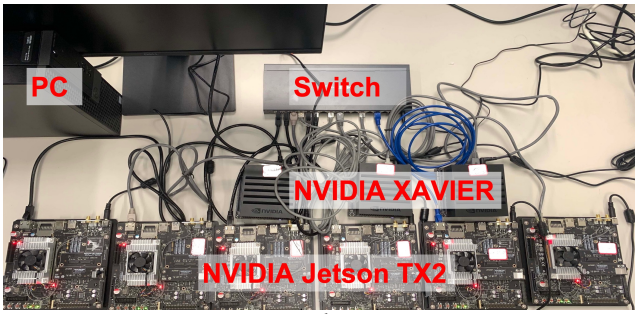


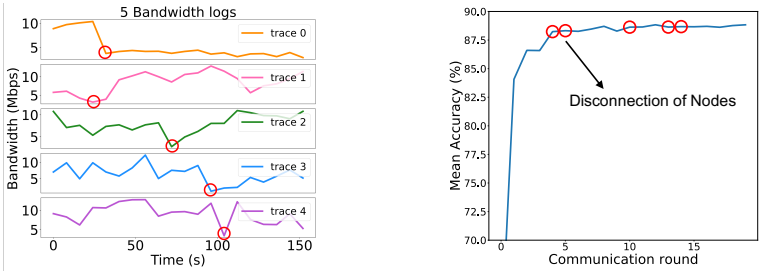
Fig. 10. Hardware Setup

8.1 Dynamic System Performance

In this section, we use the large-scale HARBox dataset which contains data from 121 subjects collected using an App on different smartphones to evaluate the dynamic system performance of ClusterFL. The evaluations include the impact of dynamic network conditions and accuracy / communication overhead performance with different numbers of nodes.

8.1.1 Impact of Dynamic Network Conditions. To verify the effectiveness of our framework in real-world settings, we measure and record the logs of uplink bandwidth of three mobile network connections, i.e., 4G LTE, WiFi, and Ethernet, in several typical indoor/outdoor environments. We find that the bandwidths of WiFi (50-100 Mbps) and Ethernet (400-600 Mbps) are relatively stable while the 4G LTE has substantially lower and more unstable bandwidth. Fig. 11a shows the recorded five bandwidth traces for devices with 4G LTE.

Next, we conduct an experiment using the data of 30 subjects from the large-scale HARBox dataset we collected and the mobile network traces. There are 10 nodes randomly chosen to communicate with the server according to one of 5 bandwidth traces of 4G LTE, 10 nodes using WiFi traces and 10 nodes using Ethernet traces, respectively. In our experiment, the nodes with a bandwidth lower than 4 Mbps will be dropped by the server, as it would cost significantly higher latency for the server to receive messages from them, slowing down the convergence process. Therefore, the nodes with 4G LTE will be unexpectedly disconnected from the server after running a period of time (e.g., at about 30s for trace 1 in Figure 11a, corresponding to the 5th communication round of federated learning in Figure 11b) due to the dynamic traffic. Note that the nodes can also be disconnected at the very beginning if they start with an extremely low uplink bandwidth, which depends on the real-time bandwidth during the federated learning process.



(a) Bandwidth traces of 4G. (b) Testing accuracy with disconnections of 4G nodes.

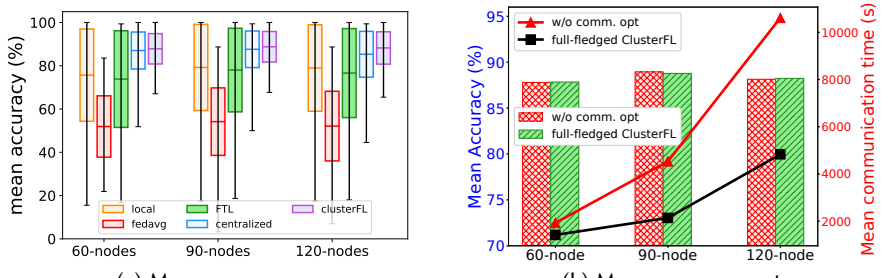
Fig. 11. Performance under dynamic network conditions. At about 30s for trace 1 in (a), the first node is disconnected from the server due to the extremely lower bandwidth, which corresponds to the 5th communication round of federated learning in (b).

Fig. 11b presents the mean testing accuracy with the communication rounds during the learning process of ClusterFL. In the rounds labeled by circles, total ten 4G LTE nodes are disconnected from the server due to extremely low network bandwidth. However, the mean accuracy during the whole process increases steadily since the server dynamically learns the cluster relationship of nodes even though there are unexpected disconnections of nodes, which shows the robustness of ClusterFL.

8.1.2 Scalability. To evaluate the scalability of ClusterFL, we evaluate the performance of different numbers of nodes (60, 90, 120) using data from the large-scale HARBox dataset.

Overall Accuracy. Fig. 12a shows the accuracy of different methods on the HARBox dataset. Generally, when the number of nodes increases, the mean accuracy of centralized learning slightly decreases, which shows the heterogeneity of subjects' data. Moreover, FedAvg performs the worst as it does not converge to the centralized model. In all configurations, ClusterFL outperforms FTL, local learning, FedAvg, and its accuracy even exceeds centralized learning for 60 and 90 nodes, which demonstrates the scalability of ClusterFL. Moreover, the standard deviation of node accuracy for centralized learning is very large for configurations with 60, 90 and 120 nodes, while ClusterFL has a significantly smaller variation of accuracy among nodes, which means that ClusterFL can improve model accuracy for most nodes.

Communication Overhead. Fig. 12b shows the mean accuracy and mean communication cost per node involving different numbers of nodes with or without the two mechanisms we proposed in Section 6. The result shows that, when the number of involved nodes increases, the mean communication cost of nodes increases drastically. However, our two communication optimization mechanisms can save over 50% communication time while keeping accuracy improvement in different system settings, verifying the scalability of ClusterFL.



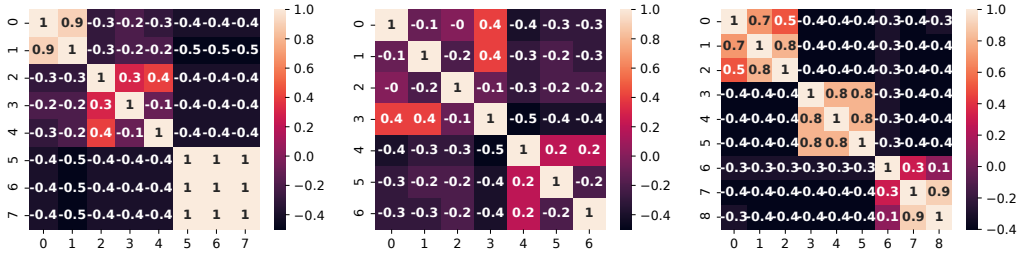
(a) Mean accuracy (b) Mean commu. cost

Fig. 12. Mean accuracy and communication cost per node with the number of nodes increasing.

8.2 Performance on Different Datasets

In this section, we use the three small-scale in-lab HAR datasets to show that ClusterFL can improve overall accuracy through capturing the relationship of nodes. Our baselines include centralized learning, local learning, FedAvg (federated average) and FTL (federated transfer learning). We use the Support Vector Machine model for UWB-based human movement detection, a neural network with two fully connected layers for IMU-based walking activity recognition, and a CNN with two convolutional layers and two fully connected layers for depth camera-based gesture recognition.

Capturing relationship of nodes. Fig. 13 plots the correlation matrix of nodes learned by ClusterFL for UWB, IMU and depth dataset, respectively. For UWB dataset, it shows that nodes with data from the same place (parking lot/corridor/room) are classified to the same cluster by ClusterFL even without prior knowledge, consistent with the observation that the multi-path effect of the UWB signal varies in different locations. For the IMU dataset, the nodes with data from the same building are classified to the same cluster by ClusterFL. However, compared to Fig.13a, the data in the same cluster here yields a higher variance. This is caused by a variety of subjects in different buildings when capturing the cluster relationship. For the depth dataset, the nodes with data from the same environment (outdoor/dark/indoor) are classified to the same cluster by ClusterFL, which can be attributed to distinct influences from the ambient light of environments on the collected depth images.

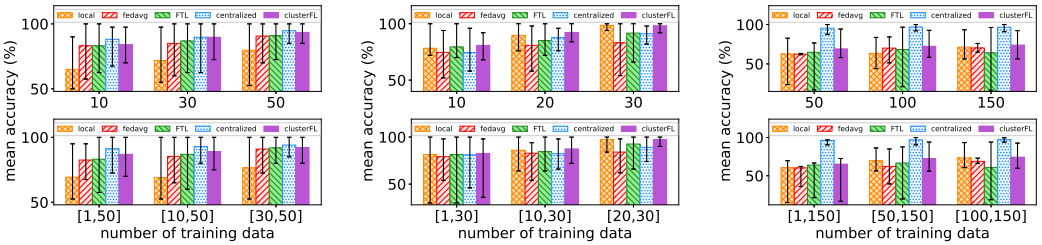


(a) UWB structure (b) IMU structure (c) Depth structure

Fig. 13. Learned relationship among nodes for UWB, IMU, Depth dataset, respectively.

Comparison of different methods. Fig. 14a, 14b and 14c plots the accuracy performance of when the numbers of training samples are same (balanced) and different (unbalanced) on nodes for the three datasets. The unbalanced configuration is motivated by the fact that nodes usually have significantly diverse data due to environmental heterogeneity in real-world human activity recognition applications. Generally, the mean accuracy of each method increases with the number of training samples; when the interval of node's data amounts becomes larger (i.e., nodes are more skewed), the accuracy performance decreases as some nodes have a very small amount of training samples. Table 4 summarizes the averaged accuracy among all of the balanced data settings for each dataset, using different methods; and Table 5 summarizes that of unbalanced data settings. ClusterFL outperforms local/FedAvg/FTL in different settings and even performs better than centralized learning (e.g., 5.99% in a balanced data setting for IMU dataset) in some configurations of node's data and learning model. Moreover, FedAvg can not converge to the centralized model for the three real-world HAR datasets due to the node heterogeneity.

Results on different datasets. Specifically, for the UWB dataset, centralized learning performs best (e.g., 92.71% in an unbalanced setting) due to the ability to train on all nodes' data. What's more, ClusterFL (92.71% in the unbalanced setting) degrades to centralized learning and approach its performance. For the IMU dataset, ClusterFl (e.g., 90.47% in the balanced setting) outperforms other



(a) UWB Accuracy Performance (b) IMU Accuracy Performance (c) Depth Accuracy Performance

Fig. 14. Comparison of Accuracy Performance for UWB, IMU, Depth dataset. ClusterFL outperforms local/FedAvg/FTL in various settings and even performs better than centralized learning for IMU dataset.

methods including the centralized model (84.48%). The main reason is that, here all paradigms use shallow machine learning models with only two fully connected layers. In this case, the single and small centralized model can not well fit the data distributions of different nodes. However, ClusterFL can further improve the accuracy (90.47%) by customizing models for different nodes while enabling collaborative learning among similar nodes. The performance comparison using deeper learning models will be shown in Section 8.3. For the depth dataset, the centralized model outperforms (96.82% in the unbalanced data setting) other methods, while its distributed implementation FedAvg (63.75%) fails to converge to the same model. In this case, ClusterFL (70.68%) outperforms local learning, FedAvg and FTL without access to other user's data.

	Local	FedAvg	FTL	Centralized	ClusterFL
UWB	72.19%	86.25%	86.98%	**90.83 %**	89.06%
IMU	88.86%	79.52%	85.42%	84.48%	**90.47%**
Depth	65.81%	67.52%	65.69%	**96.29 %**	71.82%

Table 4. Mean accuracy in balanced data settings

	Local	FedAvg	FTL	Centralized	ClusterFL
UWB	71.67%	86.25%	87.30%	**92.71%**	**92.71%**
IMU	88.29%	82.00%	86.20%	84.19%	**89.05%**
Depth	67.91 %	63.75%	63.86%	**96.82%**	70.68%

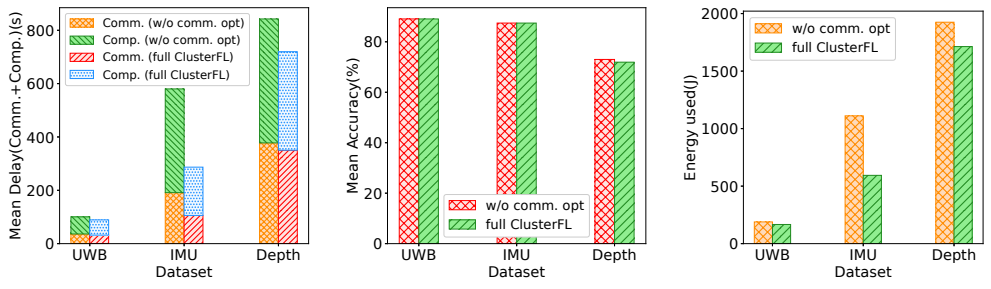
Table 5. Mean accuracy in unbalanced data settings

System overhead. We evaluate the system overhead using the three datasets on our NVIDIA testbed to show the effectiveness of the two communication optimization mechanisms of ClusterFL proposed in Section 6. Table 6 shows the hardware setup of nodes and the nodes that are dropped by the communication optimization mechanisms.

Fig. 15 shows the change of mean communication time, mean computation time, energy consumption and accuracy of nodes under the configuration without communication optimization (w/o comm. opt) and full-fledged ClusterFL. For the three datasets, ClusterFL reduces over 20% communication latency combining the *cluster-wise straggler dropout* and *correlation-based node selection* while maintaining almost the same accuracy performance. Moreover, full-fledged ClusterFL can save about 50%, 16%, 12% energy consumption for the IMU, UWB, and depth datasets, respectively, which verifies the significance of designing effective communication optimization mechanisms for edge devices.

Dataset	number of nodes	number of data records in nodes	configuration of nodes	dropped nodes
UWB	8	[22, 25, 10, 13, 13, 17, 19, 29]	node 0,1,2 on Xavier node 3,4,5,6,7 on Tx2	node 3 and 4
IMU	7	[10, 13, 13, 49, 19, 29, 31]	node 0,1,2 on Xavier node 3,4,5,6 on Tx2	node 3 and 6
Depth	9	[94, 97, 114, 117, 117, 59, 133, 71, 86]	node 0,1,2 on Xavier node 3,4,5,6,7,8 on Tx2	node 5,7,8

Table 6. Setup of nodes for three datasets



(a) Time delay (b) Accuracy (c) Energy

Fig. 15. Performance with or without communication optimization of ClusterFL for UWB, IMU, Depth dataset. w/o comm. opt means without communication optimization.

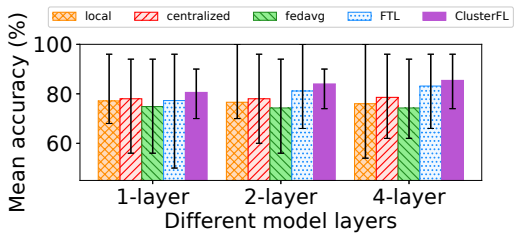
8.3 Performance under Various System Settings

This section introduces the performance of ClusterFL under various system settings, including with different local learning models, comparison with or without communication mechanisms and different choices of hyper-parameters. Note that we use the setup of nodes for the IMU dataset in this section, while the results are similar for other datasets.

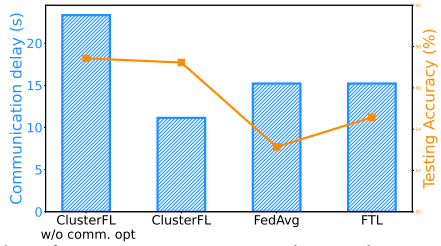
Accuracy with Different Model Depths. Fig. 16a shows the mean accuracy of different paradigms for local models with 1,2,4 hidden layers, with only 10 training samples in each node. When the model becomes deeper (from 1 to 4 layers), the accuracy of centralized learning, FTL and ClusterFL tends to increase. When the model becomes deeper, the mean accuracy of local learning decreases (e.g., 77.14 % for 1-layer while 76% for 4-layer), which means that the trained model is overfitting the limited local data. The accuracy performance of centralized learning increases with the model depth, as a deeper model is more likely to fit the data distribution of all nodes. FedAvg can not achieve the same accuracy of centralized learning and performs worse than local models due to data heterogeneity. Federated transfer learning (FTL) improves the accuracy of Fedavg through fine-tuning the learned global model on the node's local data. ClusterFL consistently outperforms the baselines under different learning models, e.g., 7.4%, 6%, 9.7%, 2.9% over local learning, centralized learning, Fedavg, federated transfer learning, respectively. However, it's likely that when the learning model is large enough, the single centralized model trained on all node's data will achieve the best performance, which corresponds to the results of the depth datasets in Section 8.2. These results also emphasize the significance of choosing proper learning models when we apply different paradigms to HAR applications.

Performance comparison with or without communication optimization. Fig. 16b compares the performance (communication delay and testing accuracy) of ClusterFL with or without communication optimization, as well as the FL baselines Fedavg and FTL. As FTL is a post-training personalized FL approach that fine-tunes the global model learned by Fedavg on the local data of nodes, it has the same communication delay as Fedavg during federated learning. However, FTL has a higher accuracy performance (i.e., 84.57%) when the nodes have heterogeneous data distributions. We observe that the mean communication delay (23.33s) of naive ClusterFL without communication optimization (w/o commu. opt.) is comparable with Fedavg and FTL (15.22s), while the former has a higher mean testing accuracy (87.43% v.s. 83.14%). Moreover, full-fledged ClusterFL with two communication optimization mechanisms can maintain almost the same accuracy performance (87.14%) while significantly reducing the overall communication delay (11.14s) and outperforming Fedavg.

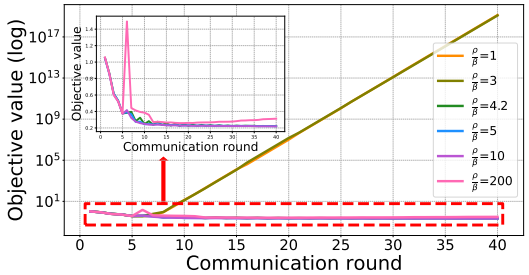
Convergence with different choices of hyper-parameters. Fig. 16c shows how the objective value changes over the communication rounds with different choices of the penalty parameter ρ. Specifically, we set $\alpha = 1 \times 10^{-3}$, $\beta = 5 \times 10^{-4}$ and set the ratio $\frac{\rho}{\beta}$ to 1, 3, 4.2, 5, 10, 200 respectively. The results show that when $\frac{\rho}{\beta} < 4.2$ (i.e., 1 and 3), the objective value diverges to even 10^{18}. Then when $\frac{\rho}{\beta} = 4.2$, there is a fluctuation on the curve of the objective value (the green curve) before the convergence. The update of the objective value shows a more steady convergence when $\frac{\rho}{\beta} > 4.2$ (i.e., 5 and 10). However, when $\frac{\rho}{\beta}$ is too large (e.g., $\frac{\rho}{\beta} = 200$ as shown in the hotpink curve), the convergence performance again becomes worse. The reason is that a larger ρ will enlarge the fluctuations of $\mathbf{U}$ (in Section 5.2) especially when $\mathbf{F}$ is not accurate when being learned at the first time (i.e., the 6th round, as we set $\mathbf{F}$ to be updated every 5 times). The experimental results are in line with our convergence analysis in 5.4.1, which shows that when the other hyper-parameters are fixed, the penalty parameter of ADMM update ρ should be larger than a threshold (e.g., > 4.2 in this experiment) to achieve convergence. However, ρ should not be set too large once the convergence is achieved as a large ρ enlarges the fluctuations of $\mathbf{U}$ and slows down its convergence.



(a) Accuracy with different model depths.



(b) Performance comparison with or without communication optimization.



(c) Convergence with different hyper-parameters ($\frac{\rho}{\beta}$).

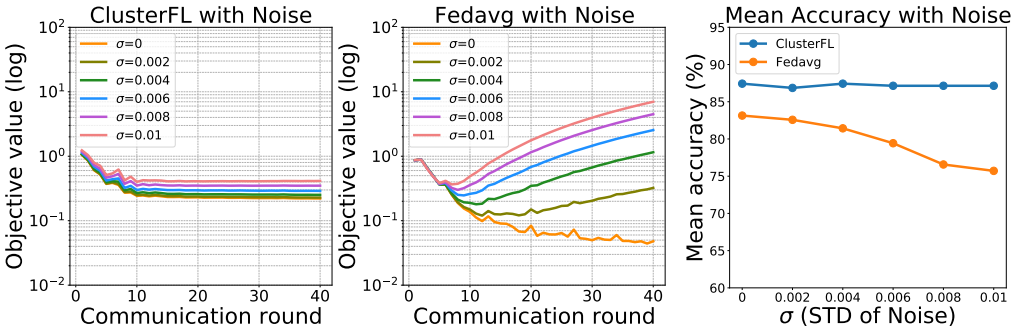
Fig. 16. Performance of ClusterFL under various system settings. Using the IMU dataset as an example.

9 DISCUSSION

9.1 Privacy Analysis of ClusterFL

While avoiding exposing users' raw data during the learning process, the model updates transmitted in ClusterFL may still reveal certain information about user activities [39]. For example, if the central server is deemed as untrustworthy, the uploaded model weights $\mathbf{w}_i (i = 1, ..., M)$ of nodes are subject to the risk of privacy leakage. A common mechanism is to add noises to the model weights before sending to the server [39]. Fig. 17 shows the results after adding different levels of Gaussian Noise $\mathcal{N}(0, \sigma)$ to the transferred model weights (the same setting as the IMU dataset

in Table 6), where a larger σ means adding more noises. Compared with Fedavg that is easier to diverge and suffer accuracy reduction, the performance (convergence and accuracy) of ClusterFL shows less sensitivity to the additive random noises. The reason is that, Fedavg is easy to diverge when a small number of nodes (total seven nodes for Fig. 17) are involved as shown in [39], in which case there will be a higher variance of the artificial noise terms. ClusterFL is more robust as it does not directly aggregate model weights but uses the model output for aggregation, while Fedavg relies on the values of the model weights. Another concern for the private issue of ClusterFL is that, as ClusterFL explores the correlations of nodes based on the model weights, it is possible to derive individual user's behavior from the clustering results [13]. In the future, we will study how to integrate privacy-preserving techniques [40] in ClusterFL and investigate the trade-off between privacy and utility for correlated users in federated learning.



(a) Convergence (b) Accuracy

Fig. 17. Performance Comparison after adding different levels of noises.

9.2 Different Network Topologies in ClusterFL

Currently, we only consider the interactions between the server and the nodes (i.e., a star network) in the framework of ClusterFL. One future work is to apply ClusterFL to FL systems with other network topologies and incorporate new mechanisms to improve the system performance. For example, for a large-scale system involving hundreds or thousands of nodes, we can reduce the communication overhead by integrating a preselection and partition phase in which the system will first run several iterations to learn a preliminary relationship among nodes and further partition the nodes into smaller sub-systems (with different sub-servers) iteratively, in order to meet the desirable requirements on the communication overhead. Moreover, if the nodes are allowed to directly communicate with each other, they can form different clusters through interacting with others to maximize their own rewards (e.g., improvement of model accuracy or the payoff for contributing to the FL systems).

9.3 Applications of ClusterFL

Besides human activity recognition, we will extend ClusterFL to a wider range of real-world applications, where the nodes' data exhibits intrinsic similarity and locality. For example, in health monitoring [36] and road traffic prediction [29] applications, the data of nodes (e.g., users or cars) are shown to share spatial-temporal similarity due to spatial proximity, models of devices/cars, user habits, etc. As a result, ClusterFL can be applied to these applications to enable collaborative learning among similar nodes without exposing users' raw data during the learning process. Moreover, the learned relationship by ClusterFL can be leveraged to guide the post-training model personalization, which is a common but challenging operation in federated learning. For example, if a node is more

correlated to other nodes, its model should be not be personalized a lot on the local data after federated model training; otherwise more post-training should be conducted for obtaining a more customized model.

10 CONCLUSION

In this paper, we propose ClusterFL, a clustering-based federated learning system for human activity recognition. ClusterFL features a novel federated learning framework enabling collaborative learning among similar nodes and integrates two effective communication optimization mechanisms based on the learned cluster structure. We theoretically prove the convergence of the proposed clustering-based federated learning framework and provide guidance on the selection of hyper-parameters for achieving the convergence. Our evaluation using four new real-world datasets shows that, ClusterFL outperforms several learning paradigms (e.g., by 21.04%, 6.46%, 5.41% to local learning, FedAvg, federated transfer learning) and sometimes even approach the accuracy of centralized learning. Moreover, ClusterFL can reduce more than 50% communication latency at the expense of minor accuracy loss.

APPENDICES

A PROOF OF THEOREM 1

To prove Theorem 1, we first show Lemma 1, 2, 3 and their proof under Assumption 1, 2, 3. Then we present the proof of Theorem 1 using the three lemmas.

Lemma 1. *The following are true:*

$$||\mathbf{U}^{t+1} - \mathbf{U}^t||_2^2 \leq \frac{L_f}{\sigma_{max}^2}||\mathbf{W}^{t+1} - \mathbf{W}^t||_2^2 \tag{10}$$

Proof. From the optimality condition of $\mathbf{W}$ update and the dual variable ($\mathbf{U}$) update, we have:

$$\begin{cases} \nabla f(\mathbf{W}^{t+1}) = -\mathbf{F}\mathbf{U}^t - \rho\mathbf{F}(\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1}) \\ \mathbf{U}^{t+1} = \mathbf{U}^t + \rho(\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1}) \end{cases} \Longrightarrow \nabla f(\mathbf{W}^{t+1}) = -\mathbf{F}\mathbf{U}^{t+1}$$

Thus we have $||\nabla f(\mathbf{W}^{t+1}) - \nabla f(\mathbf{W}^t)||_2^2 = ||\mathbf{F}(\mathbf{U}^{t+1} - \mathbf{U}^t)||_2^2 = \sigma_{max}^2||(\mathbf{U}^{t+1} - \mathbf{U}^t)||_2^2$.

Moreover, Assumption 1 implies that $f(\mathbf{W}) = \sum_{i=1}^M f_i(\mathbf{w}_i)$ has Lipschitz continuous gradient with $L_f = \max_i L_{f_i}$. That is, $||\nabla f(\mathbf{W}^{t+1}) - \nabla f(\mathbf{W}^t)||_2^2 \leq L_f||\mathbf{W}^{t+1} - \mathbf{W}^t||_2^2$. Therefore, $\sigma_{max}^2||(\mathbf{U}^{t+1} - \mathbf{U}^t)||_2^2 \leq L_f||\mathbf{W}^{t+1} - \mathbf{W}^t||_2^2$ and Lemma 1 is proved. □

Lemma 2. *We have the following to bound the difference of the augmented Lagrangian for the ADMM update:*

$$L_\rho(\mathbf{W}^{t+1}, \Omega^{t+1}, \mathbf{U}^{t+1}) - L_\rho(\mathbf{W}^t, \Omega^t, \mathbf{U}^t) \leq (\frac{L_f}{\sigma_{max}^2\rho} - \frac{\gamma_i(\rho)}{2})||\mathbf{W}^{t+1} - \mathbf{W}^t||_2^2 \tag{11}$$

Proof. The successive difference of the augmented Lagrangian can be splitted by

$$\begin{aligned} &L_\rho(\mathbf{W}^{t+1}, \Omega^{t+1}, \mathbf{U}^{t+1}) - L_\rho(\mathbf{W}^t, \Omega^t, \mathbf{U}^t) \\ =&L_\rho(\mathbf{W}^{t+1}, \Omega^{t+1}, \mathbf{U}^{t+1}) - L_\rho(\mathbf{W}^{t+1}, \Omega^{t+1}, \mathbf{U}^t) + L_\rho(\mathbf{W}^{t+1}, \Omega^{t+1}, \mathbf{U}^t) - L_\rho(\mathbf{W}^t, \Omega^t, \mathbf{U}^t) \end{aligned}$$

The first two terms of the above equation can be bounded by (according to Assumption 1):

$$\begin{aligned}&L_\rho(\mathbf{W}^{t+1},\Omega^{t+1},\mathbf{U}^{t+1})-L_\rho(\mathbf{W}^{t},\Omega^{t},\mathbf{U}^{t})\\&=\left\langle \mathbf{U}^{t+1}-\mathbf{U}^{t},\mathbf{F}^T(\Omega^{t+1}-\mathbf{W}^{t+1})\right\rangle=\frac{1}{\rho}||\mathbf{U}^{t+1}-\mathbf{U}^{t}||_2^2\le\frac{L_f}{\sigma_{max}^2\rho}||\mathbf{W}^{t+1}-\mathbf{W}^{t}||_2^2\end{aligned}$$

The last two terms can be bounded by:

$$\begin{aligned}&L_\rho(\mathbf{W}^{t+1},\Omega^{t+1},\mathbf{U}^{t})-L_\rho(\mathbf{W}^{t},\Omega^{t},\mathbf{U}^{t})\\=&L_\rho(\mathbf{W}^{t+1},\Omega^{t+1},\mathbf{U}^{t})-L_\rho(\mathbf{W}^{t},\Omega^{t+1},\mathbf{U}^{t})+L_\rho(\mathbf{W}^{t},\Omega^{t+1},\mathbf{U}^{t})-L_\rho(\mathbf{W}^{t},\Omega^{t},\mathbf{U}^{t})\\\overset{(a)}{\le}&\sum_{i=1}^{M}(\left\langle\nabla L_\rho(\mathbf{w}_i^{t+1},\Omega^{t+1},\mathbf{U}^{t}),\mathbf{w}_i^{t+1}-\mathbf{w}_i^{t}\right\rangle-\frac{\gamma_i(\rho)}{2}||\mathbf{w}_i^{t+1}-\mathbf{w}_i^{t}||_2^2)+\left\langle\nabla L_\rho(\mathbf{W}^{t},\Omega^{t+1},\mathbf{U}^{t}),\Omega^{t+1}-\Omega^{t}\right\rangle\\\overset{(b)}{\le}&\sum_{i=1}^{M}(-\frac{\gamma_i(\rho)}{2}||\mathbf{w}_i^{t+1}-\mathbf{w}_i^{t}||_2^2)\overset{(c)}{\le}-\frac{\gamma(\rho)}{2}||\mathbf{W}^{t+1}-\mathbf{W}^{t}||_2^2\end{aligned}$$

where (a) is established since $L(\mathbf{w}_i,\Omega,\mathbf{U})$ is strongly convex w.r.t each $\mathbf{w}_i$ with modulus $\gamma_i(\rho)$ (according to Assumption 3) and $L(\mathbf{W},\Omega,\mathbf{U})$ is convex w.r.t Ω; in (b) we use the optimality condition for $\mathbf{w}_i$ update and Ω update; (c) is established by using $\gamma(\rho)=\max_i\gamma_i(\rho)$ in Assumption 3. Combining the two terms, we obtain the inequality (11) and Lemma 2 is proved. □

Lemma 3. *Let $\{\mathbf{W}^t,\Omega^t,\mathbf{U}^t\}$ be generated by the ADMM update, then the following limit of the augmented Lagrangian exists and is lower bounded by $\underline{P}$ defined in Assumption 2:*

$$\lim_{t\to\infty}L_\rho(\mathbf{W}^t,\Omega^t,\mathbf{U}^t)\ge\underline{P}$$

Proof. We have the following transformations for the augmented Lagrangian:

$$\begin{aligned}&L_\rho(\mathbf{W}^{t+1},\Omega^{t+1},\mathbf{U}^{t})\\=&f(\mathbf{W}^{t+1})+g(\Omega^{t+1})+\left\langle\mathbf{U}^{t+1},\mathbf{F}^T\mathbf{W}^{t+1}-\Omega^{t+1}\right\rangle+\frac{\rho}{2}||\mathbf{F}^T\mathbf{W}^{t+1}-\Omega^{t+1}||_2^2\\\overset{(a)}{=}&f(\mathbf{W}^{t+1})+g(\Omega^{t+1})+\left\langle\nabla g(\Omega^{t+1}),\mathbf{F}^T\mathbf{W}^{t+1}-\Omega^{t+1}\right\rangle+\frac{\rho}{2}||\mathbf{F}^T\mathbf{W}^{t+1}-\Omega^{t+1}||_2^2\\\overset{(b)}{=}&f(\mathbf{W}^{t+1})-\beta||\Omega^{t+1}||_2^2+\left\langle-2\beta\Omega^{t+1},\mathbf{F}^T\mathbf{W}^{t+1}\right\rangle+2\beta||\Omega^{t+1}||_2^2+\beta||\mathbf{F}^T\mathbf{W}^{t+1}||_2^2\\&-\beta||\mathbf{F}^T\mathbf{W}^{t+1}||_2^2+\frac{\rho}{2}||\mathbf{F}^T\mathbf{W}^{t+1}-\Omega^{t+1}||_2^2\\\overset{(c)}{=}&f(\mathbf{W}^{t+1})+g(\mathbf{F}^T\mathbf{W}^{t+1})+(\beta+\frac{\rho}{2})||\mathbf{F}^T\mathbf{W}^{t+1}-\Omega^{t+1}||_2^2\ge P(\mathbf{W}^{t+1})\end{aligned}\tag{12}$$

where (a) is established since $\mathbf{U}^{t+1}=\nabla g(\Omega^{t+1})$ satisfies for all t; (b) is established by substituting $g(\Omega^{t+1})=-\beta||\Omega^{t+1}||_2^2$ and $\nabla g(\Omega^{t+1})=-2\beta\Omega^{t+1}$; (c) is obtained by rearranging the items and substituting $g(\mathbf{F}^T\mathbf{W}^{t+1})=-\beta||\mathbf{F}^T\mathbf{W}^{t+1}||_2^2$.

Equation (12) and Assumption 2 imply that $L_\rho(\mathbf{W}^{t+1},\Omega^{t+1},\mathbf{U}^t)$ is lower bounded. Moreover, if ρ is chosen large enough such that Assumption 3 is satisfied, then Lemma 2 implies $L_\rho(\mathbf{W}^{t+1},\Omega^{t+1},\mathbf{U}^t)$ is monotonically decreasing. Therefore $L_\rho(\mathbf{W}^{t+1},\Omega^{t+1},\mathbf{U}^t)$ is convergent over the iterates and Lemma 3 is proved. □

Proof. (Theorem 1)

- For the first item of Theorem 1: Lemma 2 and Lemma 12 implies that $L_\rho(\mathbf{W}^{t+1},\Omega^{t+1},\mathbf{U}^t)$ converges over the iterates. Moreover, we have $||\mathbf{W}^{t+1}-\mathbf{W}^t||_2^2\to 0$ when $t\to\infty$ according to the relationship of (11) in Lemma 2, which implies the convergence of $\mathbf{W}$ in (4a); Then the

relationship in (10) of Lemmar 1 implies the convergence of $\mathbf{U}^{t+1}$ thus the convergenc of $\mathbf{U}$ in (4b) satisfies; Lastly, from the update of $\mathbf{U}$: $\mathbf{U}^{t+1} - \mathbf{U}^t = \rho(\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1})$, $||\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1}||_2^2$ is convergent and thus (4c) is satisfied.

- For the second term of Theorem 1, since $\{\mathbf{W}^*, \Omega^*, \mathbf{U}^*\}$ is a limit point of $\{\mathbf{W}^{t+1}, \Omega^{t+1}, \mathbf{U}^{t+1}\}$, we have:

$$\mathbf{W}^{t+1} \to \mathbf{W}^*, \Omega^{t+1} \to \Omega^*, \mathbf{U}^{t+1} \to \mathbf{U}^* \tag{13}$$

From the optimal condition of $\mathbf{W}$ update and Ω update, we have:

$$\nabla f(\mathbf{W}^{t+1}) + \mathbf{F}\mathbf{U}^t + \rho\mathbf{F}(\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1}) = 0$$
$$\nabla g(\Omega^{t+1}) - \mathbf{U}^t + \rho(\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1}) = 0$$

Then due to (4c), we have $\lim_{t\to\infty} \nabla f(\mathbf{W}^{t+1}) + \mathbf{F}\mathbf{U}^t = 0$ and $\lim_{t\to\infty} \nabla g(\Omega^{t+1}) - \mathbf{U}^t = 0$. Combining (13), we have the KKT conditions in (5a) and (5b). Moreover, combining the update of $\mathbf{U}$, the convergence of $\mathbf{U}$ in (4b) and (13), the last KKT condition in (5c) is satisfied.

- For the third term of Theorem 1, if we assume the compactness of $\mathcal{W}$, then $\mathbf{W}^{t+1}$ will converge to at least one limit point of the sequence. As shown in the proof for the first term, Ω^{t+1} and $\mathbf{U}^{t+1}$ will also converge to the corresponding limit points. Moreover, according to the second term (Any limit point of Problem (1) is a stationary point), the sequence $\{\mathbf{W}^{t+1}, \Omega^{t+1}, \mathbf{U}^{t+1}\}$ will converge to the set of stationary points satisfying the KKT conditions.

□

B PROOF OF THEOREM 2

Remark 2. *Due to the convexity in Assumption 4, the necessary and sufficient optimality conditions for the ADMM problem are primal feasibility and dual feasibility from the KKT conditions:*

$$\mathbf{F}^T\mathbf{W}^* - \Omega^* = \mathbf{0} \qquad \text{(Primal feasibility)}$$
$$0 \in \partial f(\mathbf{W}^*) + \mathbf{F}\mathbf{U}^* \qquad \text{(Dual feasibility A)}$$
$$0 \in \partial g(\Omega^*) - \mathbf{U}^* \qquad \text{(Dual feasibility B)}$$

Proof. (Theorem 2) Assumption 4 implies that $f(\mathbf{W}) = \sum_{i=1}^M f_i(\mathbf{w}_i)$ is strongly convex. With $m_f = \min_i m_{f_i}$, we have:

$$m_f||\mathbf{W}^{t+1} - \mathbf{W}^*||_2^2 \le \left\langle \nabla f(\mathbf{W}^{t+1}) - \nabla f(\mathbf{W}^*), \mathbf{W}^{t+1} - \mathbf{W}^* \right\rangle \tag{14}$$

For the update of $\mathbf{W}^{t+1}$, the optimal condition is:

$$\nabla L_\rho(\mathbf{W}^{t+1}, \Omega^t, \mathbf{U}^{t+1}) = \nabla f(\mathbf{W}^{t+1}) + \mathbf{F}\mathbf{U}^t + \rho\mathbf{F}(\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1}) = 0$$
$$\implies \nabla f(\mathbf{W}^{t+1}) = -\mathbf{F}\mathbf{U}^t - \rho\mathbf{F}(\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1})$$

Combining the Dual feasibility A and $\mathbf{U}^{t+1} = \mathbf{U}^t + \rho(\mathbf{F}^T\mathbf{W}^{t+1} - \Omega^{t+1})$, we have:

$$\nabla f(\mathbf{W}^{t+1}) - \nabla f(\mathbf{W}^*) = \rho\mathbf{F}(\Omega^t - \Omega^{t+1}) - \mathbf{F}(\mathbf{U}^{t+1} - \mathbf{U}^*)$$

Then for the right hand side of (14):

$$
\begin{aligned}
&\left\langle \mathbf{W}^{t+1}-\mathbf{W}^{*}, \nabla f(\mathbf{W}^{t+1})-\nabla f(\mathbf{W}^{*})\right\rangle \\
=&\left\langle \mathbf{W}^{t+1}-\mathbf{W}^{*}, \rho \mathbf{F}(\Omega^{t}-\Omega^{t+1})-\mathbf{F}(\mathbf{U}^{t+1}-\mathbf{U}^{*})\right\rangle \\
=&\rho\left\langle \mathbf{F}^{T}(\mathbf{W}^{t+1}-\mathbf{W}^{*}), \Omega^{t}-\Omega^{t+1}\right\rangle-\left\langle \mathbf{F}^{T}(\mathbf{W}^{t+1}-\mathbf{W}^{*}),-(\mathbf{U}^{t+1}-\mathbf{U}^{*})\right\rangle \\
\overset{(a)}{=}&\rho\left\langle \Omega^{t+1}-\Omega^{*}, \Omega^{t}-\Omega^{t+1}\right\rangle+\frac{1}{\rho}\left\langle \mathbf{U}^{t}-\mathbf{U}^{t+1}, \mathbf{U}^{t+1}-\mathbf{U}^{*}\right\rangle-\left\langle \mathbf{U}^{t}-\mathbf{U}^{t+1}, \Omega^{t}-\Omega^{t+1}\right\rangle-\left\langle \Omega^{t+1}-\Omega^{*}, \mathbf{U}^{t+1}-\mathbf{U}^{*}\right\rangle \\
\overset{(b)}{=}&\rho\left\langle \Omega^{t+1}-\Omega^{*}, \Omega^{t}-\Omega^{t+1}\right\rangle+\frac{1}{\rho}\left\langle \mathbf{U}^{t}-\mathbf{U}^{t+1}, \mathbf{U}^{t+1}-\mathbf{U}^{*}\right\rangle+2\beta||\Omega^{t}-\Omega^{t+1}||_{2}^{2}+2\beta||\Omega^{t+1}-\Omega^{*}||_{2}^{2} \\
\overset{(c)}{=}&\frac{1}{2}(||\mathbf{V}^{t}-\mathbf{V}^{*}||_{\mathbf{G}}^{2}-||\mathbf{V}^{t+1}-\mathbf{V}^{*}||_{\mathbf{G}}^{2}-||\mathbf{V}^{t}-\mathbf{V}^{t+1}||_{\mathbf{G}}^{2})+2\beta||\Omega^{t}-\Omega^{t+1}||_{2}^{2}+2\beta||\Omega^{t+1}-\Omega^{*}||_{2}^{2}
\end{aligned}
\tag{15}
$$

In the derivation of (15): (a) is established since $\mathbf{F}^{T}(\mathbf{W}^{t+1}-\mathbf{W}^{*})=\mathbf{F}^{T}(\mathbf{W}^{t+1}-\mathbf{F}^{T}\mathbf{W}^{*})=\frac{1}{\rho}(\mathbf{U}^{t+1}-\mathbf{U}^{t})+\Omega^{t+1}-\Omega^{*}$. (b) is established since $\nabla g(\Omega^{t+1})-\mathbf{U}^{t+1}=0 \Leftrightarrow \mathbf{U}^{t+1}=-2\beta\Omega^{t+1}$ satisfies for all t. (c) is established due to Definition 2. Then substituting (15) to (14), we have:

$$
\begin{aligned}
&m_{f}||\mathbf{W}^{t+1}-\mathbf{W}^{*}||_{2}^{2} \leq \frac{1}{2}(||\mathbf{V}^{t}-\mathbf{V}^{*}||_{\mathbf{G}}^{2}-||\mathbf{V}^{t+1}-\mathbf{V}^{*}||_{\mathbf{G}}^{2}-||\mathbf{V}^{t}-\mathbf{V}^{t+1}||_{\mathbf{G}}^{2})+2\beta||\Omega^{t}-\Omega^{t+1}||_{2}^{2}+2\beta||\Omega^{t+1}-\Omega^{*}||_{2}^{2} \\
&\Leftrightarrow ||\mathbf{V}^{t}-\mathbf{V}^{*}||_{\mathbf{G}}^{2}-||\mathbf{V}^{t+1}-\mathbf{V}^{*}||_{\mathbf{G}}^{2} \geq ||\mathbf{V}^{t}-\mathbf{V}^{t+1}||_{\mathbf{G}}^{2}+2m_{f}||\mathbf{W}^{t+1}-\mathbf{W}^{*}||_{2}^{2}-4\beta||\Omega^{t}-\Omega^{t+1}||_{2}^{2}-4\beta||\Omega^{t+1}-\Omega^{*}||_{2}^{2}
\end{aligned}
$$

Moreover, the Q-linear convergence in Theorem 2 is equivalent to: $||\mathbf{V}^{t}-\mathbf{V}^{*}||_{\mathbf{G}}^{2}-||\mathbf{V}^{t+1}-\mathbf{V}^{*}||_{\mathbf{G}}^{2} \geq \delta||\mathbf{V}^{t+1}-\mathbf{V}^{*}||_{\mathbf{G}}^{2}$. Therefore, we only need to proof the following for Theorem 2:

$$
\begin{aligned}
&||\mathbf{V}^{t}-\mathbf{V}^{t+1}||_{\mathbf{G}}^{2}+2m_{f}||\mathbf{W}^{t+1}-\mathbf{W}^{*}||_{2}^{2}-4\beta||\Omega^{t}-\Omega^{t+1}||_{2}^{2}-4\beta||\Omega^{t+1}-\Omega^{*}||_{2}^{2} \geq \delta||\mathbf{V}^{t+1}-\mathbf{V}^{*}||_{\mathbf{G}}^{2} \\
\overset{(a)}{\Longleftrightarrow}&\rho||\Omega^{t}-\Omega^{t+1}||_{2}^{2}+\frac{1}{\rho}||\mathbf{U}^{t}-\mathbf{U}^{t+1}||_{2}^{2}-4\beta||\Omega^{t}-\Omega^{t+1}||_{2}^{2}-4\beta||\Omega^{t+1}-\Omega^{*}||_{2}^{2}+2m_{f}||\mathbf{W}^{t+1}-\mathbf{W}^{*}||_{2}^{2} \\
&\geq \delta\rho||\Omega^{t+1}-\Omega^{*}||_{2}^{2}+\frac{\delta}{\rho}||\mathbf{U}^{t+1}-\mathbf{U}^{*}||_{2}^{2} \\
\overset{(b)}{\Longleftrightarrow}&(\rho+\frac{4\beta^{2}}{\rho}-4\beta)||\Omega^{t}-\Omega^{t+1}||_{2}^{2}+2m_{f}||\mathbf{W}^{t+1}-\mathbf{W}^{*}||_{2}^{2} \geq (\delta(\rho+\frac{4\beta^{2}}{\rho}-4\beta)+4\beta)||\Omega^{t+1}-\Omega^{*}||_{2}^{2} \\
\overset{(c)}{\Longleftrightarrow}&\frac{(\rho+\frac{4\beta^{2}}{\rho}-4\beta)}{(\delta(\rho+\frac{4\beta^{2}}{\rho}-4\beta)+4\beta)}||\Omega^{t}-\Omega^{t+1}||_{2}^{2}+\frac{2m_{f}}{(\delta(\rho+\frac{4\beta^{2}}{\rho}-4\beta)+4\beta)}||\mathbf{W}^{t+1}-\mathbf{W}^{*}||_{2}^{2} \geq ||\Omega^{t+1}-\Omega^{*}||_{2}^{2}
\end{aligned}
\tag{16}
$$

In the above derivation of (16): (a) is established due to Definition 2. (b) is established by substituting $\mathbf{U}^{t+1}=-2\beta\Omega^{t+1}, \mathbf{U}^{t}=-2\beta\Omega^{t}, \mathbf{U}^{*}=-2\beta\Omega^{*}$. (c) is established by dividing $(\delta(\rho+\frac{4\beta^{2}}{\rho}-4\beta)+4\beta>0)$ for the two sides. Moreover, as $\mathbf{F}^{T}(\mathbf{W}^{t+1}-\mathbf{W}^{*})=\Omega^{t+1}-\Omega^{*}+\frac{1}{\rho}(\mathbf{U}^{t+1}-\mathbf{U}^{t})=\Omega^{t+1}-\Omega^{*}-\frac{2\beta}{\rho}(\Omega^{t+1}-\Omega^{t})$, let $a=\mathbf{F}^{T}(\mathbf{W}^{t+1}-\mathbf{W}^{*}), b=\frac{2\beta}{\rho}(\Omega^{t+1}-\Omega^{t}), a+b=\Omega^{t+1}-\Omega^{*}$. According to the triangle inequality $||a||_{2}^{2}+||b||_{2}^{2} \geq ||a+b||_{2}^{2}$, we have:

$$
\begin{aligned}
&||\mathbf{F}^{T}(\mathbf{W}^{t+1}-\mathbf{W}^{*})||_{2}^{2}+\frac{4\beta^{2}}{\rho^{2}}||(\Omega^{t}-\Omega^{t+1})||_{2}^{2} \geq ||\Omega^{t+1}-\Omega^{*}||_{2}^{2} \\
\Longleftrightarrow\ &\frac{4\beta^{2}}{\rho^{2}}||(\Omega^{t}-\Omega^{t+1})||_{2}^{2}+\sigma_{max}^{2}||\mathbf{W}^{t+1}-\mathbf{W}^{*}||_{2}^{2} \geq ||\Omega^{t+1}-\Omega^{*}||_{2}^{2}
\end{aligned}
\tag{17}
$$

Here σ_{max} is the largest singular value of the cluster indicator matrix $\mathbf{F}$. We do not assume $\sigma_{max} = 1$ to tolerate a non-optimal solution of $\mathbf{F}$ in Section 5.3. According to (16) and (17), we have:

$$\begin{cases} \dfrac{(\rho + \frac{4\beta^2}{\rho} - 4\beta)}{(\delta(\rho + \frac{4\beta^2}{\rho} - 4\beta) + 4\beta)} \geq \dfrac{4\beta^2}{\rho^2} \Longrightarrow \quad \delta \leq \dfrac{\rho^2 - 4\beta\rho}{4\beta^2} \\ \dfrac{2m_f}{(\delta(\rho + \frac{4\beta^2}{\rho} - 4\beta) + 4\beta)} \geq \sigma_{max}^2 \Longrightarrow \quad \delta \leq \dfrac{\rho(\frac{2m_f}{\sigma_{max}^2} - 4\beta)}{\rho^2 - 4\beta^2} \end{cases}$$

Therefore, δ in Theorem 2 satisfies (16) hence proving the Q-linear convergence. □

REFERENCES

[1] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. 2013. A public domain dataset for human activity recognition using smartphones.. In *Esann*, Vol. 3. 3.

[2] James C Bezdek and Richard J Hathaway. 2003. Convergence of alternating optimization. *Neural, Parallel Sci. Comput.* 11, 4 (2003), 351–368.

[3] Chongguang Bi, Guoliang Xing, Tian Hao, Jina Huh, Wei Peng, and Mengyan Ma. 2017. FamilyLog: A mobile system for monitoring family mealtime activities. In *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 21–30.

[4] Andreas Biri, Neal Jackson, Lothar Thiele, Pat Pannuto, and Prabal Dutta. 2020. SociTrack: infrastructure-free interaction tracking through mobile sensor networks. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–14.

[5] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.* 3, 1 (2011), 1–122.

[6] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao. 2020. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intell. Syst.* (2020).

[7] Wei Deng and Wotao Yin. 2016. On the global and linear convergence of the generalized alternating direction method of multipliers. *J. Sci. Comput.* 66, 3 (2016), 889–916.

[8] Chris Ding and Xiaofeng He. 2004. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*. 29.

[9] Jie Feng, Can Rong, Funing Sun, Diansheng Guo, and Yong Li. 2020. Pmf: A privacy-preserving human mobility prediction framework via federated learning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 1 (2020), 1–21.

[10] Alzheimer's Drug Discovery Foundation. 2021. ALZHEIMER'S BIOMARKERS, EXPLAINED. Retrieved 2021 from https://www.alzdiscovery.org/news-room/blog/alzheimers-biomarkers-explained

[11] Avishek Ghosh, Justin Hong, Dong Yin, and Kannan Ramchandran. 2019. Robust federated learning in a heterogeneous environment. (2019). arXiv:1906.06629

[12] Gene Glass and Kenneth Hopkins. 1996. Statistical methods in education and psychology. *Psyccritiques* 41, 12 (1996).

[13] Google. 2020. FLOC Whitepaper of Google. Retrieved 2020 from https://github.com/google/ads-privacy/blob/master/proposals/FLoC/FLOC-Whitepaper-Google.pdf

[14] Tian Hao, Guoliang Xing, and Gang Zhou. 2013. iSleep: unobtrusive sleep quality monitoring using smartphones. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*. 1–14.

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. (2015). arXiv:1503.02531

[16] Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. 2015. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 3836–3840.

[17] Chen Huang, Chen Change Loy, and Xiaoou Tang. 2016. Local similarity-aware deep feature embedding. (2016). arXiv:1610.08904

[18] Wenjun Jiang, Chenglin Miao, Fenglong Ma, Shuochao Yao, Yaqing Wang, Ye Yuan, Hongfei Xue, Chen Song, Xin Ma, Dimitrios Koutsonikolas, et al. 2018. Towards environment independent device free human activity recognition. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. 289–304.

[19] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. 2016. Federated optimization: Distributed machine learning for on-device intelligence. (2016). arXiv:1610.02527

[20] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2016. Federated learning: Strategies for improving communication efficiency. (2016). arXiv:1610.05492

[21] Solomon Kullback. 1997. *Information theory and statistics*. Courier Corporation.
[22] Fan Lai, Xiangfeng Zhu, Harsha V Madhyastha, and Mosharaf Chowdhury. 2021. Oort: Efficient federated learning via guided participant selection. In *15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21)*. 19–35.
[23] Seulki Lee and Shahriar Nirjon. 2020. Fast and scalable in-memory deep multitask learning via neural weight virtualization. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*. 175–190.
[24] Jia Li, Yu Rong, Helen Meng, Zhihui Lu, Timothy Kwok, and Hong Cheng. 2018. Tatc: Predicting alzheimer's disease with actigraphy data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 509–518.
[25] Jing Li, Zhaofa Zeng, Jiguang Sun, and Fengshan Liu. 2012. Through-wall detection of human being's movement by UWB radar. *IEEE Geosci. Remote. Sens. Lett.* 9, 6 (2012), 1079–1083.
[26] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2018. Federated optimization in heterogeneous networks. (2018). arXiv:1812.06127
[27] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems* 2 (2020), 429–450.
[28] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. 2016. Communication-efficient learning of deep networks from decentralized data. (2016). arXiv:1602.05629
[29] Wanli Min and Laura Wynter. 2011. Real-time road traffic prediction with spatio-temporal correlations. *Transp. Res. Part C Emerg. Technol.* 19, 4 (2011), 606–616.
[30] NVIDIA. 2021. NVIDIA Jetson Linux Developer Guide, 32.4.3 Release. Retrieved 2021 from https://docs.nvidia.com/jetson/l4t/index.html#page/Tegra
[31] Xiaomin Ouyang, Zhiyuan Xie, Jiayu Zhou, Jianwei Huang, and Guoliang Xing. 2021. ClusterFL: a similarity-aware federated learning system for human activity recognition. In *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*. 54–66.
[32] Sashank J Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and Hugh Brendan McMahan. 2021. Adaptive Federated Optimization. In *International Conference on Learning Representations*.
[33] Zhou Ren, Junsong Yuan, and Zhengyou Zhang. 2011. Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In *Proceedings of the 19th ACM international conference on Multimedia*. 1093–1096.
[34] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. 2017. Federated multi-task learning. In *Advances in Neural Information Processing Systems*. 4424–4434.
[35] Konstantin Sozinov, Vladimir Vlassov, and Sarunas Girdzijauskas. 2018. Human Activity Recognition Using Federated Learning. In *2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*. IEEE, 1103–1111.
[36] Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. 2015. Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition. In *Proceedings of the 13th ACM conference on embedded networked sensor systems*. 127–140.
[37] Linlin Tu, Xiaomin Ouyang, Jiayu Zhou, Yuze He, and Guoliang Xing. 2021. Feddl: Federated learning via dynamic layer sharing for human activity recognition. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 15–28.
[38] Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and computing* 17, 4 (2007), 395–416.
[39] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Trans. Inf. Forensics Secur.* 15 (2020), 3454–3469.
[40] Liyang Xie, Inci M Baytas, Kaixiang Lin, and Jiayu Zhou. 2017. Privacy-preserving distributed multi-task learning with asynchronous updates. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1195–1204.
[41] Zhiyuan Xie, Xiaomin Ouyang, Xiaoming Liu, and Guoliang Xing. 2021. UltraDepth: Exposing High-Resolution Texture from Depth Cameras. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 302–315.
[42] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. 2020. Salvaging federated learning by local adaptation. (2020). arXiv:2002.04758
[43] Tianlong Yu, Tian Li, Yuqiong Sun, Susanta Nanda, Virginia Smith, Vyas Sekar, and Srinivasan Seshan. 2020. Learning Context-Aware Policies from Multiple Smart Homes via Federated Multi-Task Learning. In *2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, 104–115.

[44] Hongyuan Zha, Xiaofeng He, Chris Ding, Ming Gu, and Horst Simon. 2001. Spectral relaxation for k-means clustering. *Advances in neural information processing systems* 14 (2001).

[45] Hanbin Zhang, Chenhan Xu, Huining Li, Aditya Singh Rathore, Chen Song, Zhisheng Yan, Dongmei Li, Feng Lin, Kun Wang, and Wenyao Xu. 2019. Pdmove: Towards passive medication adherence monitoring of parkinson's disease using smartphone-based gait assessment. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 3 (2019), 1–23.

[46] Shizhen Zhao, Wenfeng Li, and Jingjing Cao. 2018. A user-adaptive algorithm for activity recognition based on k-means clustering, local outlier factor, and multivariate gaussian distribution. *Sensors* 18, 6 (2018), 1850.

[47] Jiayu Zhou, Jianhui Chen, and Jieping Ye. 2011. Clustered multi-task learning via alternating structure optimization. *Adv. Neural Inf. Process. Syst.* 2011 (2011), 702.