ClusterFL: A Similarity-Aware Federated Learning System for Human Activity Recognition

Xiaomin Ouyang
The Chinese University of Hong Kong
Hong Kong SAR, China
xmouyang@link.cuhk.edu.hk

Zhiyuan Xie
The Chinese University of Hong Kong
Hong Kong SAR, China
xavier_ie@link.cuhk.edu.hk

Jiayu Zhou
Michigan State University
East Lansing, MI, USA
jiayuz@egr.msu.edu

Jianwei Huang
The Chinese University of Hong Kong, Shenzhen
Shenzhen Institute of Artificial Intelligence and Robotics for Society
Shenzhen, China
jianweihuang@cuhk.edu.cn

Guoliang Xing*
The Chinese University of Hong Kong
Hong Kong SAR, China
glxing@cuhk.edu.hk

ABSTRACT

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 similarity-aware 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 maximizes the training accuracy of multiple learned models while automatically capturing the intrinsic clustering relationship among the data of different nodes. Based on the learned cluster relationship, ClusterFL can efficiently drop out the nodes that converge slower or have little correlation with other nodes 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 total 145 users. The results show that, ClusterFL outperforms several state-of-the-art FL paradigms in terms of overall accuracy, and save more than 50% communication overhead at the expense of negligible accuracy degradation.

CCS CONCEPTS

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

*Corresponding author

MobiSys '21, June 24-July 2, 2021, Virtual, WI, USA

ACM ISBN 978-1-4503-8443-8/21/07…$15.00
https://doi.org/10.1145/3458864.3467681

KEYWORDS

Human activity recognition, Federated learning, Clustering, Multi-task learning, Communication optimization

ACM Reference Format:
Xiaomin Ouyang, Zhiyuan Xie, Jiayu Zhou, Jianwei Huang, and Guoliang Xing. 2021. ClusterFL: A Similarity-Aware Federated Learning System for Human Activity Recognition. In *The 19th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '21), June 24-July 2, 2021, Virtual, WI, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3458864.3467681

1 INTRODUCTION

Recently deep learning has been increasingly adopted in human activity recognition [11, 51, 61]. 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 [65]. 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 [24, 26].

Federated learning (FL) [29, 38] 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) [9] 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 [12, 50] due to the heterogeneity of different users' data. Recently, post-training personalization [12, 27] was proposed to improve the accuracy of the global FL model through personalizing the model on-device. However, it's challenging to balance local personalization and learning from other nodes [47, 59]. Another method is *Federated multi-task learning* [49], which optimizes multiple models simultaneously in a distributed setting. However, current federated multi-task learning approaches [49, 60] 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 similarity-aware 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 significant spatial-temporal similarity [11, 51]. We show that such similarity can be captured by cluster structures in a wide range of HAR applications, based on the analysis of six HAR datasets consisting of data of total 184 users.

Motivated by this observation, we propose to exploit the similarity of users' data to simultaneously improve the model accuracy and communication efficiency of FL. First, ClusterFL features a novel *clustered multi-task federated learning* formulation. By introducing a cluster indicator matrix indicating the similarity of users, our new formulation aims to minimize 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) [10], 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 achieve the data locality of nodes. Compared with previous solutions based on post-training personalization [12, 27], our learning framework enables collaborative learning among similar nodes and is applicable to general non-convex learning models (e.g., DNN and CNN 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 stragglers who converge slower and the nodes that are less related to others in each cluster, which is demonstrated to reduce the overall communication overhead while maintaining the overall accuracy performance. 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 involing 121 subjects collected using a smartphone application in a crowdsourcing manner, and three in-lab HAR datasets. 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.

Our key contributions are summarized as follows:

- We proposed ClusterFL, a similarity-aware 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.
- 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 collect four new HAR datasets with significant dynamics, including a large-scale dataset collected using an Android App and three HAR datasets collected in indoor environments.
- We conduct extensive experiments on our NVIDIA edge testbed using four new HAR datasets.We demonstrate the superior performance of ClusterFL compared with several state-of-the-art baselines under dynamic system configurations and various datasets.

2 RELATED WORK

Human Activity Recognition (HAR). Machine Learning has been increasingly adopted in the area of human activity recognition [48, 51, 61]. Various algorithms based on handcrafted features [8, 22] and deep neural networks [26, 37, 44] 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 [31, 63]. 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 [9, 29] aim to learn a single model for all users by averaging the model weights. However, the single learned model usually has limited generality, making it poorly suited for heterogeneous user data in HAR applications [12]. To deal with this issue, federated transfer learning (FTL) is proposed [12, 15] to improve the accuracy of the global learning model by personalizing it for local data. However, compared with ClusterFL, such an approach usually has limited accuracy improvement as it can not efficiently take advantage of the intrinsic relationship among the data of a large number of users. Moreover, Smith et al. [49] propose to learn multiple models simultaneously under the FL settings. However, their method only works for convex models such as SVM. ClusterFL, in contrast, is applicable to general non-convex models (e.g., DNN and CNN). Previous studies on communication-effective FL either choose stragglers (the nodes who converge slower) from all nodes [9, 35] or focus on the quantization techniques [29, 46], 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 [12, 15, 50, 60]. Specifically, Sozinov et al. [50] utilize FedAvg for HAR and achieve slightly worse accuracy compared to centralized models. Chen et al. [12] 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. [15] apply FL to human mobility prediction and propose a fine-tuned personal adaptor to improve the prediction performance. However, these transfer learning methods are based on the FedAvg model without exploiting the relationships among nodes. Yu et al. [60] adopt federated multi-task learning to learn contextual access control policies for smart home applications, which only works 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 [6, 23, 64]. We investigate data similarity of six real-word HAR datasets listed in Table 1, including a public smartphone-based human activity recognition (SHAR) dataset [5], a public heterogeneous HAR (HHAR) dataset [51]; 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 [17] of the data from different users, which is a statistical metric between 0 and 1 and quantifies the clustering tendency of data by measuring the probability that a given data set is generated by uniform distribution. 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.

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.

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 Figure 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.

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

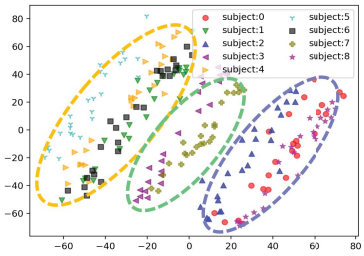


Figure 1: The data of "walking" from the HHAR dataset after reducing dimension to 2D using PCA. There exists a clear cluster relationship among different subjects' data.

(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 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 [9, 49], where nodes only upload model weights to the server to generate one model by averaging their model weights. Federated transfer learning [12, 15] 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. 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	73.17
STD (%)	9.74	12.812	10.19	6.06	3.40

Table 2: Accuracy comparison of different paradigms.

Firstly, 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 [36, 50]. 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 73.17% 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 [33, 62], wearable and ambient sensors continuously track a patient's daily activities such as indoor/outdoor time, sleeping, etc, which are important digital biomarkers [1] for early Alzheimer's diagnosis. Other representative applications include fitness tracking [28], family daily routine monitoring [8] and social distancing detection [57]. 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, ClusterFL on the devices can communicate through the cloud 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. ClusterFL features a novel similarity-aware 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[1]. Specifically, we formulate 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. In addition, the server will drop nodes that are less related to others in the same cluster. In this case, there will be less nodes interacting with the server so that the overall communication time can be reduced while "more important" nodes are kept to perform ClusterFL.

Figure 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 weights and the update

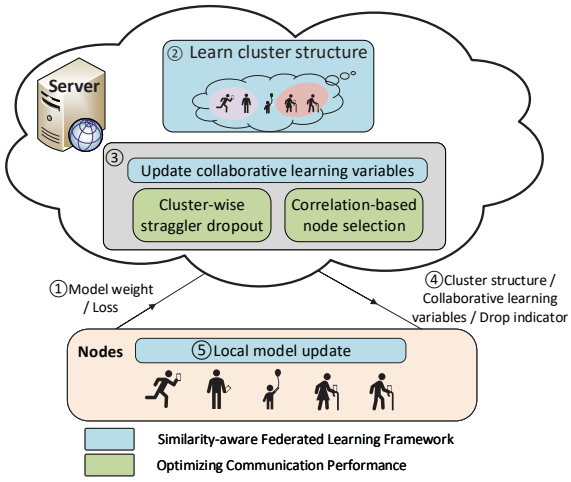


Figure 2: System Architecture of ClusterFL

of 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 cluster indicator matrix learned by the server, 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 information, 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 SIMILARITY-AWARE FEDERATED LEARNING FRAMEWORK

The design of ClusterFL is based on the key observation that data of many applications in human activity recognition have 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:

[1] 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.

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

- 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 $F_{i,j} = \frac{1}{\sqrt{N_j}}$ if node i belongs to j-th cluster and $F_{i,j} = 0$ otherwise. Here K is the number of clusters and N_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 [7] 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.

5.2 Optimize Model Weights

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

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

Here, we propose to use the Alternating Direction Method of Multipliers (ADMM) approach [10] 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 (1) as follows:

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

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

$$\text{where: } f(\mathbf{W}) = \sum_{i=1}^{M} \frac{1}{N_i} \sum_{r=1}^{N_i} l_t(\mathbf{w}_i^T \mathbf{x}_i^r, y_i^r) + \alpha tr(\mathbf{W}\mathbf{W}^T)$$

$$g(\Omega) = -\beta tr(\Omega\Omega^T)$$

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. Finally, an iteration t+1 of 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 $(\mathbf{x}_i, \mathbf{y}_i)$, the cluster structure $\mathbf{F}$ and the collaborative learning variables $\Omega, \mathbf{U}$ from the server.

$$\mathbf{w}_i^{t+1} = \arg\min_{\mathbf{w}_i} \frac{1}{N_i} \sum_{r=1}^{N_i} l_t(\mathbf{w}_i^T \mathbf{x}_i^r, y_i^r) + (\alpha + \frac{\rho}{2} \sum_{j=1}^{K} F_{ij}^2) ||\mathbf{w}_i||_2^2 + \sum_{j=1}^{K} F_{ij} (\mathbf{U}_j^t - \rho \Omega_j^t)^T \cdot \mathbf{w}_i$$

- **Server Update** (line 6-10 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 the collaborative learning variables $\Omega, \mathbf{U}$. For j = 1,...,K:

$$\Omega_j^{t+1} = \arg\min_{\Omega_j} -\beta ||\Omega_j||_2^2 - \Omega_j \cdot (\mathbf{U}_j^t)^T + \frac{\rho}{2} ||\Omega_j - \mathbf{F}_j^T \mathbf{W}^{t+1}||_2^2$$

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

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 broadcast $\mathbf{F}, \Omega, \mathbf{U}$ to all nodes after aggregating the weights. Here $\mathbf{U}_j \in \mathbb{R}^{1 \times D} (j = 1, ..., K)$ is dual variables introduced in the augmented Lagrangian of ADMM.

Algorithm 1: Alternating Optimization of ClusterFL

```
   Initialization : set W = 0^{M×D}, F = 0^{M×M}
1  for h = 0 to H do
2      1. Optimization of W with F fixed.
3      for t = 0 to T do
4          node update: parallelly update w_i(i = 1,...,M)
5              w_i^{t+1} ← Local SGD(w_i^t, (x_i^r, y_i^r), F, Ω, U)
6          server update: update Ω and U
7          for j = 1 to K do
8              Ω_j^{t+1} ← arg min_{Ω_j} L_ρ(Ω_j, U_j^t, F_j^t, W^{t+1})
9              U_j^{t+1} ← U_j^t + ρ(F_j W^{t+1} − Ω_j^{t+1})
10         end
11     end
12     2. Optimization of F with W fixed.
13         server update: Update F based on Section 5.3
14 end
```

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 [16, 25, 45], 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 [30] is used to measure how one probability distribution is different from the other and is widely used in knowledge distillation [20, 40], model adaptation [58, 59] and similarity measurement[13, 21, 52]. 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 KL divergence matrix $\mathbf{D} \in \mathbb{R}^{M \times M}$, with $\mathbf{D}_{i,j} = D_{KL}(\mathbf{w}_i, \mathbf{w}_j)$.

Optimize cluster relationship. Next, the server will learn cluster indicator matrix $\mathbf{F}$ with the model KL divergence matrix $\mathbf{D}$ of nodes. As defined in Section 5.1, $\mathbf{F}$ should satisfy the constraint $F_{i,j} = \frac{1}{\sqrt{N_j}}$ if node i belongs to j-th cluster and $F_{i,j} = 0$ otherwise. This constraint defines a discrete feasible set, which not only requires a known total number of clusters K but also makes finding the optimal $\mathbf{F}$ NP-hard [41]. Therefore, we choose to relax this constraint of $\mathbf{F}$ to obtain a continuous solution that quantifies correlation among all nodes and then recover $\mathbf{F}$.

According to [14], principal components are the continuous solutions to the discrete cluster membership indicators for K-means clustering. Suppose there are total M clusters among nodes, then $\mathbf{P} = \mathbf{Q}_{M-1}\mathbf{Q}_{M-1}^T \in \mathbb{R}^{M \times M}$ will be the continuous solution of F, where $\mathbf{Q}_{M-1} = (\mathbf{v}_1, ..., \mathbf{v}_{M-1})$ 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 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.

6 OPTIMIZING COMMUNICATION PERFORMANCE

In traditional FL systems [9, 38], a large number of update iterations between nodes and the server are required, which makes communication overhead the bottleneck of the learning process. Some previous work on communication-effective FL focuses on the model compression and quantization techniques [29, 46], 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 [9, 35], which does not consider the differences 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. Figure 3 shows the update of the cluster indicator matrix $\mathbf{F}$, where we use the L1-norm of $\mathbf{F}$ to show how $\mathbf{F}$ changes over the number of times that $\mathbf{F}$ is updated, using the depth dataset (see Section 7). The task here is to recognize five gestures using the depth images collected from three environments (outdoor/dark/indoor). We can see that $\mathbf{F}$ remains almost unchanged after updated for 8 times even if 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.

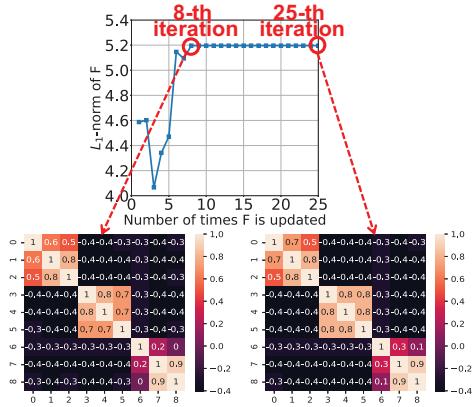


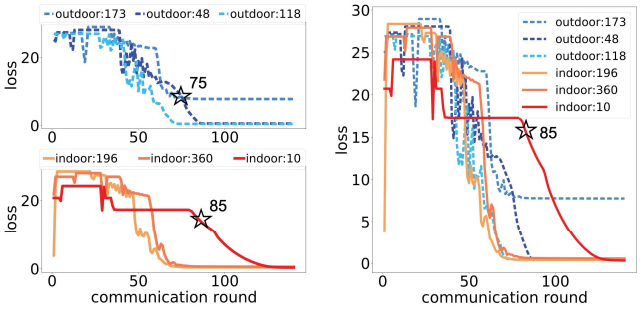
Figure 3: The cluster structure of nodes (F) can be captured after updated several times.

Motivated by this key observation, we propose two novel mechanisms to reduce communication overhead while maintaining the accuracy of learned models. First, using the clustering relationship and loss update of nodes, the server will classify *stragglers* within clusters and drop them early. Second, the server will drop nodes whose data is less related to other nodes in the same cluster.

6.1 Cluster-wise Straggler Dropout

We first use an example to illustrate the key advantage of dropping stragglers within clusters. Figure 4 shows the loss update of six nodes from the environment "outdoor" and "indoor" of the depth dataset.

As shown in Figure 4a, our cluster-wise straggler dropout 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). In contrast, the baseline identifies stragglers from all nodes and drops them simultaneously (Figure 4b), resulting in only one node in the cluster "indoor" dropped. This example suggests that it's more flexible and efficient to identify stragglers within clusters, which is also demonstrated to reduce the communication overhead while maintaining the overall accuracy performance in our experiments in Section 8. Next, we discuss how to measure



(a) Within clusters (b) Among all nodes

Figure 4: Identifying stragglers within clusters (a) and among all nodes (b). Our cluster-wise straggler dropout can dynamically identify and drop stragglers within each cluster. "indoor:196" means the node has 196 data samples from the environment "indoor".

the convergence performance of nodes to identify stragglers for a specific cluster. For cluster j (j = 1,...,K) which contains N_j nodes, we define the averaged convergence rate γ_q for node q (q = 1,...,N_j) in the cluster during the latest T_c iterations as follows:

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

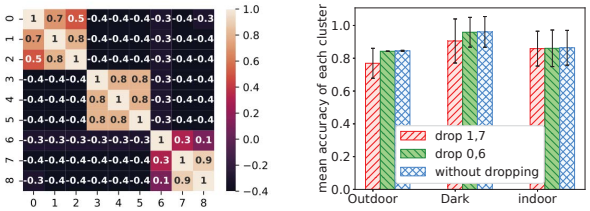
$$\varepsilon_q^t = \frac{|loss_q(t) - loss_q(t-1)|}{\sum_{q=1}^{N_j}|loss_q(t) - loss_q(t-1)|}$$

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 interference while also putting off the iteration to recognize 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 Figure 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 Figure 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.



(a) Correlation matrix of nodes (b) Mean accuracy of each cluster

Figure 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 **F** to measure their degree of correlation with other nodes in each cluster. Suppose node q is in the cluster j which contains N_j nodes, we define the importance metric σ_q for node q as follows:

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

As shown in Figure 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 $M - M_d$. 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 DATA COLLECTION

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.[2]

[2]All the data collection was approved by IRB of the authors' institution. More details of the datasets are available at http://aiot.ie.cuhk.edu.hk/.

Application	Task	Data Dimension	Number of Subjects	Number of Data Records	Sensor	Environment
Human Movement Detection using UWB	with/without Human Movement	55	8	663	Decawave DWM1000 UWB	node 0,1 from parking lot node 2,3,4 from corridor node 5,6,7 from room
Walking Activity Recognition using IMU	walking on corridors/ upstairs/downstairs	900	7	1369	LPMS-B2 IMU	node 0,1,2,3 from building 1 node 4,5,6 from building 2
Gesture Recognition using Depth Camera	good/ok/ victory/stop/fist	1296	9	7422	PicoZense DCAM710	node 0,1,2 from "outdoor" node 3,4,5 from "dark" node 6,7,8 from "indoor"
HARBox: ADL Recognition using Smartphones	walking/hopping/ phone calls/waving/typing	900	121	32935	77 different smartphone models	121 subjects (17-55 years old) Sampling rate: 43.5-57.5Hz

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

The reasons we use self-collected datasets are as follows. First, most of the existing HAR datasets lack 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 datasets collected in real-world settings. Using a new smartphone App, we collect data of total 121 users with 77 different models of personal 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 in recent years [18, 22]. 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 will label the activities themselves by clicking the "start" and "end" buttons shown in the app before and after performing each activity. After removing the invalid and repeated data from total 137 submissions, we finally obtained 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.

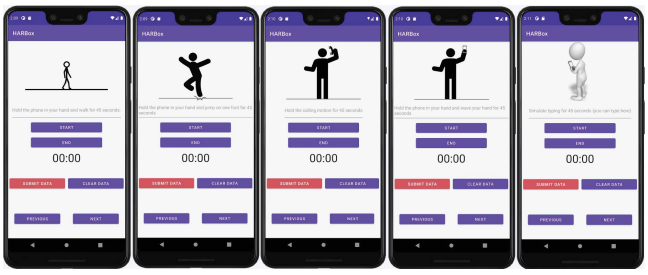


Figure 6: Large-scale HARBox Dataset: Five activities of daily life (ADL) are recorded using an Android App. We received data record from 121 users with 77 different models of smartphones in total.

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 [32, 34]. As Fig.7 shows, we conduct experiments to detect human movement in 3 different environments (parking lot/corridor/room) using two UWB nodes that measure distance by two-way ranging. 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.

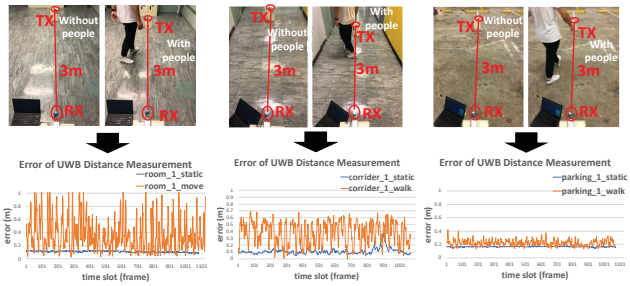


Figure 7: UWB-based Human Movement Detection. The UWB devices are placed 3m away from each other, with or without a person walking between them. We conduct experiments in parking lots/corridors/rooms.

Walking Activity Recognition using IMU: As Figure 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/ walking upstairs/ walking 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.

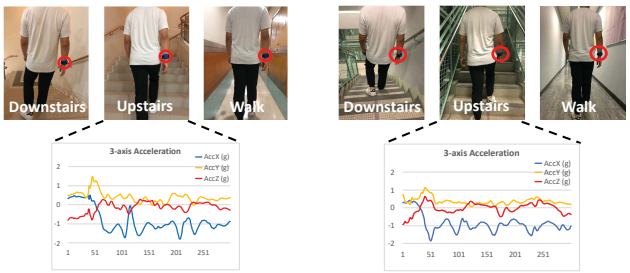


Figure 8: IMU-based Walking Activity Recognition. Seven participants are recruited (4 males and 3 females) to conduct three types of activities in two buildings using an off-the-shelf IMU module.

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 [42, 43, 54]. Figure 9 shows the experiment setting of collecting depth data. We record five types of gestures (good/ok/vic-tory/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.

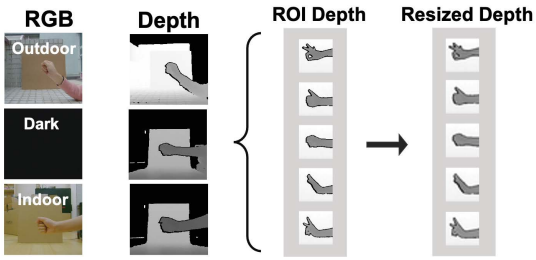


Figure 9: Depth Camera based Gesture Recognition. Five gestures of two subjects are recorded in outdoor, dark and indoor environments.

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 Figure 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 [2], 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.

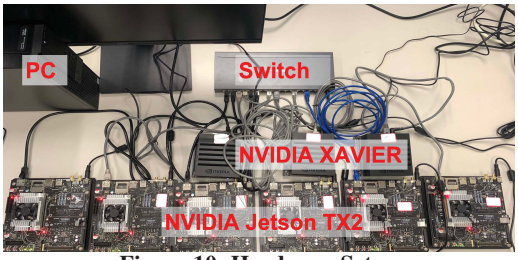


Figure 10: Hardware Setup

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 connectivites, 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. Figure 11 (left) shows the recorded five bandwidth traces for devices with 4G LTE.

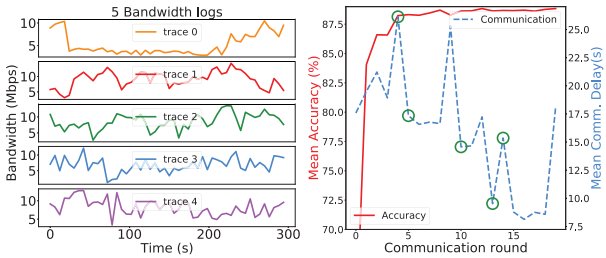


Figure 11: Performance under dynamic network conditions. Left: Five bandwidth traces of 4G LTE. Right: Performance during the communication rounds

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 due to the dynamic traffic.

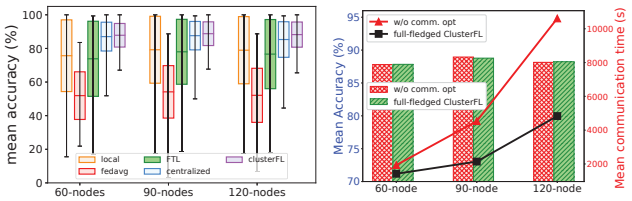
Figure 11 (right) presents the mean communication delay per node and testing accuracy during the communication rounds. Firstly, there is a peak of mean communication delay every five rounds as the inner iteration in Algorithm 1 is set as 5 to update the cluster indicator matrix $\mathbf{F}$. In the rounds labeled by circles, the 4G LTE

nodes are disconnected from the server due to extremely low network bandwidth. After that, the mean communication latency decreases since the number of involved nodes is reduced. However, the mean accuracy during the whole process increases stably 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. Figure 13a 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. Figure 13b 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.



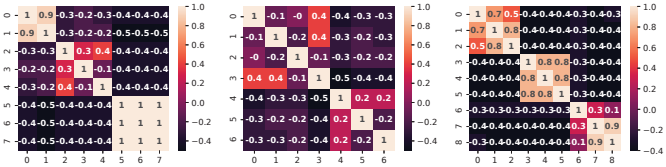
(a) Mean accuracy **(b) Mean commu. cost per node**

Figure 13: 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 SVM (Support Vector Machine) 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. Figure 14 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 Figure 14a, 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**

Figure 14: Learned relationship among nodes for UWB, IMU, Depth dataset, respectively.

	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%

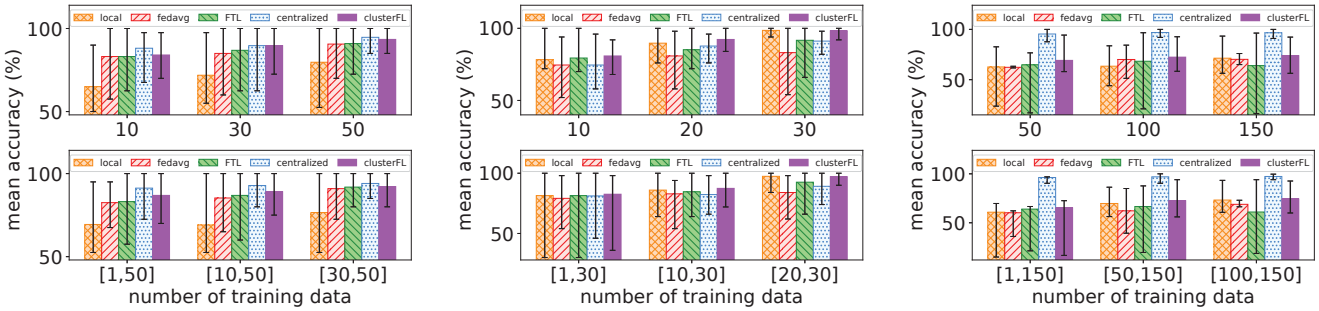
Table 4: Mean accuracy in balanced data settings

	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

Comparison of different methods. Figure 12a, 12b and 12c 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



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

Figure 12: Comparison of Accuracy Performance for UWB, IMU, Depth dataset, respectively. ClusterFL outperforms local/FedAvg/FTL in various settings and even performs better than centralized learning for IMU dataset.

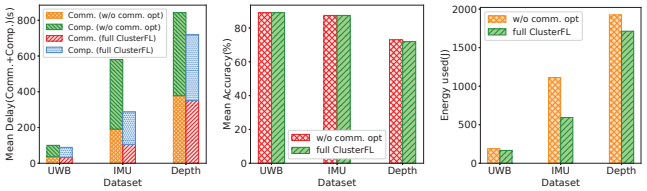
setting) due to the ability to train on all nodes' data. What's more, ClusterFL (92.71% in the unbalanced setting) degrades to centralized learning and approach its performance. For the IMU dataset, ClusterFl (e.g., 90.47% in the balanced setting) outperforms other methods including the centralized model (84.48%). The main reason is that, 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 user's data.

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 setup of nodes and the nodes that are dropped by the communication optimization mechanisms.

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

Figure 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.



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

Figure 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 with Different Model Depths

In this section, we fix the data configurations of nodes and change the depth of learning models to evaluate the accuracy performance of different paradigms. Specifically, we use the IMU dataset, and the number of data samples on each node is 10, 13, 13, 49, 19, 29, 31, respectively. We try models with 1,2,4 fully connected layers for all paradigms.

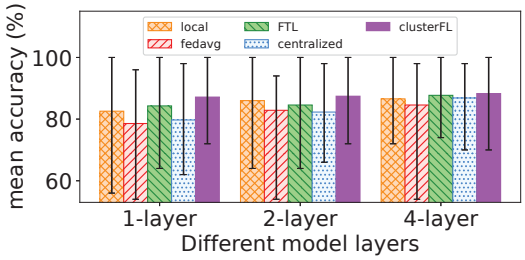


Figure 16: The accuracy of IMU dataset for models with different depths.

Figure 16 summarizes the accuracy results. Basically, when the model becomes deeper (from 1 to 4 layers), the accuracy of all methods tends to increase. An important observation is that when using the 1-layer model, the centralized learning fails to fit all nodes and even performs worse than local learning; however, its performance approaches and then surpasses local learning when the model becomes deeper. FedAvg can not achieve the same accuracy performance of centralized learning and performs worse than local models due to data heterogeneity. Moreover, ClusterFL has a more "stable" accuracy performance under different learning models and outperforms

other methods. 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 UWB and 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.

9 DISCUSSION

Convergence of ClusterFL. In our experiments (Section 8), ClusterFL is demonstrated to converge on the four real-world HAR datasets using different machine learning models (e.g., DNN and CNN models). We now provide some insights into the convergence guarantee of ClusterFL. Firstly, the penalty hyper-parameter ρ of ADMM update (see Section 5.2) plays an important role on the convergence of ClusterFL. Specifically, ClusterFL will likely converge (to at least a stationary point) when ρ is larger than a threshold; otherwise, the objective value of the problem (1) will diverge. Second, it is shown in our experiments that model initialization (e.g., random seed or zeros initialization) will not have a major impact on the performance of the learned model. We will leave it as the future work to study the impact of different initializations on the convergence process.

Future work. Here we discuss some extensions of this work. First, while avoiding exposing users' raw data during the learning process, the model updates transmitted in ClusterFL may still reveal certain information about user activities [27, 56]. In the future, we will study how to integrate privacy-preserving techniques [55] and investigate the trade-off between privacy and utility [3, 53]. Second, we will extend ClusterFL to a wider range of applications, where the nodes' data exhibits intrinsic similarity and locality. For example, in health monitoring [4] and road traffic prediction [39] 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. Third, currently, we only consider the interaction between the server and the nodes in our framework, without direct communication among nodes (clients). We will leave it as future work to apply our approach to other network topologies.

10 CONCLUSION

In this paper, we propose ClusterFL, a similarity-aware 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. 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 approaches the accuracy of centralized learning. Moreover, ClusterFL can reduce more than 50% communication latency at the expense of minor accuracy loss.

ACKNOWLEDGMENTS

This work is supported by the Research Grants Council (RGC) of Hong Kong, China, under GRF grants #14203420 and #14209619, and the Shenzhen Institute of Artificial Intelligence and Robotics for Society.

REFERENCES

[1] 2017. ALZHEIMER'S BIOMARKERS, EXPLAINED. https://www.alzdiscovery.org/news-room/blog/alzheimers-biomarkers-explained.
[2] 2020. NVIDIA Jetson Linux Developer Guide, 32.4.3 Release. https://docs.nvidia.com/jetson/l4t/index.html#page/Tegra.
[3] 2021. FLOC Whitepaper of Google. https://github.com/google/ads-privacy/blob/master/proposals/FLoC/FLOC-Whitepaper-Google.pdf.
[4] Khan Alam, Salman Qureshi, and Thomas Blaschke. 2011. Monitoring spatio-temporal aerosol patterns over Pakistan based on MODIS, TOMS and MISR satellite data and a HYSPLIT model. *Atmospheric environment* 45, 27 (2011), 4641–4651.
[5] 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.
[6] Akram Bayat, Marc Pomplun, and Duc A Tran. 2014. A study on human activity recognition using accelerometer data from smartphones. *Procedia Computer Science* 34 (2014), 450–457.
[7] James C Bezdek and Richard J Hathaway. 2002. Some notes on alternating optimization. In *AFSS International Conference on Fuzzy Systems*. Springer, 288–300.
[8] 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.
[9] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konecny, Stefano Mazzocchi, H Brendan McMahan, et al. 2019. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046* (2019).
[10] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning* 3, 1 (2011), 1–122.
[11] Liang Cao, Yufeng Wang, Bo Zhang, Qun Jin, and Athanasios V Vasilakos. 2018. GCHAR: An efficient Group-based Context—Aware human activity recognition on smartphone. *J. Parallel and Distrib. Comput.* 118 (2018), 67–80.
[12] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao. 2020. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems* (2020).
[13] Oliver M Cliff, Mikhail Prokopenko, and Robert Fitch. 2018. Minimising the Kullback–Leibler divergence for model selection in distributed nonlinear systems. *Entropy* 20, 2 (2018), 51.
[14] 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.
[15] Jie Feng, Can Rong, Funing Sun, Diansheng Guo, and Yong Li. 2020. Pmf: A privacy-preserving human mobility prediction framework via federated learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–21.
[16] Avishek Ghosh, Justin Hong, Dong Yin, and Kannan Ramchandran. 2019. Robust federated learning in a heterogeneous environment. *arXiv preprint arXiv:1906.06629* (2019).
[17] Gene Glass and Kenneth Hopkins. 1996. Statistical methods in education and psychology. *Psyccritiques* 41, 12 (1996).
[18] 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.
[19] H.B.McMahan and D.Ramage. 2017. Federated Learning: Collaborative Machine Learning without Centralized Training Data. https://www.googblogs.com/federated-learning-collaborative-machine-learning-without-centralized-training-data/.
[20] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).
[21] Chen Huang, Chen Change Loy, and Xiaoou Tang. 2016. Local similarity-aware deep feature embedding. *arXiv preprint arXiv:1610.08904* (2016).
[22] Sinh Huynh, Rajesh Krishna Balan, JeongGil Ko, and Youngki Lee. 2019. VitaMon: measuring heart rate variability using smartphone front camera. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 1–14.
[23] Tâm Huynh and Bernt Schiele. 2005. Analyzing features for activity recognition. In *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*. 159–163.
[24] Andrey Ignatov. 2018. Real-time human activity recognition from accelerometer data using Convolutional Neural Networks. *Applied Soft Computing* 62 (2018), 915–922.

[25] Bin Jiang, Jian Pei, Yufei Tao, and Xuemin Lin. 2011. Clustering uncertain data based on probability distribution similarity. *IEEE Transactions on Knowledge and Data Engineering* 25, 4 (2011), 751–763.

[26] 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.

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

[28] Harini Kolamunna, Yining Hu, Diego Perino, Kanchana Thilakarathna, Dwight Makaroff, Xinlong Guan, and Aruna Seneviratne. 2016. AFit: Adaptive fitness tracking by application function virtualization. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. 309–312.

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

[30] Solomon Kullback. 1997. *Information theory and statistics*. Courier Corporation.

[31] 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.

[32] Jing Li, Lanbo Liu, Zhaofa Zeng, and Fengshan Liu. 2012. Simulation and signal processing of UWB radar for human detection in complex environment. In *2012 14th International Conference on Ground Penetrating Radar (GPR)*. IEEE, 209–213.

[33] 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.

[34] Jing Li, Zhaofa Zeng, Jiguang Sun, and Fengshan Liu. 2012. Through-wall detection of human being's movement by UWB radar. *IEEE Geoscience and Remote Sensing Letters* 9, 6 (2012), 1079–1083.

[35] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2018. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127* (2018).

[36] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2019. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189* (2019).

[37] Yong-Lu Li, Liang Xu, Xinpeng Liu, Xijie Huang, Yue Xu, Shiyi Wang, Hao-Shu Fang, Ze Ma, Mingyang Chen, and Cewu Lu. 2020. PaStaNet: Toward Human Activity Knowledge Engine. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 382–391.

[38] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. 2016. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629* (2016).

[39] Wanli Min and Laura Wynter. 2011. Real-time road traffic prediction with spatio-temporal correlations. *Transportation Research Part C: Emerging Technologies* 19, 4 (2011), 606–616.

[40] Gaurav Kumar Nayak, Konda Reddy Mopuri, Vaisakh Shaj, R Venkatesh Babu, and Anirban Chakraborty. 2019. Zero-shot knowledge distillation in deep networks. *arXiv preprint arXiv:1905.08114* (2019).

[41] R Gary Parker and Ronald L Rardin. 2014. *Discrete optimization*. Elsevier.

[42] Zhou Ren, Jingjing Meng, and Junsong Yuan. 2011. Depth camera based hand gesture recognition and its applications in human-computer-interaction. In *2011 8th International Conference on Information, Communications & Signal Processing*. IEEE, 1–5.

[43] 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.

[44] Charissa Ann Ronao and Sung-Bae Cho. 2016. Human activity recognition with smartphone sensors using deep learning neural networks. *Expert systems with applications* 59 (2016), 235–244.

[45] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. 2020. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[46] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. 2019. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems* (2019).

[47] Neta Shoham, Tomer Avidor, Aviv Keren, Nadav Israel, Daniel Benditkis, Liron Mor-Yosef, and Itai Zeitak. 2019. Overcoming forgetting in federated learning on non-iid data. *arXiv preprint arXiv:1910.07796* (2019).

[48] Deepika Singh, Erinc Merdivan, Ismini Psychoula, Johannes Kropf, Sten Hanke, Matthieu Geist, and Andreas Holzinger. 2017. Human activity recognition using recurrent neural networks. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*. Springer, 267–274.

[49] 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.

[50] 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.

[51] 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.

[52] Huibing Wang, Lin Feng, Xiangzhu Meng, Zhaofeng Chen, Laihang Yu, and Hongwei Zhang. 2017. Multi-view metric learning based on KL-divergence for similarity measurement. *Neurocomputing* 238 (2017), 269–276.

[53] 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 Transactions on Information Forensics and Security* 15 (2020), 3454–3469.

[54] Lu Xia and JK Aggarwal. 2013. Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2834–2841.

[55] 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.

[56] Runhua Xu, Nathalie Baracaldo, Yi Zhou, Ali Anwar, and Heiko Ludwig. 2019. Hybridalpha: An efficient approach for privacy-preserving federated learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*. 13–23.

[57] Dongfang Yang, Ekim Yurtsever, Vishnu Renganathan, Keith A Redmill, and Ümit Özgüner. 2020. A vision-based social distancing and critical density detection system for covid-19. *arXiv preprint arXiv:2007.03578* (2020), 24–25.

[58] Dong Yu, Kaisheng Yao, Hang Su, Gang Li, and Frank Seide. 2013. KL-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 7893–7897.

[59] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. 2020. Salvaging federated learning by local adaptation. *arXiv preprint arXiv:2002.04758* (2020).

[60] 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.

[61] Ming Zeng, Le T Nguyen, Bo Yu, Ole J Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. 2014. Convolutional neural networks for human activity recognition using mobile sensors. In *6th International Conference on Mobile Computing, Applications and Services*. IEEE, 197–205.

[62] 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. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies* 3, 3 (2019), 1–23.

[63] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. 2018. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 6848–6856.

[64] 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.

[65] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. 2014. Learning deep features for scene recognition using places database. *Advances in neural information processing systems* 27 (2014), 487–495.