Meta-HAR: Federated Representation Learning for Human Activity Recognition

Chenglin Li
University of Alberta
Edmonton, Canada
ch11@ualberta.ca

Di Niu
University of Alberta
Edmonton, Canada
dniu@ualberta.ca

Bei Jiang
University of Alberta
Edmonton, Canada
bei1@ualberta.ca

Xiao Zuo
Tencent
Shenzhen, China
royzuo@tencent.com

Jianming Yang
Tencent
Shenzhen, China
kimmyyang@tencent.com

ABSTRACT

Human activity recognition (HAR) based on mobile sensors plays an important role in ubiquitous computing. However, the rise of data regulatory constraints precludes collecting private and labeled signal data from personal devices at scale. Thanks to the growth of computational power on mobile devices, federated learning has emerged as a decentralized alternative solution to model training, which iteratively aggregates locally updated models into a shared global model, therefore being able to leverage decentralized, private data without central collection. However, the effectiveness of federated learning for HAR is affected by the fact that each user has different activity types and even a different signal distribution for the same activity type. Furthermore, it is uncertain if a single global model trained can generalize well to individual users or new users with heterogeneous data. In this paper, we propose Meta-HAR, a *federated representation learning* framework, in which a signal embedding network is *meta-learned* in a federated manner, while the learned signal representations are further fed into a personalized classification network at each user for activity prediction. In order to boost the representation ability of the embedding network, we treat the HAR problem at each user as a different task and train the shared embedding network through a Model-Agnostic Meta-learning framework, such that the embedding network can generalize to any individual user. Personalization is further achieved on top of the robustly learned representations in an adaptation procedure. We conducted extensive experiments based on two publicly available HAR datasets as well as a newly created HAR dataset. Results verify that Meta-HAR is effective at maintaining high test accuracies for individual users, including new users, and significantly outperforms several baselines, including *Federated Averaging*, *Reptile* and even centralized learning in certain cases. Our collected dataset will be open-sourced to facilitate future development in the field of sensor-based human activity recognition.

CCS CONCEPTS

• **Human-centered computing → Mobile computing**; • **Computing methodologies → Learning latent representations**; *Distributed artificial intelligence.*

KEYWORDS

Human Activity Recognition, Model-agnostic Meta-learning, Federated learning, Personalization, Representation Learning.

ACM Reference Format:
Chenglin Li, Di Niu, Bei Jiang, Xiao Zuo, and Jianming Yang. 2021. Meta-HAR: Federated Representation Learning for Human Activity Recognition. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3442381.3450006

1 INTRODUCTION

Human activity recognition (HAR) is the problem of recognizing human activity types based on mobile sensor data, playing an important role in ubiquitous and pervasive computing. State-of-the-art approaches rely on deep neural network models to replace traditional manual feature engineering and have greatly improved the accuracy of HAR [33, 34]. However, most existing solutions rely on centrally collected data, e.g., signal samples, including gyroscope and accelerometer time-series, collected from mobile users. Such labelled signal samples may contain private user information and may cause privacy concerns, according to data regulatory constraints that arise along with the widespread practice of data science, such as GDPR [1]. Due to this reason, to date few large-scale user activity dataset have been collected and made public, which hinders the development of HAR techniques.

Thanks to the rapid advancement of computational and storage capability on mobile devices, *Federated learning* (FL) has emerged as an alternative distributed learning framework, which aims to train machine learning models based on decentralized data scattered on mobile devices without collecting them. In federated learning, a global model is downloaded by each mobile device and updated with its local data, while the local updates are aggregated into a renewed global model iteratively. Similar to other mobile applications such as keyboard input prediction [9], HAR is another well-motivated scenario that can benefit from federated learning [28], which simplifies privacy management and gives each user a greater flexibility

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450006

on controlling which local data samples are selected and how they should contribute to the overall application improvement. Although Federated Learning has claimed decent performance in multiple tasks in the literature, e.g., image classification and keyboard input prediction [9, 12], it is still a question whether it can be used to solve HAR. Federated HAR has been tested on a simple deep neural network model with an accuracy reduction of up to 6% reported by [28], which is a significant degradation from centralized learning. We point out that there are three major obstacles to performing HAR with federate learning:

First, while federated learning is known to approximate centralized learning well if the training samples are independent and identically distributed (IID) across devices, such an IID assumption does not hold to activity signals. In fact, each user does not necessarily feature the same activity types—most users only perform a subset of all activities, e.g., one user may only have {*walking*, *driving*} recorded on his/her device, while another without any vehicle may only have {*walking*, *biking*}. Therefore, the local training datasets have unbalanced label distributions. Such a heterogeneity in label distribution across users can cause serious performance degradation to federated learning [37], and will exacerbate as the number of participating users increases.

Second, even for the same activity, two users may exhibit dramatically different signal distributions. The reason is because users may perform the same action in different styles. For example, two persons might have completely different walking patterns, one with large stride lengths, while the other with a high frequency and yet smaller step sizes. In other words, there exists a large degree of heterogeneity in the signal distributions of the same activity across users. We show through experiments that such heterogeneity in input signal distributions will also seriously affect the model accuracy achieved by federated learning.

Third, a single global model found by federated learning can hardly adapt and generalize to individual users, especially to a new user with his or her own activity characteristics. To handle real-world HAR tasks, a personalized model combining the insights jointly mined from all users with a predictor specifically fit to its local data is desired for each user. Such personalization is especially desirable for a new user or an existing user with newly introduced activity types.

To solve these challenges, in this paper we propose Meta-HAR, a *federated representation learning* framework for human activity recognition, where a shared, global deep embedding network is meta-trained by federated learning across users, while the signal representations given by the embedding network are fed into a separate classification network tailored to each device for personalized activity prediction. Such a representation learning framework is inspired by the fact that a good signal representation cannot be trained locally based on sparse data, but must take advantage of the abundant yet heterogeneous data residing on different devices, whereas the final personalized prediction at each user should augment the shared representation locally.

To train the shared embedding network, we treat each device (user) as a separate task, which possibly has its unique label distribution and input signal distributions. Inspired by model-agnostic meta-learning [7], we meta-train an embedding network that adapts to the distribution of the tasks instead of to each individual task, thus preserving the generalization ability to new tasks from this task distribution. Specifically, we adopt a federated version of *Reptile* [10], a first-order meta-learning algorithm [21], to train the embedding network. Each device iteratively updates the embedding network with its local dataset through a pairwise similarity loss and pushes the updated embedding network to the server for aggregation. By minimizing the pairwise loss instead of cross-entropy loss, samples from the same class are encouraged to cluster in the embedded space, while those from different classes are pushed apart. Since every pair of samples yields a loss value for optimization, we have effectively bootstrapped the sparse data on each device into a larger amount of training samples. We show that the shared embedding network trained with this method is robust to the heterogeneous data distributions across users.

Once a generalizable signal representation is acquired, activity types are predicted on a device through a two-stage adaptation procedure. We first fine-tune the embedding network on the local dataset of the device with pairwise loss. We then add a client-specific output layer on top of the embedding network for each user for activity classification, and fine-tune the embedding network and the output layer jointly on the same local data. We show that the embedding network can be sufficiently fine-tuned on the small local dataset, and that the personalized models can even outperform a centrally trained model sometimes, especially for new users, due to the effective local adaptation.

We have performed extensive evaluation of the proposed Meta-HAR on two publicly available datasets: 1) Heterogeneous Human Activity Recognition (HHAR) dataset [29] with 9 users and 6 activities, and 2) USC-HAD [36] dataset, with 14 users and 6 different activities, as well as on a much larger newly collected dataset[1] with 48 users and 6 different activities. Note that the two publicly available datasets were created in carefully controlled environments such that each user has all activity types and a balanced label distribution. To mimic the real-world scenario, for these two datasets, we randomly removed several activities from each user to simulate the case of Non-IID label distributions. In each experiment, we randomly left several users out which served as the *meta-test users* to test the generalizability of Meta-HAR to new users, and trained our model on the remaining *meta-train users* using the proposed method. For each meta-train user, a portion of its local data was also left out for testing. We repeated every experiment 5 times and averaged the results. We have achieved test accuracies of 92.5%, 98.39% and 91.07%, 93.79% for meta-test users and meta-train users on HHAR and USC-HAD datasets, respectively. On the larger collected dataset, similar test accuracies of 93.29% and 90.76% are observed on meta-test users and meta-train users, demonstrating the scaling capability of Meta-HAR. Results on these datasets suggest that Meta-HAR clearly outperforms FedReptile [10]. We also merged the two publicly available datasets to stress-test Meta-HAR under a significantly heterogeneous and unbalanced scenario with 23 users and 7 different activities in total (with 5 overlapping activities in both datasets). In this case, we show that the proposed Meta-HAR significantly outperforms FedReptile.

[1] We open source the collected dataset and all source code on Github: https://github.com/Chain123/Meta-HAR

2 PROBLEM AND MOTIVATION

Human activity recognition (HAR) aims to classify multitudes of sensor readings on mobile devices (e.g. signal segments from the gyroscope and accelerometer) into human activity types. In this section, we introduce the HAR problem in a federated learning setting where sensor data cannot be centrally collected, together with the new challenges that HAR has posed to federated learning.

Suppose there are n participating mobile devices (or users). Let the local dataset of user i be represented by $D_i = \{(s_{ij}, a_{ij}) | a_{ij} \in A^i, j = 1, 2, \ldots, N_i\}$, where s_{ij} is the sensor signal of the j-th sample in D_i, and a_{ij} is its corresponding activity label. $N_i = |D_i|$ represents the number of samples in D_i, while A^i denotes the set of activity types observed at user i. Notice that each user may have different activity types, i.e., A^i is different across users. The ultimate goal is to solve the activity recognition problem on each individual user.

A naive idea is to perform local supervised learning based on D_i for each device separately. However, the pitfall here is that from the perspective of each user, the scheme has failed to leverage the vast amount of data residing on other users. Moreover, since the activity types on each user are potentially sparse, a locally trained model can not make predictions about activities new to the user.

2.1 Challenges to Federated HAR

A seemingly plausible solution is federated learning, e.g., Federated Averaging (FedAvg) [18], which is able to learn a global model on decentralized datasets residing on mobile users. However, this scheme does not work well for the HAR problem, mainly due to the heterogeneity that exists in both label and input signal distributions. First, it is shown that in image classification, the heterogeneity in label distribution among users, which is also referred to as the issue of Non-IID and unbalanced data, would cause substantial performance degradation to federated learning [37]. Aside from Non-IID label distribution, in HAR, users have heterogeneous input signal distributions even for the same activity, which can also cause performance degradation, but has not been reported in literature.

Here, we demonstrate with experiments on HHAR dataset [29] that the heterogeneity in signal distributions alone can cause significant performance degradation. Note that in this dataset, each user has IID activity types (IID labels). We split the data for each user into a train set (80%) and a test set (20%). The neural network model used for HAR task is shown in Fig. 4 and will be introduced in detail in Section 3. Three schemes are evaluated:

- **Central**: Collecting all data on a server and train the HAR model centrally.
- **FedAvg-User**: *FedAvg* is applied to learn a global HAR model across users.
- **FedAvg-Shuffle**: *FedAvg* is applied to learn a global model, where all the samples are first collected and shuffled on a server then redistributed to all users.

Note that the original HHAR dataset is collected in a controlled situation thus it does not suffer from heterogeneity in label distribution (Non-IID or unbalance), thus the only difference between **FedAvg-User** and **FedAvg-Shuffle** is whether there is heterogeneity in signal distribution across users. The results are shown in Fig. 1, where the x-axis represents training epochs for **Central** approach and the federated update rounds for two **FedAvg** schemes. One can easily observe an accuracy reduction of **FedAvg-User** compared to **Central** from over 97% to below 80%. While, without heterogeneity in signal **FedAvg-Shuffle** can learn a model that has performance close to centrally trained model. In our experiment, the performance reduction from Central model to FedAvg approach is nearly 20%, much higher than that reported in previous work [28], which is 6%, due to the fact that [28] adopts generalizable handcrafted features and simple classification models, softmax regression and a DNN model. However, the performance of the state-of-the-art model for HAR under federated learning setting remains unknown and is worthy of further study.

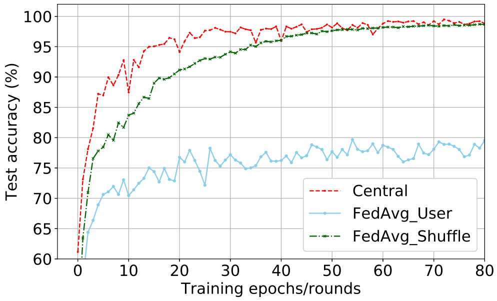
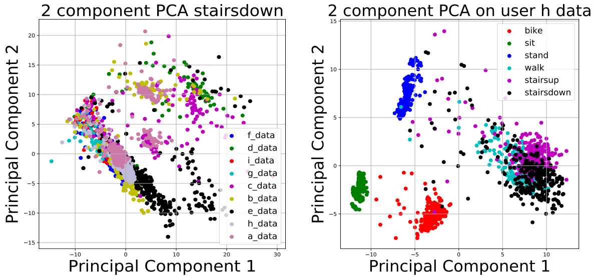


Figure 1: Experimental results that demonstrate the insufficiency of *FedAvg* on the HHAR dataset.



(a) "Stair-down" activity data from all 9 users (a-i) in HHAR dataset.

(b) Different activities from user *h*.

Figure 2: (a) Distribution of samples from activity "Stair-down" for all the users in HHAR dataset; (b) Distributions of samples from all activities of user *h*.

To further show the existence of heterogeneity among users even for the same activity, we extract traditional handcrafted features [28] (mean, standard deviation, maximum, minimum of signal amplitudes on each axis) and use PCA to visualize the sample distribution from HHAR dataset. The results are shown in Fig. 2, as we can see in Fig. 2(b), samples from different activities of user *h* are well clustered which shows the efficiency of the handcrafted features. However, samples for the same activity "Stair-down" is also clustered by different users as shown in Fig. 2(a), which demonstrates the heterogeneity in signal distribution across users.

Aside from the performance issues we discussed above, there are also practical factors that shows the disadvantages of federated learning for our problem. First, without collecting user data, it can be hard to know the global activity set, $A = A^1 \cup A^2 \ldots A^n$, n is the number of users, for the classification problem, which determines the output dimension of the global model. Furthermore, to train the model locally on user data, we need to unify the labels across different users, for example, we need to make sure label "0" represents the same activity across different users in federated HAR problem. This work is tedious and time-consuming. Finally, solving a classification problem with a tremendous number of classes is harder than a problem with fewer categories.

Therefore, a novel framework which is capable of overcoming the heterogeneity in both label and input signal distributions is desired for successful application of federated learning in human activity recognition.

3 METHODS

In this section we describe *Meta-HAR*, our proposed federated representation learning framework for solve the HAR problem without centrally collecting the data. Instead of training a global classification network for all activity types, Meta-HAR first learns a common deep representation model (or a signal embedding network) parameterized by Θ, through a model-agnostic meta-learning framework across all users. The goal of the embedding network is to embed any given input signal, regardless of its activity type, into a fixed length vector, which is fed into a classifier separately learned at each individual user to conform to its own activity set and output dimension. Such a design avoids the complexity to train a large global classifier for the global activity set and eliminates the need to unify labels across different users, as has been mentioned in Section 2.

In the following, we first provide an overview of the training procedure of the embedding network, followed by a description of the neural architectures of the embedding network adopted by Meta-HAR and its local training procedure. Finally, we present the procedures for model personalization and activity inference at the users. The overall workflow of Meta-HAR is shown in Algorithm 1.

3.1 Federated Representation Learning

We first introduce our method to meta-learn the embedding network in a federated manner in order to achieve a strong generalization ability while leveraging heterogeneous data. To introduce robustness in the presence of signal and label heterogeneity, the global embedding network is not trained by supervised classification tasks, but is meta-trained by pairwise comparison tasks, which compare whether two segments of signals belong to the same type of activities or not.

In federated HAR, although each user may have a local dataset with different activity types and signal distributions, yet the local task of activity classification is conceptually similar among different users. This is highly similar to the problem of optimization-based meta-learning [24], that attempts to learn an initialization of a model from multiple similar tasks, which can then be adapted to a new task at inference time through a fine-tuning stage, also known as *adaptation*. Model-Agnostic Meta-Learning (MAML) is an emerging approach to learning to learn, whose goal is to train a model on a variety of similar learning tasks so that it can solve new learning tasks with only a small number of training samples and training steps. Therefore, we treat the activity recognition problem at each individual user as a separate task. Each task has its unique input signal distributions, a different output activity set and even a different number of activity types. Yet, each task is assumed to be sampled from $p(\mathcal{T})$, a global distribution of tasks. According to optimization-based meta-learning, e.g., MAML [7, 21], it is possible to train a model through a proper optimization procedure, such that the model adapts to the underlying distribution of tasks $p(\mathcal{T})$ and thus can generalize to any new task sampled from $p(\mathcal{T})$.

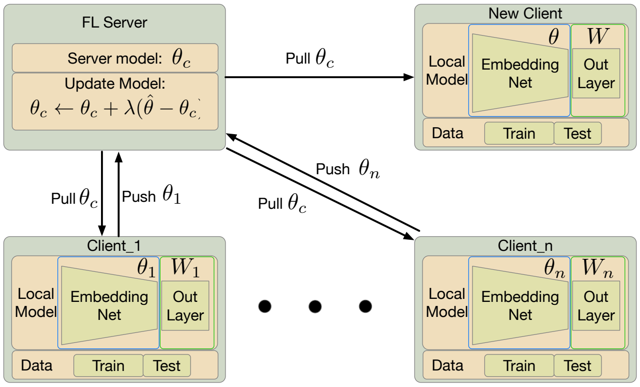


Figure 3: An overview of the Meta-HAR framework. In our system, there is a FL server, n meta-train users and a new user as meta-test user. Each user holds a local dataset (which is further divided into train and test parts), a classifier parameterized by $\{\Theta, W\}$ where Θ represents the parameters of the embedding network and W represents the output layer. Note that, only the embedding network Θ is meta-learned in a federated manner.

Motivated by [10] which points out that FedAvg is a special case of Reptile, a scalable first-order meta-learning algorithm, we adopt a federated version of Reptile [21], which we call Federated Reptile (FedReptile), to meta-train the embedding network Θ in a federated manner among decentralized users. The workflow of using FedReptile to train the embedding network, with parameters Θ, is depicted in Fig. 3, in which there are n users participating in training, a Federated Learning server (FL server) and a new user for testing of generalization ability. Each user holds a local dataset, pulls model parameters from the FL server, updates them using its local dataset, and then pushes those updates to the FL server. The FL server is responsible for coordinating the collaboration process of all users and updating the global model.

As shown in Algorithm 1, suppose n users are involved with local datasets $\{D_i\}, i = 1, 2, \ldots n$. In each round, a subset of users, U, is selected to update the global parameters. (According to [5], in each round of Federated Learning, only a subset of users will participate to avoid excessive waiting time in a distributed setting.) Every user $i \in U$ first pulls the current parameters Θ_c from the FL server and performs m epochs of local training before pushing the updated model Θ_i to the FL server. The FL server then averages the

Algorithm 1: The Meta-HAR Algorithm

```
Input: $n$ users with local train datasets $D = \{D_i\}, i = 1, \dots, n$.
    A FL server with initialized embedding network $\Theta_c$.
Output: Personalized HAR models for every user.
Meta-training:
for round = 1,2,3 ... do
    Randomly select a subset $U$ of users.
    for User j in U do
        Pull model parameters $\Theta_c$ from server.
        Train $m(\geq 1)$ epochs of the embedding network on
          local dataset $D_j$ through pairwise loss, get locally
          updated parameters $\Theta_j$.
        Push the updated parameters $\Theta_j$ to server.
    FL server update central model : $\Theta_c \leftarrow \Theta_c + \lambda(\hat{\Theta} - \Theta_c)$
      where $\hat{\Theta} = \frac{1}{|U|}\sum_{j \in U} \Theta_j$
Personalization:
for user j in all users do
    Pull parameters of embedding network $\Theta_c$ from FL server.
    Fine-tune $\Theta_c$ with pairwise loss on local dataset to obtain
      local embedding network $\Theta_j$.
    Further fine-tune local classifier $\{\Theta_j, W_j\}$ with
      cross-entropy loss on local dataset.
    Return personalized classification model $\{\Theta_j^t, W_j^t\}$ for user
      $j$.
```

collected updates Θ_i to update the global model Θ_c as follows:.

$$\hat{\Theta} = \frac{1}{|U|}\sum_{j \in U} \Theta_j, \tag{1}$$

$$\Theta_c = \Theta_c + \lambda(\hat{\Theta} - \Theta_c). \tag{2}$$

Finally, a personalization procedure is performed to obtain personalized models for all users, which we will discuss in detail in Section 3.3.

3.2 Embedding Network Architecture and Local Training

In this subsection, we briefly introduce the structure of the embedding network and how the it is locally updated by individual users.

Similar to prior deep neural networks for HAR (e.g.,[22, 34, 35]), our embedding network Θ involves the combined use of convolutional networks and recurrent networks. However, Θ is not locally updated by minimizing the local classification loss (cross-entropy loss). Instead, a weight-sharing siamese network is used to predict whether two signal samples are of the same class or not. We now describe the embedding network architecture and local training methods.

Embedding Network Architecture. Similar to previous deep learning methods, we preprocess the raw readings from sensors before feeding them into a neural network. We first compute the amplitude series to serve as an additional input dimension for each sensor. For example, readings from motion sensor **Gyroscope** s_g has three dimensions $s_g = \{s_{gx}, s_{gy}, s_{gz}\}$. The additional amplitude axis is defined as $\sqrt{s_{gx}^2 + s_{gy}^2 + s_{gz}^2}$. To model sequential dependencies of signal series, we split the sensor data into k time intervals, each of width τ. After extending the amplitude axis and segmenting the sequential signals, we apply a Fourier transformation to each axis of each segmented data block, extracting the frequency domain representation. Finally, we stack all the outputs of Fourier transformation, magnitudes and their corresponding frequencies, into a tensor of shape $k \times 2(d_g + 1) \times f$, where d_g is the dimensionality of the sensor s_g, which is 3 in our example, and f is the length of frequency domain representations. The result is then fed to the embedding network Θ.

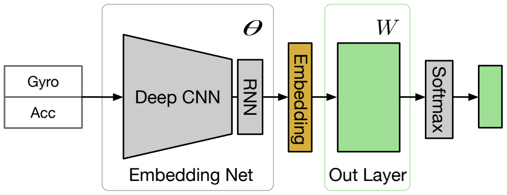


Figure 4: Deep classifier structure for HAR. The classifier consists of an embedding net and a fully connected output layer parameterized by Θ and W, respectively.

Similar to DeepSense [34], the embedding network leverages both CNN and RNN to process sensor readings. As shown in Fig. 4, multiple convolutional layers are applied to the processed sensor signals (from the Gyroscope and Accelerometer) to model spatial relevance among different axises of the same sensor as well as relevance across sensors. To be specific, each of the k data blocks of a given input sample, we first apply convolution to the stacked pair of magnitudes and frequencies along each axis of each sensor signal. Then, another convolutional layer is applied to all axes within the same sensor. Finally, we fuse the convolution outputs of different sensors through the last convolution layer. Then two Long Short-Term Memory (LSTM) layers are used to extract temporal relevance of the k CNN outputs and output a fixed length embedding vector.

Thereafter, shown in Fig. 4, by adding a fully connected output layer on top of the embedding network, we can build a classifier to get the predicted probabilities for each category with a *Softmax* function.

Training with Pairwise Loss. There are two loss functions that can be used to train the embedding network locally on the user side: *cross-entropy* loss and *pairwise* loss. Categorical cross-entropy loss, defined as:

$$H(p, q) = -\sum_{j=1}^{M} p_j log(q_j), \tag{3}$$

where M represents the dimensionality of the two discrete probability distributions, measures the distance between a proposed probability distribution q and target distribution p, is often used for multi-class classification problems. In our setting, the local HAR task on each user is a multi-class classification problem with the output dimension of $|A^i|$ for user i. However, as mentioned in Section 2, different users have different local activity set. To leverage cross-entropy loss under the federated learning setting, one need to figure out the global activity set and unify the labels across different users so that local models have the same output layer W, which is time-consuming and hard to scale to new activity types.

Pairwise loss, on the other hand, does not require local users to be aware of the global activity set, thus is flexible and scalable to new activity types. Furthermore, pairwise loss encourages the clustering of sample embeddings in a real space, bootstraps the training set and makes the embedding network more robust to heterogeneous inputs. Well clustered embeddings can make the subsequent classification task much easier. Specifically, for a given pair of input samples $\{(e_i, a_i), (e_j, a_j)\}$ where e_i, e_j are the embeddings output from embedding network for sample i, j and a_i, a_j are their corresponding labels. First, we use the cosine distance, defined as

$$\varphi_{ij} = \frac{e_i^T e_j}{|e_i| \cdot |e_j|} \tag{4}$$

to measures the similarity between any two embedding vectors. Then, we the pairwise loss for sample pair (i, j) is defined as

$$l_{i,j} = -\delta(a_i, a_j)log(\sigma(\varphi_{ij})) - (1 - \delta(a_i, a_j))log(1 - \sigma(\varphi_{ij})) \tag{5}$$

where $\sigma(x) = 1/(1 + e^{-kx})$ is the logistic sigmoid function with a tunable parameter k, $\delta(a_i, a_j) = 1$ if $a_i = a_j$, otherwise $\delta(a_i, a_j) = 0$. Obviously, by minimizing the $l_{i,j}$, the cosine similarity φ_{ij} between the embedded vectors e_i and e_j will reach the maximum if they are from the same class, i.e., $a_i = a_j$, and φ_{ij} will reach the minimum if they are from different classes, i.e., $a_i \neq a_j$.

To fit the model on a given dataset, we can sample a batch of B pairwise samples, each in the form of (s_i, s_j, a_i, a_j) to perform batched learning. For each training sample (s_i, s_j, a_i, a_j), a weight-sharing siamese network is used to get the embeddings of the two input signals at the same time, that is, $e_i = \Theta(s_i)$ and $e_j = \Theta(s_j)$. Then the parameters of embedding network Θ will be updated by error back-propagation according to (5).

3.3 Personalized Classification

Finally, to solve the activity recognition problem on each user i, we introduce a personalized output layer parameterized by W_i, which together with the global embedding network Θ forms a classification model that conforms to the local output dimension and activity set at user i. However, the meta-learned global embedding network Θ_c, when transferred to user i, only serves as an initialization of the embedding network to be used by user i.

For each user i, we use a two-step fine-tuning strategy to adapt the global embedding network Θ_c into its personalized local classifier. First, the embedding network Θ_c pulled from the server is fine-tuned on the local training set of user i to obtain Θ_i. The fine-tuning of the embedding network is also performed with the pairwise loss given by Equation 5. The embedding network fine-tuned this way is able to encourage clustering of embedded signal representations based on the local dataset with user-specific activity types.

The output of the local embedding network is then fed into a fully connected layer for activity classification. The weight parameters of the output layer for user i are represented by W_i with shape $(|E| \times |A_i|)$ where $|E|$ is the size of the output embedding vector, i.e., the dimension of the embedded space, while A_i is the activity set of user i. Then, the output layer parameters W_i together with the embedding network are fine-tuned with the cross-entropy loss. That is, in the second stage of fine-tuning, back-propagation is applied to $\{\Theta_i, W_i\}$ to minimize the classification error.

By leveraging a user-specific output layer, we have simplified the local classification by reducing its expected number of output categories in contrast to the global activity recognition problem for all users. On the other hand, as the embedding network is collaboratively learned across all users on abundant data and on all activity types, local classifiers built on top of the embedding network are capable of dealing with activities that have seldom seen by the user before. In other words, a well learned signal representation network has reduced the demand on the number of local samples required to fine-tune the local classifier.

4 EXPERIMENTS

In this section, extensive experiments on three datasets, two public and one collected, are conducted to evaluate our proposed *Meta-HAR* framework. Comparisons with several alternative baselines are also included to demonstrate the effectiveness of our method.

4.1 Datasets

In this paper, two widely used public datasets: the Heterogeneous Human activity recognition (HHAR) dataset [29] with 9 users and 6 activities: {*Standing*, *Sitting*, *Walking*, *Upstairs*, *Downstairs* and *Biking*}, and the USC-HAD [36] dataset, with 14 users and 6 different activities {*Standing*, *Sitting*, *Walking*, *Upstairs*, *Downstairs* and *Running* } are adopted. Preprocessing is applied to maintain input consistency between the two public datasets and reformat the original sensor readings to fit our proposed architecture. First, we down-sample sensor readings in the HHAR dataset to a frequency of 25Hz. Signals in USC-HAD are down-sampled to 50Hz. Second, the long consecutive sensor data (up to 5 minutes in HHAR dataset) are segmented into several short samples, signals in HHAR dataset are segmented into 6-second samples, while every 3-second sensor readings in the USC-HAD dataset form a sample. This way, samples from different datasets would have the same input length.

Originally, users in the two public datasets have balanced samples for all activities. To mimic the real world, for each user we randomly remove 0 to 2 activities from its local dataset to simulate the scenario where datasets follow Non-IID distributions across users. Thereafter, the dataset has heterogeneity in both label and signal distributions. Finally, we merged these two public datasets to form an even more heterogeneous dataset with 23 users and 7 activities in total to stress-test the proposed method in comparison with baselines.

Aside from the two public datasets, we collected a new, larger HAR dataset involving 48 users and 6 types of activities, including {*Walking*, *Biking*, (walking) *Upstairs*, (walking) *Downstairs*, *Running* and *Taking Bus/Taxi* }. Our dataset was collected through an Android app specifically developed for activity signal logging. Participants can use any Android smartphone to choose an activity type, after which their signal samples are logged by the app while performing the activity. Furthermore, there are no constraints or control on how users perform a certain activity. Therefore, our dataset is inherently more noisy and has more heterogeneity in terms of activity types, hardware devices, the position of the phone on the body, signal distributions, and sampling frequencies, etc., than prior datasets collected in controlled environments. Details on this collected dataset of 48 users are given in the Appendix A.

4.2 Experiment Setups

To begin with, we split all the users in a dataset into **Meta-train users**, which participate in the meta-learning process, and **Meta-test users**, which served as new users for testing the generalization ability of the meta-learned model. To be specific, on the HHAR or USC-HAR dataset, we randomly select *one user* as the meta-test user. On the merged dataset, *two users* are randomly selected as meta-test users from HHAR and USC-HAD datasets, respectively. For the collected dataset, *five users* are selected as meta-test users. Our ultimate goal is to learn a personalized model for all users, meta-train and meta-test users, that can solve the local human activity recognition problem. To train the model and test the performance, we further split the local dataset of each user into a train set (80%) and a test set (20%). Specifically, the following schemes are evaluated to demonstrate the effectiveness of our proposed method:

- **Central**: The HAR classification model is trained on all data samples collected on a server.
- **FedAvg**: The original Federated Averaging method [18], where a global, shared classification model is learned in a federated manner.
- **FedReptile**: The Federated Reptile [10] is applied to first meta-learn a global initialization of a classification model. Then personalization is achieved by fine-tuning the classification model with cross-entropy loss on each user.
- **Meta-HAR**: The proposed *Meta-HAR* framework.
- **Meta-HAR-CE**: A variant of the proposed *Meta-HAR*, where the embedding network is trained with a cross-entropy loss instead of pairwise loss.

Note that compared to **FedAvg**, **FedReptile** simply adds extra adaptation steps to generate personalized models. However, different from previous works, only the embedding network is federated learned in our proposed method.

Furthermore, the following fine-tuning strategies for personalization are evaluated and compared:

- **Separated**: The parameters of the embedding network and the output layer are fine-tuned independently. Specifically, the embedding network is first fine-tuned with pairwise loss, then we fix the embedding network to further fine-tune the output layer with cross-entropy loss.
- **Merged**: The embedding network and the user-specific output layer are jointly fine-tuned with cross-entropy loss.
- **Two-stage**: The proposed two-stage fine-tuning strategy.

Note that the FedAvg [18] scheme is well designed to overcome systematic challenges such as communication efficiency and limited computational power on mobile devices, etc. Our proposed framework follows the same system design as FedAvg, i.e, we have similar system efficiency and communication cost as FedAvg does. In this work, our goal is to solve the data heterogeneity problem posed by federated HAR, therefore, instead of addressing the communication challenges, we focus on the model performance for the HAR problem. The performance of a model is evaluated with the averaged prediction accuracy which is defined as follows:

$$Acc = \frac{1}{\sum_{i=1}^{n} m_i} \sum_{i=1}^{n} m_i \cdot acc_i, \tag{6}$$

where acc_i is the test accuracy on the ith user and m_i is the number of test samples on that user. The averaged accuracies on both meta-train and meta-test users are used to evaluate the generalization ability to future data of users who participated in the meta-learning as well as to new users who have not participated in the embedding network training.

In each experiment, we randomly select the meta-test and meta-train users to evaluate the performance for the proposed method and all baselines. We repeat the whole process 5 times on each dataset. The mean accuracies and their standard deviations are presented.

4.3 Implementation Details

Our models are implemented using Pytorch with python 3.6. All experiments are carried out on Tesla P40 GPUs with memory size of 22.38 GiB and 1.53 GHz memoryClock-Rate. ADAM optimizer[11] with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 1^{-8}$ is used to update all network parameters. We use $k = 10$ for the sigmoid function $\sigma(x) = 1/(1 + e^{-kx})$ to calculate pairwise loss defined in Equation (5). In federated learning procedure, we set $\lambda = 1.0$ and perform $m = 2$ epochs of local training at each update round.

Our human activity recognition model consists of multiple 1-D and 2-D convolutional layers all with 64 filters and two layers of LSTM layers. The size of the latent vector is set to be 100. We adopt dropout to prevent over-fitting in the training stage and the dropout rate is set to be 0.3. As the number of samples residing on the mobile devices of users is usually small, we adopt a batch with size 64 for local training.

4.4 Experimental results

Comparison between *Central*, *FedAvg*, *Meta-HAR* and *FedReptile*. Table 1 compares our proposed method with baseline approaches on the three datasets: two public datasets and one collected dataset. The results on the merged dataset are given in table 2. Comparing the *Central* model with all other federated learning-based methods, we can see there is a great performance degradation from *Central* to *FedAvg* in terms of the activity prediction accuracy on all datasets. On the other hand, *Meta-HAR* and *FedReptile* can achieve comparable performance on meta-train users and are even able to outperform the *Central* model by a great margin on meta-test users. This demonstrates the superior of personalized models over a single federated learned global model. The advantage of the personalized models, generated by *Meta-HAR* and *FedReptile*, over the single global model, learned through *FedAvg*, can be contributed to the adaptation of global model to the local datasets. After fine-tuning, personalized models on the user side can be better fitted to the distributions of the corresponding local datasets.

Notice that on the USC-HAD dataset, there is a 5.33% performance reduction from the `Central` model to the best-personalized models on meta-train users. This is because, on the USC-HAD dataset, each user only holds a small local dataset with only a few samples, therefore, it is hard to get a personalized model with a

Table 1: Test Results on HHAR, USC-HAD and collected Datasets. The number in the parenthesis denotes the fine-tuning epochs performed, e.g. Meta-HAR(x) means Meta-HAR with x epochs of fine-tuning on local datasets. All numbers are in percentage (%).

Algorithms	HHAR Dataset		USC-HAD Dataset		Collected Dataset	
	Meta-train user	Meta-test user	Meta-train user	Meta-test user	Meta-train user	Meta-test user
Central	98.55 ± 0.11	83.14 ± 8.40	99.31 ± 0.14	81.63 ± 10.51	90.18 ± 0.14	79.84 ± 0.41
FedAvg	79.56 ± 0.62	66.79 ± 1.84	84.44 ± 0.26	80.24 ± 1.54	69.19 ± 5.02	64.29 ± 6.86
FedReptile(1)	87.16 ± 0.29	86.00 ± 1.66	87.96 ± 0.21	85.33 ± 2.20	88.29 ± 0.66	83.96 ± 2.31
FedReptile(2)	92.64 ± 0.26	88.04 ± 2.65	91.02 ± 0.30	86.44 ± 3.05	90.84 ± 0.18	89.59 ± 1.29
FedReptile(3)	95.70 ± 0.25	91.84 ± 1.85	**93.98 ± 0.31**	89.17 ± 2.26	**91.49 ± 0.31**	92.30 ± 1.52
Meta-HAR(1)	98.32 ± 0.06	85.23 ± 1.80	92.01 ± 0.16	83.59 ± 2.28	89.16 ± 0.76	92.38 ± 0.43
Meta-HAR(2)	98.36 ± 0.04	91.25 ± 1.82	92.54 ± 0.11	90.19 ± 2.81	90.24 ± 0.63	**93.33 ± 0.80**
Meta-HAR(3)	**98.39 ± 0.02**	**92.50 ± 1.26**	93.79 ± 0.14	**91.07 ± 1.74**	90.76 ± 0.67	93.29 ± 1.03

performance comparable to the *Central* model. The insufficiency of local datasets on the mobile devices of users further motivates the need for federated learning where we can leverage all the data samples scattered on mobile devices.

Comparing the *Meta-HAR* model with the *FedReptile* approach, we can see these two methods achieved decent and close performance after fine-tuning on local datasets. *FedReptile* can sometimes even be a slightly better than *Meta-HAR* on meta-train users, e.g. on USC-HAD dataset. This is reasonable due to two factors: first, there are only a few heterogeneities in label and signal distributions on the simple public datasets, particularly on the USC-HAD dataset. Second, in meta-train users, all parameters including the last output layer of the local classification models are federated learned, i.e. they have better initial weights when performing personalization. However, on complex datasets with more heterogeneities and meta-test users, *Meta-HAR* can significantly outperform *FedReptile*, e.g. on the merged dataset, the *Meta-HAR* model achieves an accuracy of 95.35% greatly outperform *FedReptile* which gives an accuracy of 70.65% on meta-train users. On meta-test users, *Meta-HAR* outperforms *FedReptile* on all datasets, for example, on the merged dataset, *Meta-HAR* achieves an averaged accuracy of 75.83% and 90.23% on meta-test users from HHAR and USC-HAD datasets, respectively, greatly outperform the accuracy achieved by *FedReptile*, which is 69.4% and 74.83%. This demonstrates our model's capability of handling datasets with high heterogeneity and can be easily adapted to new (meta-test) users. The advantages of *Meta-HAR* over *FedReptile* can be contributed to two main reasons. First, in our framework, only the embedding network is federated learned with pairwise loss, which is more robust to heterogeneity in both labels and signals. Second, with pairwise loss, we can boost the local dataset and improve the learning efficiency, which is suitable for a small dataset residing on user's mobile devices.

Comparison between *Meta-HAR* and *Meta-HAR-CE*. Comparison between our proposed method, *Meta-HAR*, and its variant, *Meta-HAR-CE*, is performed as an ablation test to further demonstrate the advantage we get by adopting pairwise loss to meta-train the embedding network. Show in table 2, one can see, for most of the time, *Meta-HAR* outperforms *Meta-HAR-CE* with a great margin on meta-test users. On meta-train users, it achieved a slightly worse accuracy than *Meta-HAR-CE* did. This makes sense, cause in the meta-training procedure of *Meta-HAR-CE*, all parameters of the local classifier on meta-train users are already well trained on the local dataset. The higher performance achieved by *Meta-HAR* over *Meta-HAR-CE* on the meta-test user demonstrates the superior generalization ability of *Meta-HAR*. Furthermore, mentioned in Section 3.2, *Meta-HAR* is scalable to new activity types in federated learning setting while *Meta-HAR-CE* can not due to fixed output dimensionality of cross-entropy loss. These advantages of *Meta-HAR* over *Meta-HAR-CE* can all be contributed to the adoption of pairwise loss.

Table 2: Test Results on Merged Dataset for FedAvg, FedReptile, Meta-HAR-CE and Meta-HAR methods. The number in the parenthesis denotes the fine-tuning epochs performed. All numbers are in percentage (%).

Algorithms	Meta-train	Meta-test (H)	Meta-test (U)
FedAvg	48.97 ± 0.62	39.99 ± 1.56	52.71 ± 1.97
FedReptile(1)	58.12 ± 0.55	59.52 ± 1.87	66.16 ± 2.52
FedReptile(2)	65.83 ± 0.65	66.35 ± 3.67	71.95 ± 2.72
FedReptile(3)	70.65 ± 0.53	69.40 ± 2.64	74.83 ± 2.19
Meta-HAR-CE(1)	94.05 ± 0.41	62.69 ± 1.90	61.37 ± 3.62
Meta-HAR-CE(2)	97.01 ± 0.36	62.74 ± 3.00	71.12 ± 2.52
Meta-HAR-CE(3)	**97.70 ± 0.24**	67.19 ± 3.47	80.96 ± 2.95
Meta-HAR(1)	91.42 ± 0.49	47.28 ± 1.65	73.98 ± 2.68
Meta-HAR(2)	93.64 ± 0.39	54.38 ± 2.47	87.00 ± 2.88
Meta-HAR(3)	95.35 ± 0.28	**75.83 ± 2.27**	**90.23 ± 2.44**

Evaluation of different fine-tune strategies. Finally, we evaluate several fine-tuning strategies, to show the effectiveness of our proposed two-stage adaptation approach. Evaluations are done on the merged dataset. Results of proposed two-stage method, *Merged* and *Separated* approaches are presented in Table 3. All the strategies can achieve decent test accuracies on meta-train users, however, our two-stage fine-tuning method significantly outperforms other baseline approaches on the meta-test users by a great margin. *Merged*

Table 3: Test Results of Meta-HAR with different fine-tune methods on Merged Dataset. The number in the parenthesis denotes the fine-tuning epochs performed. All numbers are in percentage (%).

Tune methods	Meta-train	Meta-test (H)	Meta-test (U)
Merged(1)	89.03 ± 0.83	47.60 ± 2.07	60.76 ± 3.62
Merged(2)	91.84 ± 0.62	48.47 ± 2.36	69.79 ± 2.62
Merged(3)	**95.38 ± 0.47**	50.38 ± 1.59	80.19 ± 3.39
Separated(1)	88.75 ± 0.65	47.22 ± 2.54	51.98 ± 1.79
Separated(2)	90.47 ± 0.69	48.69 ± 2.85	64.44 ± 1.16
Separated(3)	91.91 ± 0.71	50.98 ± 2.97	73.35 ± 1.72
Two-stage(3)	95.35 ± 0.28	**75.83 ± 2.27**	**90.23 ± 2.44**

and *Separated* methods fine-tune both the embedding network and output layer the same way as we do in the two-stage method. However, the improvement is insignificant and thus cannot be used for fast adaptation to new users. The *Merged* method achieved better performance than *Separated* did which shows the advantage by jointly fine-tuning the embedding network and output layer.

5 RELATED WORK

HAR with deep learning model. There are several recent studies leveraging deep neural network models to different mobile sensing or HAR applications. Deep Boltzmann Machine is adopted in DeepEar [14] to improve the performance of audio sensing tasks in an environment with background noise. DeepX [13] and RedEye [17] reduce the energy consumption of deep neural networks, based on software and hardware, respectively. IDNet [8] uses CNNs for the biometric gait analysis. RBM [4] and MultiRBM [23] combine deep Boltzmann Machine and Multimodal DBMs to boost performance of human activity recognition task. Deepsense [34] applied RNN on top of CNN to acquire the sequential information of the input sensor data.

Federated optimization. Federated learning [12] aims to train a high-quality centralized model with data defining the optimization problem being unevenly distributed over large number of nodes. This data decentralization can help reduce data transmission and protect user privacy. Most of the recent studies focus on addressing the communication efficiency challenge faced by federated learning. FederatedAveraging [18], which combines local stochastic gradient descent (SGD) on client nodes with model averaging on the server side, is able to reduce communication rounds between clients and server. Bonawitz et al. [6] allows a server to sum up large vectors from mobile devices in a secure manner through a communication-efficient, failure-robust protocol. Differential privacy [19] achieves user-level privacy protection for the federated averaging algorithm. McMahan et al. [18], Sattler et al. [25] also show the robustness of federated learning algorithm when clients hold Non-IID data. Zhao et al. [37] proposed to improve FederatedAveraging on Non-IID data by sharing a small subset of client data on the server side. However, sharing local samples is infeasible for real world federated learning application where the number of clients is extremely large and the data on client's side may update frequently, for example the log data of APPs installed on smart phones. Sozinov et al. [28] first applied federated to the problem of human activity recognition. However, it simply applied FedAvg to HAR without trying to solve the new challenges posed by federated HAR. In this paper, apart from Non-IID and unbalanced distribution in label, we pointed out that there is also heterogeneity in signal distribution can cause performance degradation in Federated Learning. We propose to leverage model personalization with MAML algorithm on a global embedding network to addressing the challenges face by federated HAR.

Smith et al. [27] propose a Federated Multi-Task learning framework MOCHA to solve the general multi-task learning problem, which is a plausible solution to our problem. However, they focus on solving high communication cost, stragglers, and fault tolerance for distributed multi-task learning and did not discuss the case of heterogeneous input signal distribution and can not adapt to new arriving users due to the fact that there is no global shared representation model in MTL setting. FAVOR [32] uses reinforcement learning to achieve intelligent device selection to improve federated learning performance and overcome Non-IID data distribution over participants. However, it can not provide personalized models for all participant users.

Meta-learning. Meta-learning aims to solve the problem of learning to learn [20, 30]. Early works focus on the design of meta-trainers, i.e., a model that learns how to train another model such that better performance can be achieved on a given task [3, 26]. Andrychowicz et al. [2] adopts deep neural networks to train a meta-learner and proposes an optimizer-optimizee setup, where each component is learned with an iterative gradient-descent procedure. Li and Malik [15] follows a guided policy search strategy and automatically learns the optimization procedure for updating a model. Model-agnostic meta-learning (MAML) [7] is another popular approach that does not impose a constraint on the architecture of the learner. Ravi and Larochelle [24] proposes an LSTM meta-learner to learn an optimization procedure for few-shot image classification. Li et al. [16] develops an SGD-like meta-learning process and also experiment on few-shot regression and reinforcement learning problems. Reptile [21], i.e., the approach adopted in this paper, simplifies the learning process of MAML by conducting first-order gradient updates on the meta-learner. Jiang et al. [10] interpreted federated learning as a MAML algorithm and implement a federated version of the first-order MAML algorithm, Reptile. However, Jiang et al. [10] focus on the parameter tuning to get a global model which is readily to personalize.

Learn personal models. Several algorithms have been developed for training personalized model on decentralized peer-to-peer network [31]. Users with local datasets collaborate with each other through peer-to-peer exchange in the network in order to learn personalized model. Personalized model is convenient for local users, due to the fact that the newly generated data on the mobile devices are often consumed locally on that device for many applications, for example, users' interaction log with APP. However, the peer-to-peer network is unrealistic for the federated learning setting where the number of clients if extremely large and the relationships between clients are complicated and even dynamic. Jiang et al. [10] combined federated learning with MAML algorithm and shown with numerical results the benefits by adopting personalization for

Federated Learning. However, previous only made incremental contribution to the original Federated Learning framework by simply adding a personalization step. On the other hand, our framework is more flexible, personal models can even have different output dimension for different clients.

6 CONCLUSIONS

In this paper, we study the federated human activity recognition problem, which aims to train accurate personalized activity classification models for mobile users or devices, without centrally collecting their sensor data. The Non-IID activity type distribution across users as well as the heterogeneity in signal distribution across different users have posed significant challenges to federated learning for HAR. We propose a federated representation learning framework for HAR, namely *Meta-HAR*, to to meta-learn an embedding network in a federated manner, leveraging the heterogeneous yet abundant datasets residing on distributed mobile devices. A personalized model with user-specific output layer can then be obtained for each user through an adaptation strategy on top of the meta-learned embedding net.

Extensive experiments on one collected dataset, two publicly available datasets and their merged dataset are conducted. Our approach significantly outperformed all baselines methods with a great margin. For example, on the merged dataset we outperform the baseline FedReptile method by 24.7%, 6.43% and 15.4% on meta-train users, HHAR meta-test users and USC meta-test users, respectively. We also make our newly collected dataset publicly available to facilitate future development in sensor-based human activity recognition.

REFERENCES

[1] Jan Philipp Albrecht. 2016. How the GDPR will change the world. *Eur. Data Prot. L. Rev.* 2 (2016), 287.

[2] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. 2016. Learning to learn by gradient descent by gradient descent. In *Advances in neural information processing systems*. 3981–3989.

[3] Samy Bengio, Yoshua Bengio, Jocelyn Cloutier, and Jan Gecsei. 1992. On the optimization of a synaptic learning rule. In *Preprints Conf. Optimality in Artificial and Biological Neural Networks*. Univ. of Texas, 6–8.

[4] Sourav Bhattacharya and Nicholas D Lane. 2016. From smart to deep: Robust activity recognition on smartwatches using deep learning. In *Pervasive Computing and Communication Workshops (PerCom Workshops), 2016 IEEE International Conference on*. IEEE, 1–6.

[5] 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).

[6] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1175–1191.

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 1126–1135.

[8] Matteo Gadaleta and Michele Rossi. 2016. Idnet: Smartphone-based gait recognition with convolutional neural networks. *arXiv preprint arXiv:1606.03238* (2016).

[9] Andrew Hard, Kanishka Rao, Rajiv Mathews, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. 2018. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604* (2018).

[10] Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. 2019. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488* (2019).

[11] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

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

[13] Nicholas D Lane, Sourav Bhattacharya, Petko Georgiev, Claudio Forlivesi, Lei Jiao, Lorena Qendro, and Fahim Kawsar. 2016. Deepx: A software accelerator for low-power deep learning inference on mobile devices. In *Proceedings of the 15th International Conference on Information Processing in Sensor Networks*. IEEE Press, 23.

[14] Nicholas D Lane, Petko Georgiev, and Lorena Qendro. 2015. DeepEar: robust smartphone audio sensing in unconstrained acoustic environments using deep learning. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 283–294.

[15] Ke Li and Jitendra Malik. 2016. Learning to optimize. *arXiv preprint arXiv:1606.01885* (2016).

[16] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. 2017. Meta-SGD: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835* (2017).

[17] Robert LiKamWa, Yunhui Hou, Julian Gao, Mia Polansky, and Lin Zhong. 2016. RedEye: analog ConvNet image sensor architecture for continuous mobile vision. In *ACM SIGARCH Computer Architecture News*, Vol. 44. IEEE Press, 255–266.

[18] 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).

[19] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2017. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963* (2017).

[20] Devang K Naik and RJ Mammone. 1992. Meta-neural networks that learn by learning. In *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, Vol. 1. IEEE, 437–442.

[21] Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999* (2018).

[22] Henry Friday Nweke, Ying Wah Teh, Mohammed Ali Al-Garadi, and Uzoma Rita Alo. 2018. Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges. *Expert Systems with Applications* 105 (2018), 233–261.

[23] Valentin Radu, Nicholas D Lane, Sourav Bhattacharya, Cecilia Mascolo, Mahesh K Marina, and Fahim Kawsar. 2016. Towards multimodal deep learning for activity recognition on mobile devices. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. ACM, 185–188.

[24] Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning. (2016).

[25] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. 2019. Robust and communication-efficient federated learning from non-iid data. *arXiv preprint arXiv:1903.02891* (2019).

[26] Jürgen Schmidhuber. 1992. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation* 4, 1 (1992), 131–139.

[27] 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.

[28] 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.

[29] 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*. ACM, 127–140.

[30] Sebastian Thrun and Lorien Pratt. 2012. *Learning to learn*. Springer Science & Business Media.

[31] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. 2017. Decentralized collaborative learning of personalized models over networks.

[32] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. 2020. Optimizing Federated Learning on Non-IID Data with Reinforcement Learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 1698–1707.

[33] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. 2019. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters* 119 (2019), 3–11.

[34] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. 2017. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 351–360.

[35] Shuochao Yao, Yiran Zhao, Aston Zhang, Lu Su, and Tarek Abdelzaher. 2017. Deepiot: Compressing deep neural network structures for sensing systems with a compressor-critic framework. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. ACM, 4.

[36] Mi Zhang and Alexander A Sawchuk. 2012. USC-HAD: a daily activity dataset for ubiquitous activity recognition using wearable sensors. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. ACM, 1036–1043.

[37] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. 2018. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582* (2018).

A COLLECTED DATASET

We developed an Android app specifically designed for human activity sensor signal logging in a real-world scenario. The screenshot for the developed app is shown in Fig. 5, there are 9 types of activities that users can choose to perform, however, finally, we got enough data samples for only 6 of them: {*Walking*, *Biking*, *Upstairs*, *Downstairs*, *Running* and *Taking Bus*}. The published dataset is the same as we used in this paper. Every participant needs to go through the following steps to upload their sensor signals.

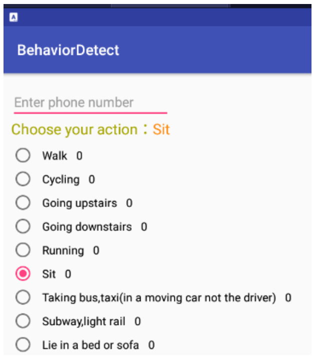
- Enter your phone numbers which served as your ID.
- Choose an activity that you are going to perform and press the "start" bottom.
- Performing corresponding activity until you are done.
- Press "Stop" (the "Start" button becomes the "Stop" button after you pressed start.) The app will automatically upload collected sensor signals to a pre-assigned server.

During data collection, every 7-second consecutive signals will be saved as one data sample. Normally, users wouldn't perform the corresponding activity immediately right after they pressed start. Therefore, we discard the signals of the first 5 seconds and the last 7 seconds to avoid bad samples and reduce sample noise.

The sampling rate is set to be 25Hz, however, due to the heterogeneity among hardware devices and operating systems (customized Android systems and different versions of the Android system), we can not let the collected samples from different users be exactly 25 Hz. Therefore, we only kept samples with a sequence length between [150, 200], which is a frequency between $[21.4Hz, 28.6Hz]$, and discard all samples that do not fall in this section. Finally, we select the first 150 data points of each sample so that we have the same input data shape across all users.

Note that, unlike previous work, where the sensor signals of human activity are collected in a carefully designed and well-controlled environment [29, 36]. In our case, the users are totally free when they performing a certain activity. Therefore, our dataset is inherently noisier with more heterogeneities in terms of local activity types, hardware devices, local data unbalances, the position of the phone on the body, signal distribution, and sampling rate. To make sure, users are performing the correct chosen activity, we require users to upload a short video recording their surroundings and action to avoid cheating.

For each participant, any of its activity type with uploaded "good" samples less than 20 will be removed. After that, We further remove users with a number of local activity types less than 3. Finally, we get 48 users each with a different number of activities as well as different activity types. As shown in Fig. 6(a), after sample and user selection, we get a heterogeneity dataset with Non-IID label distributions among users. Most users have 5 types of local activity and none of them provides all the 6 types of activities we selected. Also, our collected dataset is highly biased to activities that are common in real life, e.g. "Walking" and "Taking bus/taxi". On the other hand, we can only collect a relatively small number of samples for activities that are not commonly performed in daily life, such as "Upstairs", "Downstairs" and "Running".



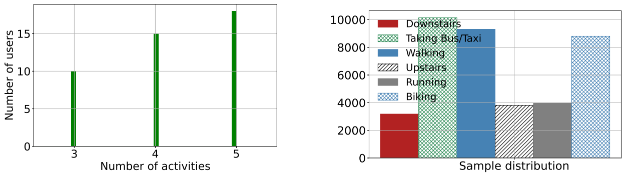
BehaviorDetect

Enter phone number

Choose your action：Sit

○ Walk 0
○ Cycling 0
○ Going upstairs 0
○ Going downstairs 0
○ Running 0
● Sit 0
○ Taking bus,taxi(in a moving car not the driver) 0
○ Subway,light rail 0
○ Lie in a bed or sofa 0

Figure 5: Screen shot of our developed Android app for sensor data logging. The number right after each action type represents the number of samples that have been collected and uploaded.



(a) Histogram for the number of activity types of local datasets

(b) Sample distributions of collected dataset.

Figure 6: (a) Distribution of samples from activity "Stair-down" for all the users in HHAR dataset; (b) Distributions of samples from all activities of user *h*.

In our experiments on the collected dataset, we do not remove the unbalancedness and directly train our model on the unbalanced dataset. This is because, in a real-world federated learning setting, we can not control the dataset distribution on the user side. However, to evaluate the performance of a given model, we use a balanced test dataset, where the numbers of data samples from each activity type is balanced.