Federated Clustering and Semi-Supervised Learning: A New Partnership for Personalized Human Activity Recognition

Riccardo Presotto[a,*], Gabriele Civitarese[a], Claudio Bettini[a]

[a] *Università degli studi di Milano, Dipartimento di Informatica, Via Celoria, 18, 20133, Milan, Italy*

Abstract

Federated Learning (FL) is currently studied by several research groups as a promising paradigm for sensor-based Human Activity Recognition (HAR) to mitigate the privacy and scalability issues of classic centralised approaches. However, in the HAR domain, data is non-independently and identically distributed (non-IID), and personalization is one of the major challenges. Federated Clustering has been recently proposed to mitigate this issue by creating specialized global models for groups of similar users. While this approach significantly improves personalization, it assumes that labeled data are available on each client. In this work, we propose SS-FedCLAR, a novel HAR framework that combines Federated Clustering and Semi-Supervised learning. In SS-FedCLAR , each client uses a combination of active learning and label propagation to provide pseudo labels to a large amount of unlabeled data, which is then used to collaboratively train a Federated Clustering model. We evaluated SS-FedCLAR on two well-known public datasets, showing that it outperforms existing semi-supervised FL solutions while reaching recognition rates similar to fully-supervised FL approaches.

Keywords: human activity recognition, federated learning, clustering, semi-supervised learning

1. Introduction

The continuous evolution of sensing technologies and intelligent systems makes it possible to design advanced methods for sensor-based Human Activity Recognition (HAR). HAR has several applications in important domains, like healthcare and well-being [1]. Currently, supervised learning methods, and in particular the ones based on Deep Learning (DL), have the potential of reaching the most accurate recognition rates [2]. These approaches rely on labeled datasets to learn the low-level correlations between the sensor data stream (e.g., inertial sensors data) and the activities actually performed by the subject (e.g., walking, jogging, sitting). While many research groups proposed effective solutions in the last decades, there are still several

[*] Corresponding author

Email addresses: `riccardo.presotto@unimi` (Riccardo Presotto), `gabriele.civitarese@unimi.it` (Gabriele Civitarese), `claudio.bettini@unimi.it` (Claudio Bettini) open research problems in HAR that require further investigation [3]. These problems still limit the deployment of HAR systems in large-scale real-world scenarios.

For instance, classic centralised models trained involving many subjects suffer from privacy and scalability issues. From the privacy point of view, activity data is sensitive since it may reveal personal habits or health conditions [4]. Considering scalability, in large-scale scenarios centralized approaches may incur high communication latency and computational costs [5]. The Federated Learning (FL) paradigm represents a promising direction [6] to mitigate these issues. FL shifts the training burden from the cloud to trusted edge devices closer to the user. Each edge device (e.g., a smartphone or home gateway) locally trains a model with available labeled data. Then, only the weights of the resulting model are transmitted to the cloud server, thus mitigating privacy and scalability issues in large-scale scenarios.

In the last few years, the FL paradigm has been studied in the HAR domain [7, 8, 9, 10] with solutions reaching recognition rates close to the ones of centralized approaches [11]. Even though FL is a promising direction toward real-world HAR, there are still some limitations. A major issue is that the FL global model should generalize over a large number of users. However, different users may perform activities with significantly different patterns depending on their physical characteristics, age, and habits. Indeed, data from different users is non-independently and identically distributed (non-IID). Hence, a trade-off between generalization and personalization is required to build accurate FL-based HAR models [12].

In a recent work, we proposed *FedCLAR* to tackle this issue [13]. *FedCLAR* uses Federated Clustering to generate (server-side) specialized global models for groups of similar users. In particular, the local models received by the server are clustered and merged considering the similarity of their weights. While *FedCLAR* significantly improves personalization, it requires that labeled data is available on each client. However, the collection of a large annotated dataset is prohibitive: it is time-consuming, costly, and intrusive. In another recent work, we proposed *FedAR*: a solution for FL-based HAR to mitigate the labeled data scarcity problem [14]. *FedAR* uses a combination of active learning and label propagation. However, it is not based on Federated Clustering to mitigate the non-IID problem.

In this work, we combine *FedCLAR* and *FedAR* by proposing SS-FedCLAR: a novel Semi-Supervised Federated Clustering method for Personalized Sensor-Based Human Activity Recognition. The integration of those two methods is not trivial, since it involves creating clusters with a limited amount of labeled data. We evaluated SS-FedCLAR on two public datasets proposing a novel evaluation methodology. The results show that SS-FedCLAR outperforms *FedAR* (both in terms of higher F1 score and a reduced number of active learning queries) while reaching results close to *FedCLAR* (and other fully supervised FL solutions) using a limited amount of labeled data.

The contributions of this paper are the following:

- We propose a novel combination of Federated Clustering and Semi-Supervised learning for sensor-based HAR. Our method mitigates both the non-IID and the data scarcity problems that are typical in FL-based HAR.
- We provide a formalization of the Semi-Supervised Federated Clustering problem.
- Our evaluation performed on two public datasets shows that SS-FedCLAR outperforms state-of-the-art FL semi-supervised HAR approaches in terms of F1 score, while requiring significantly less active learning queries. Moreover, our method reaches recognition rates close to fully supervised approaches.

2. Related work

Sensor-based HAR is a well-known research area, that has been deeply studied by several research groups in the last two decades [1]. The goal of HAR is to classify the current activity performed by the user analyzing the continuous stream of sensor data. Sensor-based HAR has been explored mainly considering two settings: using sensors of mobile/wearable devices [15], and using environmental sensors of smart-home environments [16]. Among the many proposed approaches, the most effective ones are based on Deep Learning (DL) [17]. However, the deployment of HAR systems in real-world scenarios is still limited by several open challenges. Among the many challenges, there are labeled data scarcity [18], the discovery of new activities [19], the transferability of activity models between different environments/users [20], and lifelong learning [21]. In this work, we focus on personalization and data scarcity issues related to large scale scenarios.

Fully supervised HAR approaches require a large amount of labeled data to train an accurate model. However, different users may perform the same activities with different patterns. Moreover, distinct activities may be performed with similar motion patterns. Collecting annotated datasets is costly, time-consuming, intrusive, and hence prohibitive on a large scale [22]. In the following, we summarize the main methodologies that have been proposed in the literature to mitigate the data scarcity problem for data-driven HAR. Unsupervised learning has been proposed to derive significant activity clusters from a large amount of unlabeled data [23, 24]. However, a certain amount of labels are still required to reliably annotate each cluster with the correct activity. Moreover, since distinct activities sometimes share similar patterns, purely unsupervised approaches still represent an open challenge. Data augmentation has been often adopted to mitigate the data scarcity problem in the presence of imbalanced datasets [25, 26]. With respect to our method, data augmentation is an orthogonal method that could be used to further improve the activity model. Recently, GAN models have been proposed to perform data augmentation with more realistic synthetic data [27, 28]. However, GANs require to be trained with a significant amount of data. Transfer learning has been effectively applied in the HAR domain to fine-tune models learned from a source domain (where labeled data are available) to a target domain (where target data are poorly available)[29, 30]. Semi-supervised methods learning approaches represent a promising direction in data scarcity scenarios [31, 32]. Semi-supervised approaches for HAR (e.g., self-learning, label propagation, co-learning, and active learning) only use a restricted labeled dataset to initialize the activity model, while a significant amount of unlabeled data is semi-automatically annotated. However, such approaches are based on centralized models that suffer from scalability and privacy issues.

In general, to mitigate the issues of centralized approaches, Federated Learning (FL) has been recently applied to sensor-based HAR to distribute the training of the activity recognition model among the participating devices [10, 5, 8, 33]. In FL, each client trains a local model with available labeled data and it only transmits the local model weights to the server. However, due to the above mentioned intra- and inter-variability in activity execution, activity data is non-IID (non identically and independently distributed). Since FL is mainly based on local optimization, the FL approaches based on simple server-side aggregation strategies are not accurate in the presence of non-IID data.

In order to mitigate this issue, transfer learning methods can be used on the client-side to fine-tune the global model, thus improving personalization [34, 11, 35]. However, since the proposed works consider a single global model, balancing personalization and generalization is still a challenge. Multi-task federated learning [36] mitigates the non-IID problem by collaboratively learning only the common features, while personalized layers are trained at the client side. The main drawback of these approaches is that they consider a convex objective function, which is not suitable for complex HAR models based on deep learning. Recently, a few works proposed solutions based on Federated Clustering [37, 38, 13]. The goal of Federated Clustering is to create specialized global models (server-side) by grouping users that perform activities in a similar way. Even though Federated Clustering is a promising direction, existing works ignore the above-mentioned data scarcity problem.

Semi-supervised federated learning solutions for HAR have been only partially explored. The existing works mainly focus on unsupervised methods to collaboratively learn (based on the FL setting) the feature representation from a large unlabeled stream of sensor data. A limited amount of labeled data is then used to train a classifier based on this representation [39, 40]. However, those works do not consider model personalization and they do not propose approaches to continuously obtain new labeled data from each user. FedHAR [41] is a semi-supervised approach that takes advantage of a small number of clients with labeled data and a large number of clients with unlabeled data by using a customized loss function based on an unsupervised gradients aggregation strategy. Differently from FedHAR, SS-FedCLAR does not assume the existence of clients with full availability of labeled data, and it also proposes a practical solution to continuously improve the federated model by combining active learning and label propagation [14].

To the best of our knowledge, SS-FedCLAR is the first method that combines Federated Clustering and Semi-Supervised Learning to mitigate both the non-IID and data scarcity problems.

3. Problem formulation

3.1. Non-IID problem in HAR

Let $U = \{U_1, \ldots, U_n\}$ be the set of users. Each user U_i is associated with a labeled dataset $D_i = \{(x, y)\}$, where x is a data point and y the corresponding activity label. Let $D = \{D_1, \ldots, D_n\}$ be the set of datasets, each one corresponding to a user in U. D is non-independently and identically distributed (non-IID) if at least a pair of datasets $D_i, D_j \in D$ satisfies one of the following conditions [42]:

- **Feature distribution skew**: $P_{D_i}(x) \neq P_{D_j}(x)$. This inequality between probability distributions is true when the data samples in D_i have a significantly different marginal distribution than the ones in D_j. In HAR, this often happens since each subject may perform activities in a peculiar way. Among many factors, users' physical characteristics have a strong impact on their activity patterns. For instance, a young subject would probably have a faster walking pattern than an elderly subject.

- **Label distribution skew**: $P_{D_i}(y) \neq P_{D_j}(y)$. This inequality between probability distributions is true when the labels in D_i have a significantly different marginal distribution than the ones in D_j. In HAR, this usually happens since different users may have different daily routines. For example, a sporty subject would likely spend more time *running* or *cycling* than a sedentary subject.

- **Quantity distribution skew**: This condition is true when jD_ij and jD_jj are significantly different. In HAR, is not unusual to have significantly different sizes of labeled samples for different subjects.

3.2. Why non-IID is a problem in a FL setting

Given a non-IID set of datasets D, a standard *centralized* ML approach builds a recognition model M^C by using all the annotated data points in $D^* = D_1 \bigcup D_2 ... \bigcup D_n$. In this case, the training phase consists in finding the parameters $\mathbf{w}\ 2\ \mathsf{R}^{\mathbf{d}}$ that minimize a global objective function $f(\mathbf{w})$:

$$\min_{\mathbf{w} \in \mathsf{R}^{\mathbf{d}}} f(\mathbf{w}),\ f(\mathbf{w}) := \frac{1}{jD^*j} \sum_{k=1}^{|D^*|} \ell_k(\mathbf{w}) \tag{1}$$

where $\ell_k(\mathbf{w})$ is a loss function. Intuitively, the objective is to find the parameters $\mathbf{w}$ that minimize the average loss over all the annotated samples in D^*. By considering all the annotated samples at the same time, this *centralized* approach mitigates the non-IID problem.

However, there are significant differences in an FL setting. Indeed, each user U_i locally trains a model M_i, and it transmits to the server only the M_i parameters $\mathbf{w_i}$, along with the size of the personal labeled dataset jD_ij. The server is in charge of building a global model $\overline{M}$ from the local parameters $\mathbf{W} = h\mathbf{w_1}, \ldots, \mathbf{w_n}i$, and it is not possible to directly access D^*. The objective function $\overline{f}(\overline{w})$ of the federated model to derive the global parameters $\overline{w}$ is the following:

$$\min_{\overline{\mathbf{w}} \in \mathsf{R}^d} \overline{f}(\overline{\mathbf{w}}),\ \overline{f}(\overline{\mathbf{w}}) = \sum_{i=1}^{n} \frac{jD_ij}{jD^*j} f_i(\mathbf{w_i}) \tag{2}$$

where $f_i(\mathbf{w_i})$ is the local objective function that each user U_i minimizes by using D_i to obtain $\mathbf{w_i}$:

$$\min_{\mathbf{w_i} \in \mathsf{R}^d} f_i(\mathbf{w_i}),\ f_i(\mathbf{w_i}) := \frac{1}{jD_ij} \sum_{k=1}^{|D_i|} \ell_k(\mathbf{w_i}) \tag{3}$$

Ideally, the parameters of the *federated* model should approximate the ones of the *centralized* model. However, in a non-IID setting, the overall data distribution of D^* (that is captured by the *centralized* approach) may be considerably different from the distribution of each $D_i\ 2\ D$ that is captured by the *federated* approach. For this reason, minimizing $\overline{f}(\mathbf{w})$ may lead to a global model that would significantly underperform the one derived by minimizing $f(\mathbf{w})$.

3.3. The federated clustering problem

A possible solution to tackle the non-IID problem in the FL setting issue is to partition U into s clusters $C = C_1, ..., C_s$ so that each cluster minimizes the non-IID properties among the datasets of the users assigned to the same cluster. Hence, it is possible to derive a federated model

$\overline{M}^{C_j}$ for each cluster. The objective function $\overline{f}^{C_j}(\overline{\mathbf{w}}^{C_j})$ of each model M^{C_j} can be optimized by using data from the cluster:

$$\min_{\mathbf{w}^{C_j} \in \mathrm{R}^d} \overline{f}^{C_j}(\overline{\mathbf{w}}^{C_j}),\ \overline{f}^{C_j}(\overline{\mathbf{w}}^{C_j}) = \sum_{i=1}^{|C^j|} \frac{jD_ij}{jD^{C_j}j} f_i(\mathbf{w_i}) \tag{4}$$

where D^{C_j} is the set of datasets of the users belonging to the cluster C_j. If the clusters actually capture the similarity between the distributions of the datasets, the resulting model would better approximate the one generated by a *centralized* approach on the users of the same cluster.

However, in the FL setting it is not possible to access each D_i to compute the clusters, since only the model parameters $\mathbf{w_i}$ are available. Hence, a major problem that we tackle in this work is how to compute user clustering in the FL setting.

3.4. Data scarcity assumptions

In this work, we assume that the users do not actually have labeled datasets, but that they can only observe a stream of unlabeled sensor data. Let $S_i = fx_1, x_2, \ldots g$ be the stream of unlabeled data points observed by U_i. We also assume that, given a data point x, a user U can sometimes provide feedback about the corresponding activity. Finally, we assume the existence of a small pre-training labeled dataset D^{pt} that can be used to initialize the federated model. Note that the data points in D^{pt} are not collected from the users in U.

The problem tackled in this work is to compute user clustering under the above-mentioned assumptions.

4. SS-FedCLAR: combining Federated Clustering and Semi-Supervised Learning

4.1. SS-FedCLAR's overview

We assume that each user that participates in SS-FedCLAR has a personal trusted device (e.g., a smartphone, a smartwatch, a smart-home gateway) that we will refer as *client*. The client is in charge of collecting sensor data and running the client-side SS-FedCLAR's algorithms.

At the very beginning, SS-FedCLAR uses a classic FL solution that is based on a single global model. Since we assume limited availability of labeled data, this model is initialized using a small labeled dataset called *pre-training dataset*. Considering real-world deployments, the *pre-training dataset* may be one or more public datasets, or a small training set specifically collected by appropriately rewarded volunteers. Since we assume that the clients do not have personal labeled datasets to train their local models, SS-FedCLAR relies on a semi-supervised learning strategy that analyes the periodic classifier's outputs to provide pseudo-labels to the stream of unlabeled data. The newly labeled data are then used to train the local model during FL global model updates. At the end of each FL training process, each client also uses a transfer learning method to further fine-tune the global model.

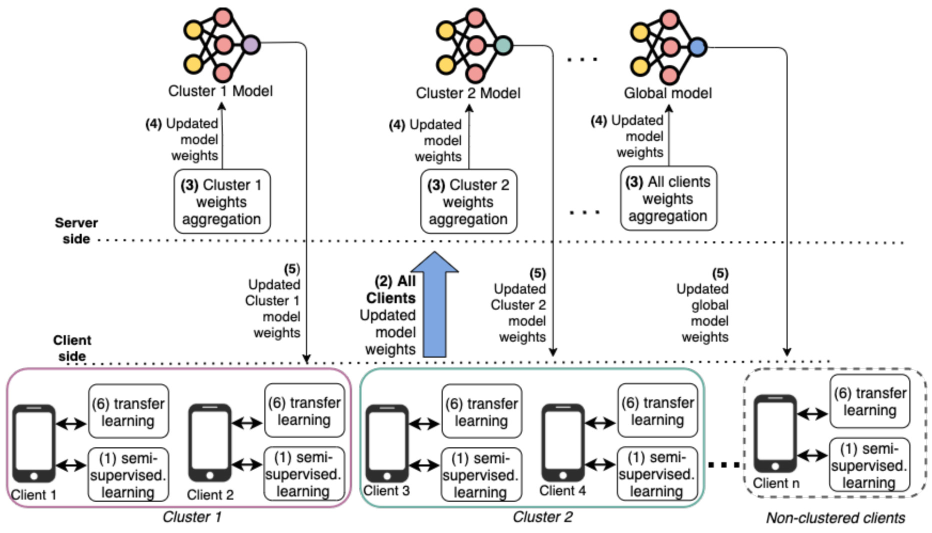

Cluster 1 Model
Cluster 2 Model
Global model
(4) Updated model weights
(4) Updated model weights
(4) Updated model weights
(3) Cluster 1 weights aggregation
(3) Cluster 2 weights aggregation
(3) All clients weights aggregation
Server side
Client side
(5) Updated Cluster 1 model weights
(2) All Clients Updated model weights
(5) Updated Cluster 2 model weights
(5) Updated global model weights
(6) transfer learning
(1) semi-supervised. learning
Client 1
Client 2
Client 3
Client 4
Client n
Cluster 1
Cluster 2
Non-clustered clients

Figure 1: Overall architecture of SS-FedCLAR

Periodically, during global model updates, SS-FedCLAR uses a server-side Federated Clustering algorithm (explained in detail in Section 4.2.2) to group the clients in clusters based on the similarity of their local model updates, and to incrementally compute a specialized model for each cluster.

Since some clients may not belong to clusters due to peculiarity in their activity execution, SS-FedCLAR also considers *non-clustered clients*. Federated Clustering is transparent to clients, that are not aware if they belong to a cluster.

After clusters are finalized, the server uses a standard aggregation method to update each specialized cluster models using the local model updates received by the clients of that cluster. The server also maintains a classic global model using the updates of all clients, including *non-clustered clients* that will be the ones receiving this model during the updates. The overall architecture of SS-FedCLAR is summarized in Figure 1.

4.2. Server Side: Federated Clustering

We first describe the tasks performed by the server side component of SS-FedCLAR.

4.2.1. Computing similarity between users

SS-FedCLAR adopts a server-side clustering approach to create specialized global models for groups of similar users. In general, clustering methods rely on a similarity metric that is computed on each pair of items that may be clustered. In sensor-based HAR, similar users are those that share similar sensor data patterns (i.e., similar activity patterns). However, in an FL learning process, only the weights of the local models are available, and not sensor data. Nonetheless, if two local models share similar weights, they were likely trained with similar patterns of data. Hence, given the parameter vectors $\mathbf{w_i}$ and $\mathbf{w_j}$ of the models corresponding to the users U_i and U_j, it is possible to compute their similarity. SS-FedCLAR relies on the cosine similarity since it proved to be effective for federated clustering [43]. The cosine similarity between the model weights of two users U_i and U_j can be computed as follows:

$$sim(\mathbf{w_i}, \mathbf{w_j}) = \frac{\mathbf{w_i} \quad \mathbf{w_j}}{\|\mathbf{w_i}\|\|\mathbf{w_j}\|} \tag{5}$$

However, considering HAR models and the recent results on transfer learning [11], we realized that computing the similarity by taking into account the whole parameter vector would not be the optimal choice. Considering local models based on deep learning, the closest layers to the input reflect high-level features that are common among all the subjects [44]. On the contrary, the layers that are close to the output are the ones that encode user-specific activity patterns.

Let $pers(\mathbf{w})$ be a function that extracts from parameter vector $\mathbf{w}$ the user-specific parameters. Hence, SS-FedCLAR computes the pairwise similarity between model weights as follows:

$$sim(\mathbf{w_i}, \mathbf{w_j}) = \frac{pers(\mathbf{w_i}) \quad pers(\mathbf{w_j})}{\|pers(\mathbf{w_i})\|\|pers(\mathbf{w_j})\|} \tag{6}$$

Since SS-FedCLAR is based on deep learning, the function $pers(\mathbf{w})$ returns the weights corresponding to the last l layers of $\mathbf{w}$.

4.2.2. Hierarchical Clustering

Using the similarity function described above, the cloud server in SS-FedCLAR can apply a clustering algorithm to derive groups of users that perform activities in a similar way. In this work, we use a hierarchical approach, since in the literature it proved to be effective for federated clustering [42].

The pseudo-code for the hierarchical clustering method of SS-FedCLAR is described in Algorithm 1. The intuition behind this process is the following. Initially, there is a cluster for each user. Then, clusters are grouped based on the pairwise similarity of the participating users and a clustering threshold ct. When two clusters are merged into a single one, a new specialized model for that cluster is generated by merging the models of the merged clusters. The process is repeated until no more clusters can be merged (i.e., there is no pair of clusters such that the similarity of their specialized models is higher than ct). The users in the singleton clusters are considered as *non-clustered clients*.

Algorithm 1 HierarchicalClustering

```
Input: $\mathbf{W} = \{\mathbf{w_1}, \ldots, \mathbf{w_n}\}$
Output: A set of clusters $C$, a set of specialized models $\mathbf{W^C}$
 1: $\mathbf{W^C}$    $\mathbf{W}$
 2: $C$    $\{\{U_1\}, \ldots, \{U_n\}\}$
 3: $cmap$    empty map from model weights to clusters
 4: $cmap[\mathbf{w_1}]$    $\{U_1\}$
 5: ...
 6: $cmap[\mathbf{w_n}]$    $\{U_n\}$
 7: do
 8:     $P$    pairwise similarity matrix on $\mathbf{W^C}$ based on $sim()$
 9:     $\mathbf{w_a}, \mathbf{w_b}$    $\arg\min_{\mathbf{w_a}, \mathbf{w_b} | a \neq b} P_{ab}$
10:     if $sim(\mathbf{w_a}, \mathbf{w_b})$    $ct$ then
11:         $\mathbf{w_{ab}}$    merge $\mathbf{w_a}$ and $\mathbf{w_b}$ using FedAvg
12:         $C_a$    $cmap[\mathbf{w_a}]$
13:         $C_b$    $cmap[\mathbf{w_b}]$
14:         $C_{ab}$    $C_a \cup C_b$
15:         $cmap[\mathbf{w_{ab}}]$    $C_{ab}$
16:         $C$    $C \setminus C_a$
17:         $C$    $C \setminus C_b$
18:         $C$    $C \cup C_{ab}$
19:         $\mathbf{W^C}$    $\mathbf{W^C} \setminus \mathbf{w_a}$
20:         $\mathbf{W^C}$    $\mathbf{W^C} \setminus \mathbf{w_b}$
21:         $\mathbf{W^C}$    $\mathbf{W^C} \cup \mathbf{w_{ab}}$
22:     else
23:         $\mathbf{W^C}$    $\{\mathbf{w} \in \mathbf{W^C}$ such that $|cmap(\mathbf{w})| > 1\}$
24:         $C$    $\{C_j \in C$ such that $|C_j| > 1\}$
25:         return $C$ and $\mathbf{W^C}$
26:     end if
27: while True
```

4.2.3. Model Update in SS-FedCLAR

The model update mechanism of SS-FedCLAR is described by Algorithm 2. Periodically (e.g., every night), the server requires an update of the models. Hence, a sequence of communication rounds is started. Each client locally trains its model and transmits the resulting weights to the server. Upon receiving the weights from the clients, the first task of the server is generating an overall global model using FedAvg.

If required, during the update the server computes clusters and specialized models as described before. Note that computing the similarity between users is effective only if performed after a certain number of communication rounds. Otherwise, we experimentally observed the risk of considering models parameters that are not sufficiently trained, thus generating unreliable clusters. For this reason, in SS-FedCLAR, our hierarchical clustering method explained in Section 4.2.2 is performed after a predefined number r of communication rounds. From the communication round r, the server will use the local model updates received from the clients to update the specialized models, while the ones received from the *non-clustered clients* are used to update the overall global model. Note that, in order to provide to *non-clustered clients* a global model with sufficient generalization capabilities, also the local models from clustered clients are used to updating the overall global model.

Based on preliminary experiments, we observed that it may be necessary to update the cluster model only when the set of participating users and/or their local data distribution significantly change. For instance, considering changes in local data distributions, each client may locally run algorithms to detect significant changes in patterns and habits. When a client detects such a change, it may notify the server. The clustering update may be performed only when a significant number of clients have notified the server. Considering changes in the set of users, a clustering update may be required only when there are several join and abandon events. A possible approach for updating clusters is that the cloud server stores every intermediate model computed during clustering (i.e., the dendograms associated with intermediate steps of hierarchical clustering). Hence, when needed, the server can recompute an optimal set of clusters by reversing some of the clustering steps and evaluating the new situation. A similar approach was proposed in [43]. However, a deeper investigation of clustering updates strategies is out of the scope of this paper.

4.3. Client Side: Semi-Supervised Learning

In SS-FedCLAR, the clients do not have a labeled dataset to train the local model. In order to overcome this problem, we proposed a semi-supervised strategy inspired by our previous work [14]. In particular, each client semi-automatically provides pseudo-labels to the unlabeled data stream by combining *Active Learning* and *Label Propagation.*

4.3.1. Classification and active learning

The main idea of active learning is to require the user to provide a feedback only to those data samples where the classifier shows uncertainty in activity classification. Indeed, the data samples associated with low classification confidence would have the most impact on improving the activity model if their label were available. Figure 2 depicts the overall data flow of classification and active learning.

SS-FedCLAR relies on a state-of-the-art non-parametric active learning approach called

Algorithm 2 SS-FedCLAR - Server Side Model Update

```
$C$   nil
$\mathbf{W}^C$   nil
for each periodic update (e.g., every night) do
    for each communication round $i$ do
        receive $\mathbf{W}{=}f\mathbf{w_1},..,\mathbf{w_n}g$ from clients
        $\mathbf{w^G}$   FedAVG using $\mathbf{W}$ (global model)
        if $i < r$ and $\mathbf{W}^C$ == nil then
            send $\mathbf{w^G}$ to each client
        else
            if $i == r$ then
                if $C$==nil then
                    $C, \mathbf{W}^C$   $HierarchicalClustering(\mathbf{W})$
                else if Clustering structure requires to be updated then
                    $C, \mathbf{W}^C$   $UpdateClusters(C,\mathbf{W})$
                end if
            else
                for $C_j$ 2 $C$ do /*for each cluster*/
                    $\mathbf{w}_{C_j}$   FedAVG using $\mathbf{w_i}$ 2 $\mathbf{W}$ from clients in $C_j$
                end for
            end if
            for $C_j$ 2 $C$ do
                send $\mathbf{w}_{C_j}$ to each client in $C_j$
            end for
            send $\mathbf{w^G}$ to *non-clustered* clients
        end if
    end for
end for
```

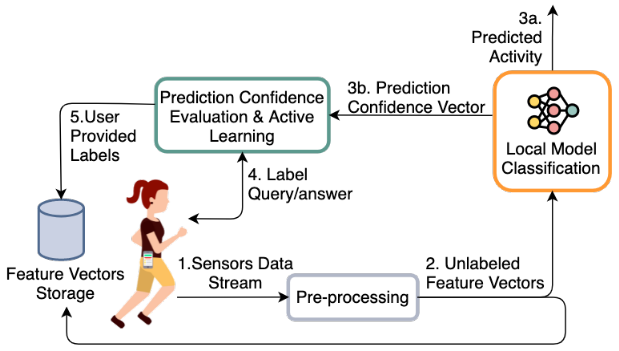

3a. Predicted Activity
Prediction Confidence Evaluation & Active Learning
3b. Prediction Confidence Vector
5.User Provided Labels
Local Model Classification
4. Label Query/answer
Feature Vectors Storage
1.Sensors Data Stream
Pre-processing
2. Unlabeled Feature Vectors

Figure 2: Classification and active learning data flow

VAR-UNCERTAINTY [45]. This approach is based on a dynamic threshold θ 2 $[0, 1]$ on the confidence of the local model when classifying unlabeled data samples. When the confidence is lower than θ, an active learning process is started by asking the user the ground truth about the current activity. The feedback and the queried data samples are then stored in the *Feature Vectors Storage*. In order to reduce the number of questions, the threshold θ is decreased when the user's feedback actually corresponds to the activity classified by the model. Otherwise, the threshold is increased. The unlabeled data samples for which an active learning query was not needed are stored in the *Feature Vectors Storage* without a label. The overall pseudo-code of classification and active learning is described in Algorithm 3.

Algorithm 3 Client side - Classification and active learning

```
1:  $lm$ ( local model
2:  for each feature vector $fv$ computed in real-time from sensor data do
3:      $\vec{p}$ ( probability distribution over the activities predicted by $lm$ on $fv$
4:      output the most likely activity according to $\vec{p}$
5:      if feedback is needed according to VAR-UNCERTAINTY [45] then
6:          $l$ ( activity label from the user
7:          add $(fv, l)$ to the *Feature Vectors Storage*
8:          update threshold of VAR-UNCERTAINTY [45]
9:      else
10:         add $(fv,\ )$ to the *Feature Vectors Storage*        ▷ unlabeled data point
11:     end if
12: end for
```

4.3.2. Label Propagation and Model Update

In the following, we describe what happens on each client of SS-FedCLAR when a model update is required from the server, as also depicted in Figure 3. During a model update, the

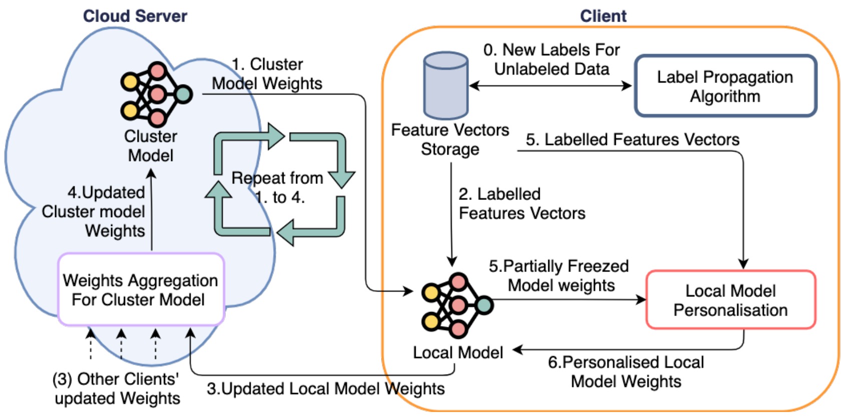

Cloud Server
Client
1. Cluster Model Weights
Cluster Model
Repeat from 1. to 4.
4.Updated Cluster model Weights
Weights Aggregation For Cluster Model
(3) Other Clients' updated Weights
3.Updated Local Model Weights
0. New Labels For Unlabeled Data
Label Propagation Algorithm
Feature Vectors Storage
5. Labelled Features Vectors
2. Labelled Features Vectors
5.Partially Freezed Model weights
Local Model Personalisation
Local Model
6.Personalised Local Model Weights

Figure 3: Local models training and personalized model update

clients' first step is to start a Label Propagation process in charge of expanding the amount of labeled data. Besides training the local model with more labeled examples, Label Propagation has the major advantage of further reducing active learning queries, and hence interactions with the user. Given a set of labeled and unlabeled data points, the Label Propagation algorithm automatically spreads pseudo-labels to a portion of unlabeled data [46]. Intuitively, data points that are close in the feature space likely correspond to the same class label. In this work, we use the Label Propagation method proposed in [14] (based on the Radial Basis Function (RBF) Kernel and euclidean distance) to update the *Feature Vectors Storage* with new pseudo-labels for a portion of unlabeled data points that did not trigger active learning queries.

After Label Propagation, the actual model update is started: for each communication round, the clients trains their local model with the available labeled data samples in the *Feature Vectors Storage*. Then, the resulting weights are transmitted to the server[1]. Finally, the clients receive the updated global model from the server. Each client is not aware if it is receiving updated weights from specialized cluster models or from a generic global model.

Even though Federated Clustering mitigates the non-IID problem, it is still possible that distinct users in the same cluster exhibit peculiar execution of some activities. Indeed, while some users may be grouped in the same cluster because they similarly perform the majority of the activities, they still may exhibit slight differences over a restricted number of activities. In order to further personalize the recognition model, SS-FedCLAR also relies on a transfer learning method for each client, inspired by solutions that proved to be effective in FL-based HAR [11]. In particular, the last p layers (i.e., the closest to the output) of the local model are fine-tuned using the *Feature Vectors Storage*, while the remaining ones are left as received by the server.

The client-side model update process of SS-FedCLAR is summarized is by Algorithm 4.

Algorithm 4 Client side - Model Update

```
1: $lm$    local model
2: Update the Feature Vectors Storage using the Label Propagation algorithm [14]
3: for each communication round do
4:     train $lm$ using available labeled data in the Feature Vectors Storage
5:     send the weights $\mathbf{w}$ of $lm$ to the server
6:     receive updated model $\mathbf{w}^{\mathbf{S}}$ from the server
7:     replace the weights of $lm$ with $\mathbf{w}^{\mathbf{S}}$
8: end for
9: freeze the layers of $lm$ except for the last $p$ layers
10: train $lm$ using available labeled data
11: unfreeze $lm$ layers
```

[1]Each client also transmits the number of labeled data points used to train the local model. This information is needed for the FedAVG algorithm.

5. Experimental evaluation

5.1. Datasets

We evaluate the effectiveness of SS-FedCLAR considering two public HAR datasets: WISDM [47] and MobiAct [48]. WISDM includes activity data from 36 subjects. Sensor data was collected from the smartphone's accelerometer only. The device was located in the front pants leg pocket. The data collection was supervised by the WISDM team members to ensure the annotations' quality. The activities included in this dataset are the following: *walking*, *jogging*, *sitting*, *standing,going upstairs*, and *going downstairs*. The data distribution of these activities is illustrated in Table 1. The MobiAct dataset includes activity data from 60 different subjects. Sensor data was collected from the accelerometer, gyroscope, and magnetometer of a smartphone positioned in a trousers' pocket freely chosen by the subject in any random orientation. The 73% of the participants were male, while the 27% were female. The subjects' age spanned between 20 and 47 (average: 26), the height ranged from 160cm to 189cm (average: 175), and the weight varied from 50kg to 120kg (average: 76). In our experiments, we considered the following physical activities [2]: *standing*, *walking*, *jogging*, *jumping*, and *sitting.* The data distribution of these activities is illustrated in Table 2.

Those datasets were selected since they involve a relatively large number of subjects with respect to other sensor-based HAR datasets. Even though a real deployment would involve a significantly larger number of participants, this aspect is crucial to evaluate our FL-based approach, considering that data (and participant) augmentation techniques may not lead to realistic results. Moreover, the subjects that participated in data collection in these datasets exhibit both data and labels distribution skew, which is necessary to evaluate the clustering capabilities of SS-FedCLAR.

5.2. Experimental setup

As activity model, in our experiments we used a simple feed-forward deep neural network composed of three fully connected layers having respectively 32, 16, and 16 neurons, and a softmax layer for classification. The inputs of the network are hand-crafted feature vectors extracted in real-time from the stream of sensor data. We consider features that proved to be

[2] In our study, we omitted some of the activities reported in Table 2 because of the small percentage of records available. Indeed, as we will explain later, our evaluation methodology requires partitioning the data of each user. Activities with a small number of samples would be insufficiently represented in each partition and hence they are not suitable for our evaluation.We believe that this problem is only related to this specific dataset and that, in realistic settings, even short activities would be represented by a sufficient number of samples.

Activity	percentage of records
Walking	38.6%
Jogging	31.2%
Sitting	5.5%
Standing	4.4%
Upstairs	11.2%
Downstairs	9.1%
TOTAL	1,098,207 records

Table 1: WISDM: distribution of the considered activities

Activity	percentage of records
Standing	32.6%
Walking	37.8%
Jogging	7.8%
Jumping	7.4%
Sitting	5.5%
Upstairs	2.5%
Downstairs	2.3%
Car step in	1,7%
step out	1,9%
TOTAL	10,788,386 records

Table 2: MobiAct: distribution of the considered activities

effective for HAR in the literature [2]. We used Adam as optimizer. Even though existing FL approaches proposed more sophisticated deep learning classifiers (even to collaboratively learn feature representation), a simpler model with hand-crafted features allowed us to focus only on the specific semi-supervised clustering problem. Moreover, we believe that an advantage of our simple model is a reduced computational effort, which is more suitable for mobile devices.

Some of the hyper-parameters were selected considering the results of our previous work [13]: $l = 1$, $p = 2$, $r = 5$, and 10 local training epochs with a batch size of 30 samples. The remaining hyper-parameters were selected using a grid search, with the objective of optimizing the overall F1 score. For instance, considering the clustering threshold ct, we chose $ct = 0.0035$ for the WISDM dataset, and $ct = 0.0030$ for the MobiAct dataset. The impact of the clustering threshold on the recognition rate and quality of clusters is reported in Section 5.4.3.

5.3. Evaluation methodology

Since we consider a semi-supervised approach, we adopt an evaluation methodology that shows the evolution of the recognition rate and the number of active learning queries. In particular, we adapted the evaluation methodology originally proposed in [14] to include Federated Clustering.

First, we split the dataset into two partitions called Pt and Tr. The partition Pt (i.e., pre-training data) includes data from users that are only used to initialize the global model. The partition Tr (i.e., training data) includes data of the users who actually participate in the FL process. In our experiments, we randomly partition the users in 15% whose data will populate Pt, and 75% whose data will populate Tr. Moreover, we partition the data for each user in Tr

into *sh* shards having the same size. Intuitively, a shard represents the time-period that separates two model updates. Unfortunately, the considered datasets have a limited amount of data and not temporally distributed in intuitive shards (e.g., day). Moreover, intuitive shards will more likely have similar data distributions among the different activities. Hence, we randomly assign to each shard of a user u a fraction $\frac{1}{sh}$ of the available u's data samples, making sure of reflecting in each shard the distribution of the data samples. This approach allows us to mimic a realistic scenario where users perform several types of activities in each shard.

In the following, we describe our evaluation strategy in detail. The global model (pre-trained with Pt) is first distributed to the clients of the users in Tr, which will use it as the first version of the *local model*. Then, the process is composed of *sh* iterations, one for each shard. During a shard, each device exploits the current *local model* to classify the continuous stream of sensor data. The classification output is used to evaluate the recognition rate in terms of F1 score. In parallel, we also apply our active learning strategy keeping track of the number of triggered questions. At the end of the shard, our Federated Learning process starts and the local models are updated. Since for each user the data distribution in its shards was similar, the hierarchical clustering algorithm is performed only at the first shard. In Section 5.4.4 we show the impact of clustering at different shards.

In our experiments, we empirically determined $sh = 4$ for the WISDM dataset and $sh = 3$ for the MobiAct Dataset. Higher values of *sh* would lead to an insufficient amount of data samples in each shard, thus negatively impacting the evaluation of SS-FedCLAR.

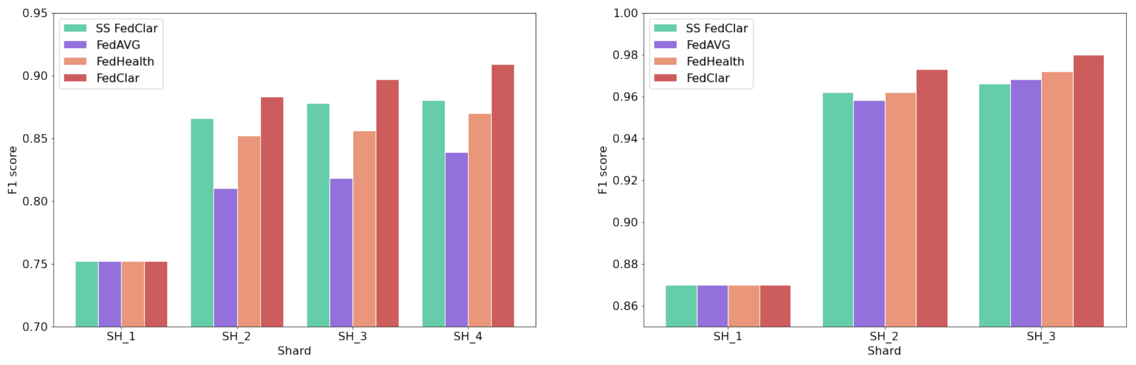
5.4. Results

In the following, we report the results of our evaluation. In particular, we compared SS-FedCLAR considering four baselines:

- **FedAvg** [6]. A classic fully supervised FL approach not considering the non-IID problem.
- **FedHealth** [11]. A fully-supervised FL-based approach for HAR that tackles the non-IID problem using transfer learning.
- **FedCLAR** [13]. A fully-supervised Federated Clustering method for HAR.
- **FedAR** [14]. A semi-supervised FL approach for HAR that combines semi-supervised learning and transfer learning to deal with both the data scarcity issue and the non-IID problem.

5.4.1. *Overall recognition rate*

Figure 4 shows, on both datasets, the recognition rate of SS-FedCLAR at each shard. This figure also compares SS-FedCLAR with the fully-supervised baselines (assuming that clients have complete availability of labeled data at each shard). Even though SS-FedCLAR uses a limited amount of labeled data, with respect to FedCLAR it is only 1.5% behind on the MobiAct dataset and 2.6% on WISDM. Considering the remaining baselines, SS-FedCLAR outperforms them on the WISDM dataset, while it reaches similar results on the MobiAct dataset.



(a) WISDM (b) MobiAct

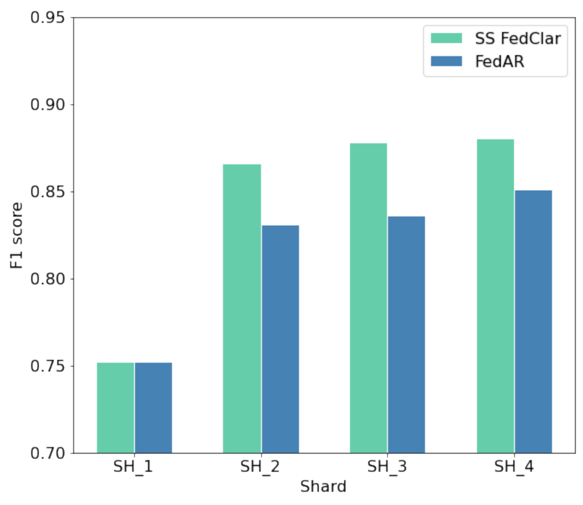
Figure 4: SS-FedCLAR vs. fully supervised baselines shard by shard (F1 score)

These results show that our method is capable of reaching results that are close to the state-of-the-art approaches without assuming labeled data availability.

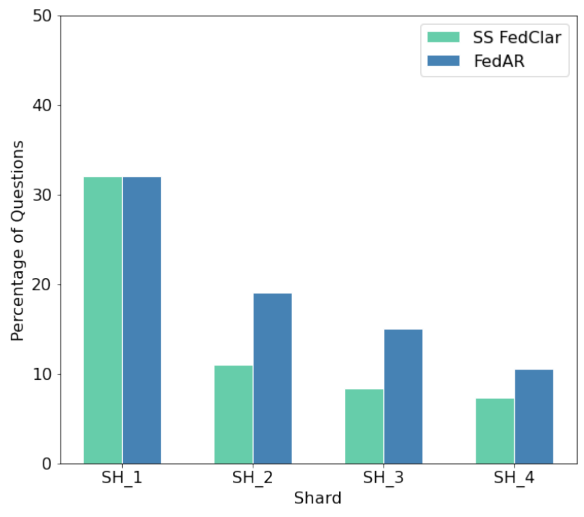
Figure 5a and Figure 6 compares SS-FedCLAR with FedAR, another FL-based semi-supervised HAR approach based on semi-supervised and transfer learning. The results indicate that SS-FedCLAR reaches higher recognition rates on each shard with respect to FedAR and, at the same time, it triggers a significantly lower number of active learning questions. This is due to the fact that our Federated Clustering algorithm better mitigates the non-IID problem, reducing uncertainty during classification.

5.4.2. *Cluster-based results*

Figure 7 and Figure 8 show the results of SS-FedCLAR at the cluster level. For each cluster generated by SS-FedCLAR, we compare the F1 score and the percentage of active learning questions of SS-FedCLAR with the ones of FedAR. Note that these results are just a detailed version of the ones proposed in Figure 5 and Figure 6. Indeed, FedAR is actually evaluated considering all the users, while we show the F1 score considering the subsets of the users based on the output of SS-FedCLAR. Our results show that the federated clustering method has a positive impact on each cluster, especially considering the WISDM dataset.

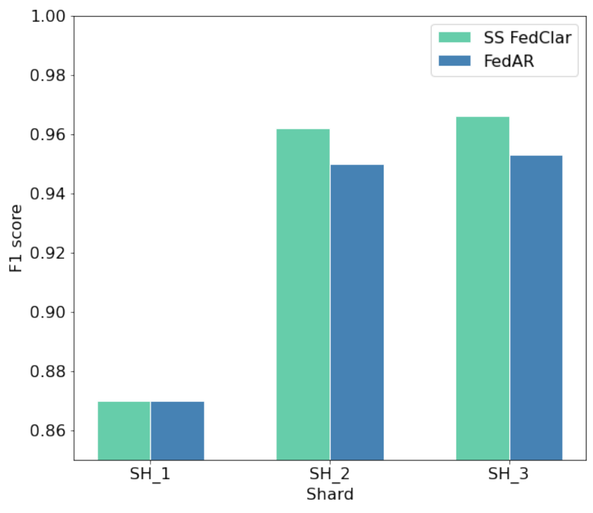


(a) Average F1 score

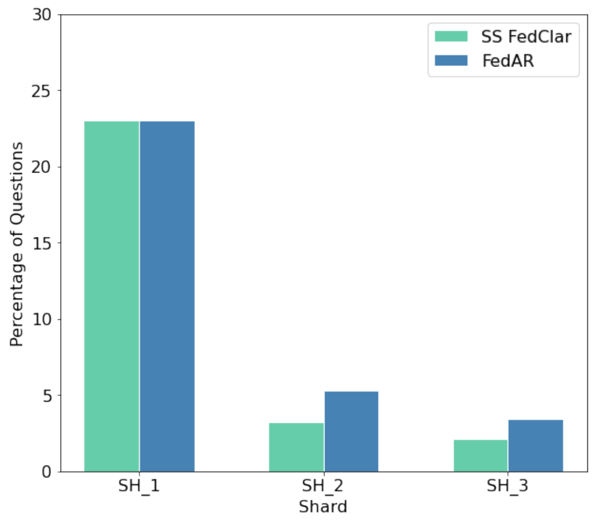


(b) Average percentage of questions

Figure 5: WISDM: Comparison shard by shard of SS-FedCLAR with FedAR in terms of the F1 score and the percentage of triggered questions



(a) Average F1 score



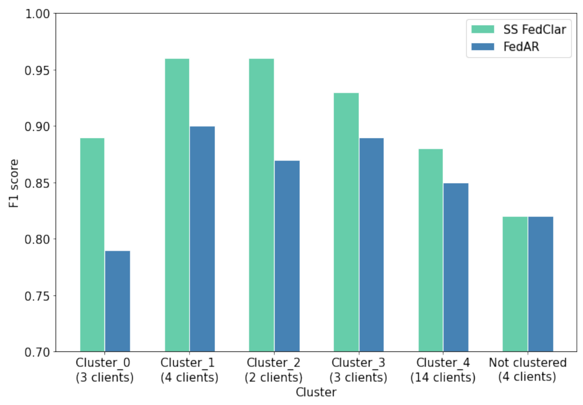
(b) Average percentage of questions

Figure 6: Mobiact: Comparison shard by shard of SS-FedCLAR with FedAR in terms of the F1 score and the percentage of triggered questions

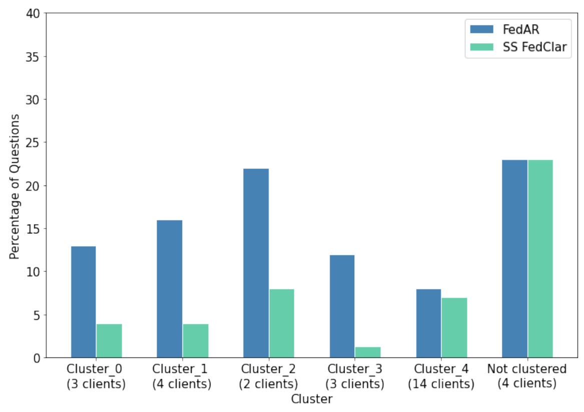
We also observed that only a small percentage of clients was not clustered. Since those clients use a general FL global model, the recognition rate and the number of active learning questions of SS-FedCLAR are similar to the ones of FedAR.

5.4.3. Impact of the clustering threshold

In the following, we show the impact of the clustering threshold *ct* on the recognition rate. Table 3 and Table 4 show that the choice of *ct* has a significant impact on the recognition

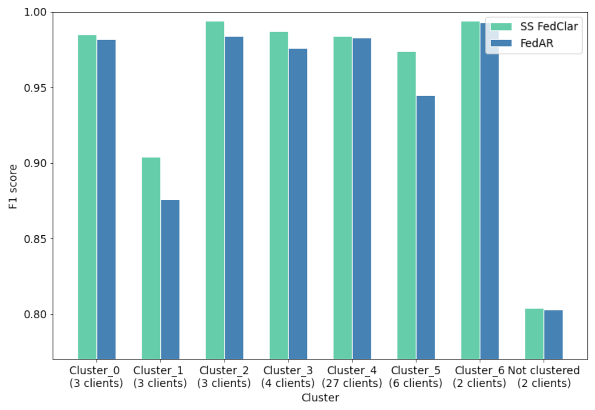


(a) Average F1 score

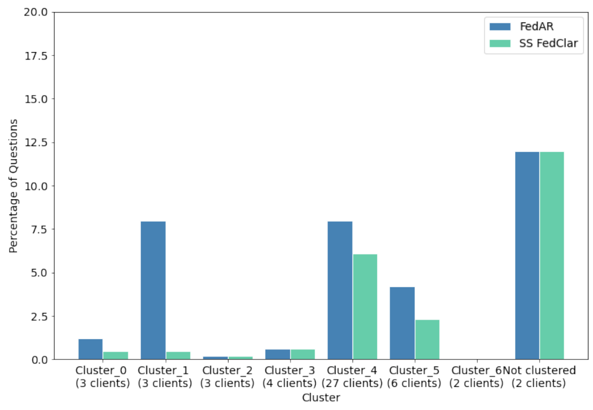


(b) Average percentage of questions

Figure 7: WISDM: Comparison of SS-FedCLAR with FedAR cluster by cluster in terms of F1 score and percentage of triggered questions



(a) Average F1 score



(b) Average percentage of questions

Figure 8: Mobiact: Comparison of SS-FedCLAR with FedAR cluster by cluster in terms of F1 score and percentage of triggered questions

rate, the number of clusters and the percentage of not-clustered clients. When *ct* is too low, SS-FedCLAR generates small clusters and a high rate of *not-clustered* clients, thus negatively impacting the recognition rate.

By closely inspecting the results on the MobiAct dataset in Table 4, we noticed that *ct* values higher than 0.003 do not change the percentage of *not-clustered clients*. This is likely due to the fact that, in this dataset, the users corresponding to not-clustered clients perform activities in a very different way with respect to all the other users.

5.4.4. The impact of clustering at different shards

As we previously mentioned, due to the nature of the considered datasets, in our experiments we performed clustering only during the model update at the first shard. Figure 9 and Figure 10

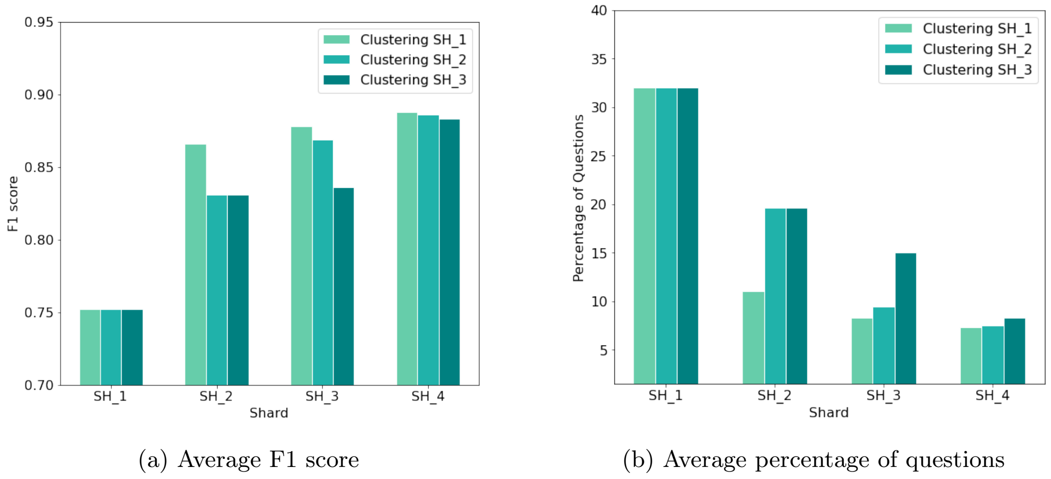
ct	F1	# clusters	clients not clustered
0.0010	0.83	6	56.67%
0.0020	0.85	8	20.00%
0.0030	0.85	6	16.67%
0.0035	0.88	5	13.33%
0.0040	0.86	4	10.00%
0.0045	0.85	4	6.67%
0.0050	0.85	4	6.67%

Table 3: WISDM: Impact of the clustering thresholds

ct	F1	# clusters	clients not clustered
0.0010	0.94	11	23.53%
0.0020	0.95	8	11.76%
0.0030	0.96	7	3.92%
0.0035	0.96	7	3.92%
0.0040	0.96	7	3.92%
0.0045	0.95	6	3.92%
0.0050	0.94	5	3.92%

Table 4: MobiAct: Impact of the clustering thresholds

show the impact of performing clustering at different shards on both datasets. We observed that the advantage of performing clustering at the first shard is that it immediately improves the recognition rate in the following shards and, at the same time, it quickly reduces the number of active learning questions. Hence, these results indicate that even if the clusters are created at the very first shard, they are reliable. This is likely due to the fact that data distribution in the different shards does not change significantly.

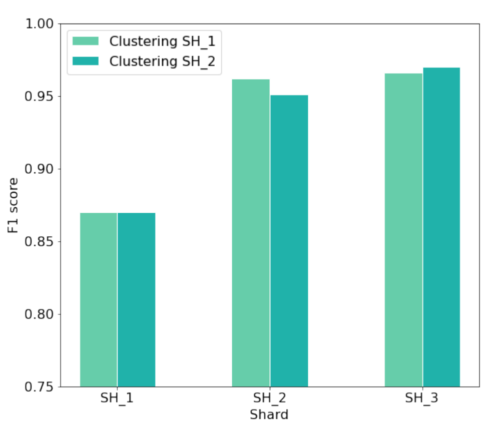


(a) Average F1 score (b) Average percentage of questions

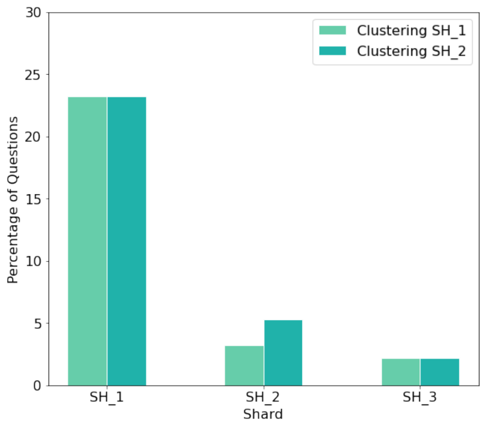
Figure 9: WISDM: The impact of clustering at different shards.

5.4.5. *The impact on non-IID data*

In the following, we show how the non-IID problem is actually mitigated by SS-FedCLAR on the considered datasets. First, we investigate the feature distribution skew. This condition occurs when different users perform the same activity with different patterns. We expect that users grouped in the same cluster perform activities in a similar way, while users in different



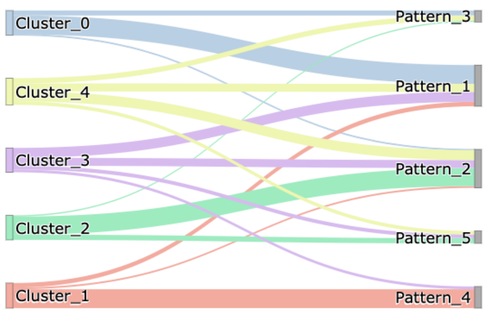
(a) Average F1 score



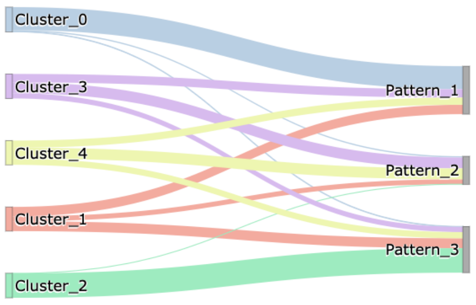
(b) Average percentage of questions

Figure 10: Mobiact: The impact of clustering at different shards.

clusters execute activities in different ways. In order to evaluate if the clusters generated by SS-FedCLAR have this property, from the raw sensor data of all users in each dataset we extract, for each activity, a set of patterns. Each pattern characterises a way of performing that activity [3]. Then, we correlate the patterns with the clusters of users generated by SS-FedCLAR. For the sake of brevity, we report a couple of examples related to the WISDM dataset in Figure 11.



(a) Going Downstairs



(b) Jogging

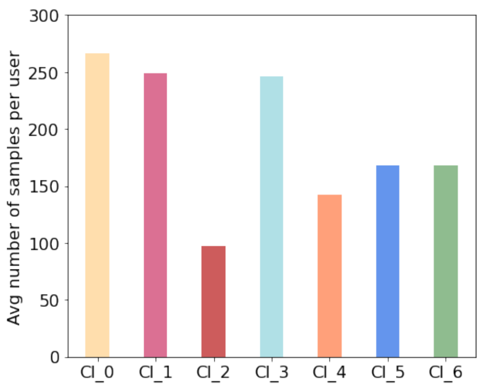
Figure 11: WISDM: examples of feature distribution skew. The plot shows the correlation between clusters generated by SS-FedCLAR and activity patterns.

From this analysis, it emerges that many clusters generated by SS-FedCLAR in WISDM exhibit a peculiar correlation with activity patterns. Hence, the non-IID problem is reduced with a positive impact on the recognition rate. For instance, considering the activities in Figure 11, the improvement in overall F1-score of SS-FedCLAR with respect to FedAR is +12% for *going downstairs* (from 0.64 to 0.76), while +4% for *jogging* (from 0.90 to 0.94). We observed an

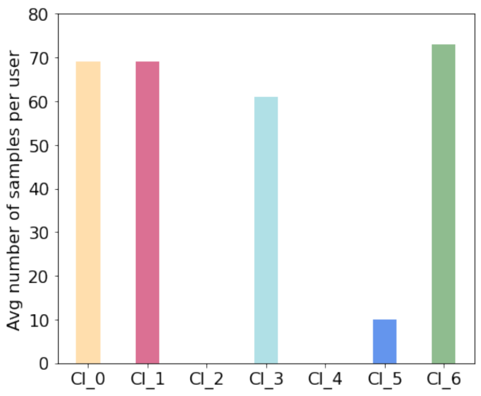
[3]We normalize raw sensor data, we apply PCA for dimensionality reduction and we apply the K-Means algorithm. In order to find the optimal number of clusters for each activity, we maximize the Silhouette score.

improvement in the F1 score for each activity in WISDM whenever there is a clear correlation between clusters and patterns.

We also noticed that the feature distribution skew does not clearly emerge in MobiAct, since most of the users in this dataset tend to perform activities with similar patterns. This is consistent with the results presented above: SS-FedCLAR has in general a minor improvement on this dataset with respect to WISDM. The improvement of SS-FedCLAR on MobiAct is still appreciable since, differently from WISDM, this dataset suffers from a label distribution skew. Hence, SS-FedCLAR is still able to improve the recognition rate by grouping users that have similar label distributions. Figure 12 shows this property for a couple of activities. Considering the examples in this figure, the improvement in F1-score of SS-FedCLAR with respect to FedAR is +5% (from 0.92 to 0.97) for *walking*, while +7% for *sitting* (from 0.86 to 0.93). We observed an improvement in the F1 score for each activity in MobiAct whenever there is a clear correlation between clusters and skewed label distributions.



(a) Walking



(b) Sitting

Figure 12: MobiAct: examples of labels distribution skew. The plot shows the average number of activity samples for each user in the clusters generated by SS-FedCLAR.

5.4.6. Impact of alternative clustering methods

SS-FedCLAR relies on hierarchical clustering due to its nice properties: a) it does not require to know the number of clusters in advance, and b) it generates a dendogram that keeps track of each clustering step, that can be used to easily update the clusters when required.

In the following, we evaluate how different clustering approaches would affect the recognition rate and clusters quality. For the sake of this work, we did not consider clustering approaches requiring a pre-determined number of clusters (e.g., K-Means). We considered DBSCAN [49], that is one of the most widely adopted clustering methods not requiring to know the number of clusters in advance.

Dataset	DBSCAN	Hierarchical
WISDM	0.87	0.88
MobiAct	0.96	0.96

Table 5: F1 score

Dataset	DBSCAN	Hierarchical
WISDM	11.3%	7.3%
MobiAct	3.6%	2.1%

Table 6: Percentage of active learning questions

Table 7: Hierarchical Clustering vs. DBSCAN

We determined the hyper-parameters of DBSCAN as follows. The minimum number of points required to form a cluster was set to 1, in order to consider singleton clusters like in hierarchical clustering. In order to automatically estimate ϵ (i.e., the density threshold) we adopted a state-of-the-art approach [50]. Hence, we determined $\epsilon = 0.0055$ for WISDM and 0.0025 for MobiAct. As distance measure, we adopted the cosine similarity for a fair comparison with our proposed approach.

Table 7 compares the F1 score and the percentage of active learning questions obtained at the last shard by DBSCAN and Hierarchical Clustering. Considering both datasets, we observed that hierarchical clustering reaches similar results with respect to DBSCAN in terms of F1 score. However, we observed a significant reduction in terms of active learning questions with Hierarchical Clustering.

The higher number of active learning queries required by DBSCAN is probably due to the fact that this method generates one large cluster including most of the users, and a few small clusters. Considering the WISDM dataset, DBSCAN generates a cluster that groups 18 users, 2 small clusters incorporating 3 users each, while 6 have not been clustered (only 4 users were not clustered with Hierarchical Clustering). A similar behaviour also emerged for Mobiact, where a big cluster includes 48 users, two other clusters contain 3 and 2 users respectively, while 7 users have not been clustered (only 2 users were not clustered with Hierarchical Clustering). Hence, we believe that Hierarchical Clustering leads to better clusters that require a reduced number of active learning queries to reach high recognition rates. Indeed, a low number of active learning questions indicate that the specialized cluster models are a good fit for the corresponding users. Another disadvantage of DBSCAN is that, differently from Hierarchical Clustering, it would require to perform clustering from scratch when updates are required.

6. Discussion

6.1. Personal data protection

Among the limitations of the current version of SS-FedCLAR, there is the potential leak of private information to an honest-but-curious service provider running the server infrastructure. It is well known that, despite only model parameters are shared with the server, some personally identifiable data could be still inferred from them. In order to mitigate this issue, FL approaches usually rely on Secure MultiParty Computation (SMC) to aggregate the local weights in a privacy-preserving fashion [51]. SMC makes it possible to hide from the service provider the mapping between each local model and the corresponding subject. Even when this type of protection is applied, FL models are exposed to several types of attacks that extract private information from the global model parameters [52]. Examples of such attacks are the reconstruction attack [53], the membership inference attack [54], and the property inference attack [55].

Among the three categories of attacks mentioned above, property inference is particularly problematic for the HAR domain and in particular for SS-FedCLAR. First, since clusters are derived server-side, the SMC-based technique cannot be applied, and the service provider could observe the relationships between local models, clusters and subjects. Moreover, a cluster may actually group users with similar sensitive conditions. For instance, suppose that SS-FedCLAR derives a cluster composed only of subjects with Parkinson disease and a similar gait impairment. An honest-but-curious service provider may use external data of a subject with the same health condition to understand in which cluster is included. The cluster indirectly identifies all the associated subjects revealing that they suffer from the same health condition. There are several possible ways to extend SS-FedCLAR in order to address this and other privacy attacks as it would be required for real-world deployment. In the following, we mention some of them but a complete investigation is out of the scope of this contribution.

A promising solution, that has been recently proposed in the literature, is a federated learning architecture based on Trusted Execution Environment (TEE) [56]. In this scenario, the server-side algorithms of SS-FedCLAR are executed within a protected environment. Indeed, code and data inside a TEE are confidential. The participating clients would transmit encrypted model weights to the service provider, which are then decrypted inside the TEE to update the global models and compute hierarchical clustering. The global model leaves the TEE encrypted. Another possible approach is to use distance-preserving homomorphic encryption to compute clusters and specialized models in a privacy-preserving fashion [57]. The drawback of this approach is that homomorphic encryption may introduce significant computational efforts. Alternatively, it would be possible to use Local Differential Privacy in order to introduce noise during the training of the local models, based on privacy preferences [58]. The major challenge, in this case, is finding an acceptable trade-off between privacy protection and recognition accuracy.

6.2. Evaluation on a large scale

In this work, we took advantage of the public datasets suited for this task and with the highest number of subjects. However, real-world FL-based HAR scenarios may involve thousands or even millions of users. Hence, while SS-FedCLAR exhibits promising results, they need to be confirmed in more realistic experiments on a larger scale. A significant limitation of SS-FedCLAR in large-scale scenarios is that, at each communication round, every participating client is involved in the global model update. However, for the sake of scalability, FL methods randomly sample a limited number of clients at each communication round [6]. Hence, we will investigate a scalable solution to distribute the clustering process in multiple communication rounds. Another significant problem related to deploying SS-FedCLAR on a large scale is the correct choice of the hyper-parameters. Indeed, the hyper-parameters that proved to be effective in our experiments may not reflect the ones that are effective on a large scale. Hence, we will study the challenging problem of choosing the correct hyper-parameters in large-scale scenarios, where only a limited amount of labeled data is actually available.

6.3. Acceptability of active learning based strategy for HAR

In this work, we assumed that users are willing to provide active learning feedback even if the number of queries will quickly decrease when using the system. This assumption is shared with other HAR papers based on active learning [59]. However, it may not be realistic in the HAR domain, since users' availability is highly influenced by the context in which the query is received. For instance, a user may not be willing to provide feedback while participating in a social event. Postponing queries is critical since it becomes challenging to locate in time and remember the activity that was performed. Indeed, each active learning query is associated with a single feature vector processed by the classifier at a specific time instant. In future work, we will perform a user-based study to investigate the contexts in which humans would more likely answer active learning queries also considering different types of interfaces. This study would be targeted to design a context-aware active learning module that issues active learning queries to mobile devices only when there is a high chance that users will provide a feedback.

7. Conclusion and future work

In this paper, we presented SS-FedCLAR, a novel methodology that combines federated clustering and semi-supervised learning for HAR. Our results show that SS-FedCLAR mitigates the non-IID problem and the data scarcity issue at the same time. Indeed, SS-FedCLAR reaches recognition rates that are close to fully-supervised methods and it outperforms state-of-the-art semi-supervised FL-based HAR approaches.

Besides the limitations described in Section 6, a significant problem of SS-FedCLAR is that it still requires some labeled data to initialize the global model. We will investigate self-supervised strategies to derive reliable activity patterns from the stream of unlabeled sensor data. Active learning may still be necessary to associate those patterns with activity labels.

In this work, we evaluated SS-FedCLAR using datasets that only include a limited number of data samples for each user. In a real-world scenario, users may change their activity patterns and habits in the long-term. Moreover, considering a real-world deployment, the set of clients may significantly vary due to new clients that join the system as well as clients that leave the system. Hence we will investigate clustering update strategies and we will perform evaluation on larger datasets.

Another limitation of our current solution is the need of storing labeled ad unlabeled feature vectors, since it may not be sustainable in the long term on a mobile device. This problem could be solved by imposing a limit on the size of the *Feature Vectors Storage*, periodically deleting old or poorly informative data samples.

Finally, we will also investigate alternative clustering strategies, in order to evaluate if other methods besides hierarchical clustering (e.g., spectral clustering, density-based clustering) can derive better groups of similar users.

References

[1] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, Z. Yu, Sensor-based activity recognition, IEEE Trans. Syst. Man Cybern. C:App. Rev. 42 (6) (2012) 790–808.

[2] J. Wang, Y. Chen, S. Hao, X. Peng, L. Hu, Deep learning for sensor-based activity recognition: A survey, Pattern Recogn. Lett. 119 (2019) 3–11.

[3] K. Chen, D. Zhang, L. Yao, B. Guo, Z. Yu, Y. Liu, Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities, ACM Comput. Surv. 54 (4) (2021) 1–40.

[4] C. Bettini, D. Riboni, Privacy protection in pervasive systems: State of the art and technical challenges, Pervasive Mob. Comput. 17 (2015) 159–174.

[5] S. Ek, F. Portet, P. Lalanda, G. Vega, A federated learning aggregation algorithm for pervasive computing: Evaluation and comparison, in: 19th IEEE International Conference on Pervasive Computing and Communications PerCom 2021, 2021.

[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, PMLR, 2017, pp. 1273–1282.

[7] S. Ek, F. Portet, P. Lalanda, G. Vega, Evaluation of federated learning aggregation algorithms: application to human activity recognition, in: Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers, 2020, pp. 638–643.

[8] K. Sozinov, V. Vlassov, S. Girdzijauskas, 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, IEEE, 2018, pp. 1103–1111.

[9] Z. Xiao, X. Xu, H. Xing, F. Song, X. Wang, B. Zhao, A federated learning system with enhanced feature extraction for human activity recognition, KNOWL-BASED SYST 229 (2021) 107338.

[10] C. Zhang, X. Ren, T. Zhu, F. Zhou, H. Liu, Q. Lu, H. Ning, Federated markov logic network for indoor activity recognition in internet of things, Knowledge-Based Systems (2022) 109553.

[11] Y. Chen, X. Qin, J. Wang, C. Yu, W. Gao, Fedhealth: A federated transfer learning framework for wearable healthcare, IEEE Intelligent Systems (2020).

[12] G. M. Weiss, J. Lockhart, The impact of personalization on smartphone-based activity recognition, in: Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence, Citeseer, 2012.

[13] R. Presotto, G. Civitarese, C. Bettini, Fedclar: Federated clustering for personalized sensor-based human activity recognition, in: 2022 IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE, 2022, pp. 227–236.

[14] R. Presotto, G. Civitarese, C. Bettini, Semi-supervised and personalized federated activity recognition based on active learning and label propagation, Personal and Ubiquitous Computing (2022) 1–18.

[15] T. Plötz, Y. Guan, Deep learning for human activity recognition in mobile computing, Computer 51 (5) (2018) 50–59.

[16] U. Bakar, H. Ghayvat, S. Hasanm, S. C. Mukhopadhyay, Activity and anomaly detection in smart home: A survey, Next Generation Sensors and Systems (2016) 191–220.

[17] J. Wang, V. W. Zheng, Y. Chen, M. Huang, Deep transfer learning for cross-domain activity recognition, in: proceedings of the 3rd International Conference on Crowd Science and Engineering, 2018, pp. 1–8.

[18] M. Z. Zadeh, A. Ramesh Babu, A. Jaiswal, M. Kyrarini, F. Makedon, Self-supervised human activity recognition by augmenting generative adversarial networks, in: The 14th PErvasive Technologies Related to Assistive Environments Conference, 2021, pp. 171–176.

[19] E. Kim, S. Helal, D. Cook, Human activity recognition and pattern discovery, IEEE pervasive computing 9 (1) (2009) 48–53.

[20] D. Cook, K. D. Feuz, N. C. Krishnan, Transfer learning for activity recognition: A survey, Knowl. Inf. Sys. 36 (3) (2013) 537–556.

[21] J. Ye, S. Dobson, F. Zambonelli, Lifelong learning in sensor-based human activity recognition, IEEE Pervasive Computing 18 (3) (2019) 49–58.

[22] D. J. Cook, K. D. Feuz, N. C. Krishnan, Transfer learning for activity recognition: A survey, Knowl. Inf. Sys. 36 (3) (2013) 537–556.

[23] Y. Kwon, K. Kang, C. Bae, Unsupervised learning for human activity recognition using smartphone sensors, Expert Systems with Applications 41 (14) (2014) 6067–6074.

[24] Y. Jain, C. I. Tang, C. Min, F. Kawsar, A. Mathur, Collossl: Collaborative self-supervised learning for human activity recognition, Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 6 (1) (2022) 1–28.

[25] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, Smote: synthetic minority over-sampling technique, Journal of artificial intelligence research 16 (2002) 321–357.

[26] K. M. Rashid, J. Louis, Times-series data augmentation and deep learning for construction equipment activity recognition, Advanced Engineering Informatics 42 (2019) 100944.

[27] J. Wang, Y. Chen, Y. Gu, Y. Xiao, H. Pan, Sensorygans: An effective generative adversarial framework for sensor-based human activity recognition, in: 2018 International Joint Conference on Neural Networks (IJCNN), IEEE, 2018, pp. 1–8.

[28] M. H. Chan, M. H. M. Noor, A unified generative model using generative adversarial network for activity recognition, Journal of Ambient Intelligence and Humanized Computing (2020) 1–10.

[29] A. R. Sanabria, F. Zambonelli, J. Ye, Unsupervised domain adaptation in activity recognition: A gan-based approach, IEEE Access 9 (2021) 19421–19438.

[30] E. Soleimani, E. Nazerfard, Cross-subject transfer learning in human activity recognition systems using generative adversarial networks, Neurocomputing 426 (2021) 26–34.

[31] C. Bettini, G. Civitarese, R. Presotto, Caviar: Context-driven active and incremental activity recognition, Knowledge-Based Systems 196 (2020) 105816.

[32] Z. S. Abdallah, M. M. Gaber, B. Srinivasan, S. Krishnaswamy, Activity recognition with evolving data streams: A review, ACM Computing Surveys (CSUR) 51 (4) (2018) 71.

[33] X. Zhou, W. Liang, J. Ma, Z. Yan, I. Kevin, K. Wang, 2d federated learning for personalized human activity recognition in cyber-physical-social systems, IEEE Transactions on Network Science and Engineering (2022).

[34] Q. Wu, K. He, X. Chen, Personalized federated learning for intelligent iot applications: A cloud-edge based framework, IEEE Computer Graphics and Applications (2020).

[35] Q. Wu, X. Chen, Z. Zhou, J. Zhang, Fedhome: Cloud-edge based personalized federated learning for in-home health monitoring, IEEE Transactions on Mobile Computing (2020).

[36] T. Yu, T. Li, Y. Sun, S. Nanda, V. Smith, V. Sekar, S. Seshan, 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, 2020, pp. 104–115.

[37] X. Ouyang, Z. Xie, J. Zhou, J. Huang, G. Xing, Clusterfl: a similarity-aware federated learning system for human activity recognition, in: Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services, 2021, pp. 54–66.

[38] L. Tu, X. Ouyang, J. Zhou, Y. He, G. Xing, Feddl: Federated learning via dynamic layer sharing for human activity recognition, in: Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems, 2021, pp. 15–28.

[39] Y. Zhao, H. Liu, H. Li, P. Barnaghi, H. Haddadi, Semi-supervised federated learning for activity recognition, arXiv preprint arXiv:2011.00851 (2020).

[40] A. Saeed, F. D. Salim, T. Ozcelebi, J. Lukkien, Federated self-supervised learning of multisensor representations for embedded intelligence, IEEE Internet of Things Journal 8 (2) (2020) 1030–1040.

[41] H. Yu, Z. Chen, X. Zhang, X. Chen, F. Zhuang, H. Xiong, X. Cheng, Fedhar: Semi-supervised online learning for personalized federated human activity recognition, IEEE Transactions on Mobile Computing (2021).

[42] C. Briggs, Z. Fan, P. Andras, Federated learning with hierarchical clustering of local updates to improve training on non-iid data, in: 2020 International Joint Conference on Neural Networks (IJCNN), IEEE, 2020, pp. 1–9.

[43] F. Sattler, K.-R. Müller, W. Samek, Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints, IEEE Trans. Neural Netw. Learn. Syst. (2020).

[44] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, in: Adv. Neural Inf. Process. Syst., 2014, pp. 3320–3328.

[45] I. Žliobaitė, A. Bifet, B. Pfahringer, G. Holmes, Active learning with drifting streaming data, IEEE transactions on neural networks and learning systems 25 (1) (2013) 27–39.

[46] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, B. Schölkopf, Learning with local and global consistency, Advances in neural information processing systems 16 (16) (2004) 321–328.

[47] J. R. Kwapisz, G. M. Weiss, S. A. Moore, Activity recognition using cell phone accelerometers, ACM SigKDD Explorations Newsletter 12 (2) (2011) 74–82.

[48] G. Vavoulas, C. Chatzaki, T. Malliotakis, M. Pediaditis, M. Tsiknakis, The mobiact dataset: Recognition of activities of daily living using smartphones., in: ICT4AgeingWell, 2016, pp. 143–151.

[49] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., A density-based algorithm for discovering clusters in large spatial databases with noise., in: kdd, Vol. 96, 1996, pp. 226–231.

[50] N. Rahmah, I. S. Sitanggang, Determination of optimal epsilon (eps) value on dbscan algorithm to clustering data on peatland hotspots in sumatra, in: IOP conference series: earth and environmental science, Vol. 31, IOP Publishing, 2016, p. 012012.

[51] W. Mou, C. Fu, Y. Lei, C. Hu, A verifiable federated learning scheme based on secure multi-party computation, in: International Conference on Wireless Algorithms, Systems, and Applications, Springer, 2021, pp. 198–209.

[52] D. Zhang, X. Chen, D. Wang, J. Shi, A survey on collaborative deep learning and privacy-preserving, in: 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC), IEEE, 2018, pp. 652–658.

[53] L. Zhu, S. Han, Deep leakage from gradients, in: Federated learning, Springer, 2020, pp. 17–31.

[54] R. Shokri, M. Stronati, C. Song, V. Shmatikov, Membership inference attacks against machine learning models, in: 2017 IEEE Symposium on Security and Privacy (SP), IEEE, 2017, pp. 3–18.

[55] L. Melis, C. Song, E. De Cristofaro, V. Shmatikov, Exploiting unintended feature leakage in collaborative learning, in: 2019 IEEE Symposium on Security and Privacy (SP), IEEE, 2019, pp. 691–706.

[56] A. Huang, Y. Liu, T. Chen, Y. Zhou, Q. Sun, H. Chai, Q. Yang, Starfl: Hybrid federated learning architecture for smart urban computing, ACM Trans. Intell. Syst. Technol. 12 (4) (2021) 1–23.

[57] H. Wang, A. Li, B. Shen, Y. Sun, H. Wang, Federated multi-view spectral clustering, IEEE Access 8 (2020) 202249–202259.

[58] Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, D. Niyato, K.-Y. Lam, Local differential privacy-based federated learning for internet of things, IEEE Internet of Things Journal 8 (11) (2020) 8836–8853.

[59] H. S. Hossain, M. A. A. H. Khan, N. Roy, Active learning enabled activity recognition, Pervasive and Mobile Computing 38 (2017) 312–330.