SelfAct: Personalized Activity Recognition based on Self-Supervised and Active Learning

Luca Arrotta, Gabriele Civitarese, and Claudio Bettini

EveryWare Lab, Dept. of Computer Science, University of Milan, Italy
{firstname.lastname}@unimi.it

Abstract. Supervised Deep Learning (DL) models are currently the leading approach for sensor-based Human Activity Recognition (HAR) on wearable and mobile devices. However, training them requires large amounts of labeled data, whose collection is often time-consuming, expensive, and error-prone. At the same time, due to the intra- and inter-variability of activity execution, activity models should be personalized for each user. In this work, we propose *SelfAct*: a novel framework for HAR that combines self-supervised and active learning to mitigate these problems. *SelfAct* leverages a large pool of unlabeled data collected from many users to pre-train through self-supervision a DL model, with the goal of learning a meaningful and efficient latent representation of sensor data. The resulting pre-trained model can be locally used by new users, which will fine-tune it thanks to a novel unsupervised active learning strategy. Our experiments on two publicly available HAR datasets demonstrate that *SelfAct* achieves results that are close to or even better than those reached by fully supervised approaches with only a few active learning queries.

Keywords: Human Activity Recognition · Self-supervised Learning · Active Learning.

1 Introduction

The recent research on sensor-based Human Activity Recognition (HAR) on mobile/wearable devices is dominated by solutions based on Deep Learning (DL) approaches [36]. Supervised learning is the most commonly proposed approach in this area since is capable of reaching significantly high recognition rates [29].

However, such approaches require a huge amount of labeled data to create models capable of generalizing. Indeed, the high intra- and inter-variability of activity execution is a significant challenge in this domain, and personalized models (i.e., models trained only with data from the target users) are the ones performing the best [37].

However, annotation is costly, time-consuming, intrusive, and thus often prohibitive. For this reason, several research groups are investigating solutions for HAR requiring limited labeled datasets [28].

Among these approaches, self-supervised learning methods are emerging [15, 19, 20]. The goal of self-supervised learning is to leverage large corpora of unlabeled data to learn to extract effective features from sensor data. While self-supervised approaches showed their effectiveness on NLP and computer vision tasks [8], their application to sensor-based HAR in real-world scenarios is still an open research question.

One of the major drawbacks of existing approaches is that they assume that a small labeled dataset for fine-tuning is available. However, those works do not investigate how this dataset can be obtained in practice.

In the literature, active learning has been widely investigated to obtain annotated data samples in HAR data scarcity scenarios [1, 18, 28]. However, those approaches assume that a pre-training labeled dataset is available to initialize the recognition model. Indeed, a query is triggered based on the confidence level of the classifier. Applying the concept of active learning to a self-supervised setting is not trivial, since it requires deciding when to trigger queries by analyzing an unlabeled data stream.

In this work, we propose *SelfAct*: a novel framework bridging the gap between self-supervised learning and active learning. *SelfAct* relies on self-supervised learning to learn, from a large number of users, an effective feature representation of sensor data. Then, each user who wants to use *SelfAct* downloads this pre-trained model on personal mobile/wearable devices. After the first phase where the personal device accumulates unlabeled sensor data, a clustering algorithm on the embeddings generated by the pre-trained model is performed. After the accumulation phase, *SelfAct* analyzes in real-time each data sample in the stream and matches it with the closest cluster. A novel unsupervised active learning strategy, based on cluster density, is adopted to decide whether to trigger a query or not. Labeled data samples obtained through active learning are finally used to fine-tune the pre-trained self-supervised model and personalize it on the specific user.

The contributions of this work are the following:

- We propose *SelfAct*: a framework for personalized sensor-based HAR on mobile/wearable devices based on self-supervised learning.
- We design and integrate into *SelfAct* a novel unsupervised active learning strategy to obtain labeled data samples. These samples are used to fine-tune a self-supervised model.
- Our experiments on two public datasets show that *SelfAct* achieves similar or even better recognition rates than fully supervised approaches with a very limited amount of active learning queries.

2   Related work

Most of the existing works in the literature tackling sensor-based HAR on mobile/wearable devices are based on supervised Deep Learning (DL) models [36]. However, during their learning process, such models require large amounts of

labeled data that are often prohibitive to acquire [11]. Moreover, activity models should be personalized for each user to capture her peculiarities in activity execution [11]. Several research groups are indeed working hard to mitigate the data scarcity problem, by proposing several categories of solutions.

Some research efforts proposed solutions based on automatic data annotation. Some of these works assume the existence of previously annotated datasets to pre-train an activity model, that is used for annotation [16]. Other approaches require auxiliary data sources in the environment (e.g., acoustic data from microphones) to automatically annotate activity data [10]. Finally, other works proposed knowledge-based heuristic approaches to generate weak labels (e.g., combining step count and GPS data) [13]. However, the scalability of such an approach is questionable since the heuristic was evaluated considering data collected by monitoring a single user. Moreover, the proposed method can not be used for data annotation when multiple activities share similar patterns in terms of step count and GPS data (e.g., standing and sitting).

Some research groups focused on solutions based on transfer learning [31, 32]. However, such methods rely on models that are pre-trained with significant amounts of labeled data available in a source domain and then fine-tuned in the target domain with a few annotated samples.

Another category of solutions to mitigate the data scarcity problem is semi-supervised learning [2]. However, these approaches still assume the existence of a small labeled dataset to initialize an activity classifier, that is then incrementally updated with samples of the unlabeled data stream annotated through techniques like self-learning, co-learning, active learning [4, 14, 18], and label propagation [28].

Unsupervised approaches have also been proposed to tackle data scarcity. For instance, some research groups directly relied on clustering techniques to distinguish the different user's activities [3, 5, 21, 35]. However, they did not consider how to match clusters with the actual activity classes the user performs in realistic scenarios.

More recently, self-supervised learning has been studied to efficiently learn (without supervision) meaningful features from sensor data for both mobile [7, 30, 34] and smart home [9] activity recognition. The majority of existing solutions assume the availability of a labeled dataset to fine-tune the self-supervised model and obtain the final activity classifier. A closely related work [22] uses a labeled validation dataset to match activities with clusters identified by k-means on self-supervised embeddings. Different from this work, *SelfAct* does not assume the knowledge of the number of activities. Also, *SelfAct* includes an unsupervised active learning strategy to select the samples representing each cluster to label only a few data samples through active learning. Overall, the major drawback of these works based on self-supervised learning is that they do not clarify how fine-tuning datasets can be obtained in realistic settings.

Finally, the work in [17] considered BERT to extract from smart home sensor data a latent representation of the action units performed by the resident (e.g., a movement in the kitchen) that are then clustered. Hence, frequently occurring

patterns of action units are found through a motif discovery process. The most frequent motifs are finally labeled by the resident thanks to active learning. However, NLP-based techniques like BERT and motif discovery work well for discrete environmental sensors, but they cannot be directly applied to the raw inertial measurements collected by mobile and wearable devices.
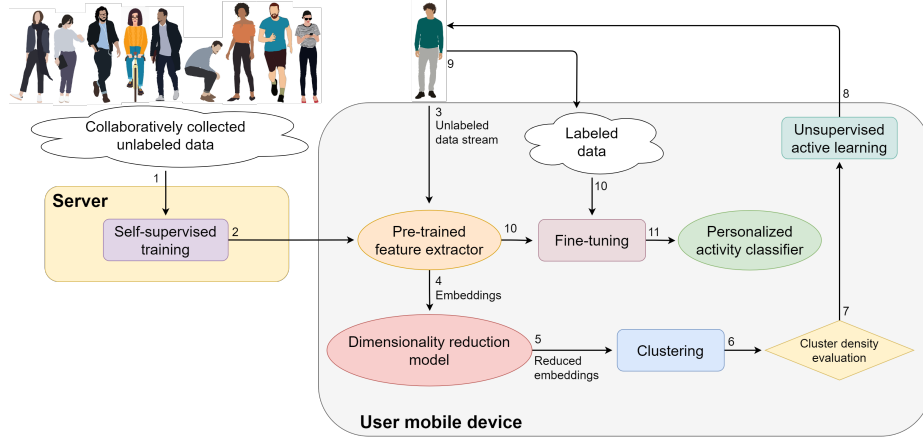
3 Methodology

Fig. 1: The overall architecture of *SelfAct*

In this section, we present *SelfAct*: our framework that generates a personalized activity classifier by combining self-supervised learning and unsupervised active learning.

As depicted in Figure 1, in the first phase (called *self-supervised phase*), *SelfAct* requires huge amounts of unlabeled inertial sensor data automatically and collaboratively collected from the mobile/wearable devices of multiple users (e.g., volunteers). This unlabeled dataset is used to train, in a self-supervised fashion, a deep neural network capable of extracting effective latent representations (i.e., embeddings) from sensor data. This process is described in detail in Section 3.1.

In a second phase (called *accumulation phase*), new users start using *SelfAct* by downloading and deploying the pre-trained self-supervised model on their devices. Each user locally accumulates unlabeled inertial sensor data for a certain period. At the end of the *accumulation phase*, such unlabeled data are transformed into embeddings with the pre-trained model and then used to train a personal dimensionality reduction model. Finally, the reduced embeddings are clustered. Intuitively, each cluster would likely represent a personal activity pattern. Since each activity may be performed with several different patterns (e.g., walking indoors or walking outdoors), we expect that each activity is captured

by different clusters, each one representing a specific execution pattern. More details about this *accumulation phase* are presented in Section 3.2.

After the *accumulation phase*, *SelfAct* starts the *active learning phase* (presented in detail in Section 3.3). This phase analyzes, in real-time, each new sensor data sample in the stream to decide whether to ask the user for feedback about the corresponding label (i.e., the activity that the user is actually performing). Intuitively, *SelfAct* aims at asking for a label only whenever a sample is recognized as representative of a cluster.

Thanks to active learning, *SelfAct* automatically collects a small amount of labeled data specific to the subject and this data is used during the *fine-tuning phase* (presented in Section 3.4) to obtain a personalized activity classifier.

The *SelfAct*'s algorithm running locally on clients' devices is summarized in Algorithm 1. In the following, we will describe each phase of *SelfAct* in detail.

Algorithm 1 Client algorithm of *SelfAct*

Input: the *Feature Extraction Layers* fe of the pre-trained self-supervised model, the accumulation threshold ACC_TH

1: $fth \leftarrow$ randomly initialized layers of the *Fine-Tuning Head*
2: $dr \leftarrow$ NIL
3: $samplesNumber \leftarrow 0$
4: $storage \leftarrow \emptyset$
5: $reducedStorage \leftarrow \emptyset$
6: $labeledSamples \leftarrow \emptyset$
7: $clusters \leftarrow \emptyset$
8: **for** s data sample in the stream **do**
9:     $samplesNumber \leftarrow samplesNumber + 1$
10:     $emb_s \leftarrow$ obtain embeddings using fe
11:     $storage \leftarrow storage \bigcup \{emb_s\}$
12:     **if** $samplesNumber == ACC\_TH$ **then**
13:         $dr \leftarrow$ data reduction model trained using $storage$
14:         $reducedStorage \leftarrow$ embeddings in $storage$ reduced using dr
15:         $clusters \leftarrow$ apply clustering algorithm on $reducedStorage$
16:     **else**
17:         **if** $samplesNumber > ACC\_TH$ **then**
18:             $emb'_s \leftarrow$ dimensionally reduced emb_s using dr model
19:             $matchingClust \leftarrow$ closest cluster to emb'_s in $clusters$
20:             **if** $ActiveLearningNeeded(emb'_s, matchingClust)$ **then**
21:                 $l \leftarrow$ feedback from user on the activity associated with s
22:                 $labeledSamples \leftarrow labeledSamples \bigcup \{(s, l)\}$
23:             **end if**
24:             $matchingClust \leftarrow matchingClust \bigcup \{emb'_s\}$
25:             **if** Fine-tuning conditions are verified **then**
26:                 adjust fth to cover all the classes in $labeledSamples$
27:                 fine-tune $fe + fth$ using $labeledSamples$
28:                 $fe + fth$ can be used to classify unlabeled samples
29:             **end if**
30:         **end if**
31:     **end if**
32: **end for**

3.1 Self-supervised phase

The goal of the *self-supervised phase* is to train, in a self-supervised fashion, a deep learning model capable of transforming raw sensor data samples into

an efficient latent representation (i.e., embeddings). Intuitively, data samples representing the same activity pattern should be close in the latent space, while data samples from different activity patterns should be distant.

In this phase, *SelfAct* relies on unlabeled raw sensor data that can be collaboratively and automatically collected from the mobile/wearable devices of several users. For the sake of this work, we do not focus on how such data are actually collected. For instance, such users can be volunteers involved in a large data collection campaign. Alternatively, unlabeled data could be collected by combining existing publicly available datasets for HAR.

SelfAct pre-processes and segments the available unlabeled data into fixed-length segmentation windows that are used to train the self-supervised model. During the *self-supervised phase*, this model is composed of two parts: i) *Feature Extraction Layers* that learns how to extract embeddings from sensor data, and ii) a *Self-Supervised Head*, necessary to perform the self-supervised training process. *SelfAct* does not impose a specific self-supervised approach. As we will discuss in Section 4, in our experiments we implemented *SelfAct* with a version of the SimCLR [12] approach specifically adapted for the HAR domain [34].

The output of the *self-supervised phase* is a pre-trained deep-learning model only composed of the *Feature Extraction Layers*, that is able to extract embeddings from unlabeled sensor data. This model can be locally deployed on the mobile devices of the new users that start using *SelfAct*.

3.2   Accumulation phase

The *accumulation phase* starts when a new user downloads and locally deploys on her device the model pre-trained in the *self-supervised phase* (i.e., *Feature Extraction Layers*). The goal of the *accumulation phase* is to accumulate from the user sufficient amounts of unlabeled data to reliably discover personal activity patterns through clustering.

As depicted in Figure 2, the user's mobile device starts collecting the stream of unlabeled inertial sensor data that is segmented into fixed-length windows until an arbitrary number of ACC_TH (accumulation threshold) samples is acquired. This parameter should be tuned based on the specific application, and, in real-world scenarios, its value should be high in order to collect unlabeled data over a sufficiently long time period (e.g., a day, or a week).

Once unlabeled data accumulation is completed, each sample is provided to the pre-trained feature extractor to obtain embeddings. Then, to efficiently perform clustering to find activity patterns, we leverage all the generated embeddings to train a personalized dimensionality reduction model. Indeed, recent works demonstrated the impact of dimensionality reduction in improving the performance of clustering techniques [6]. Finally, the dimensionally reduced accumulated embeddings are provided to a clustering algorithm that does not make any prior assumption on the number of clusters (e.g., DBSCAN). The goal is to group in the same cluster those embeddings that have been generated while the user was performing the same activity pattern. We expect our method to
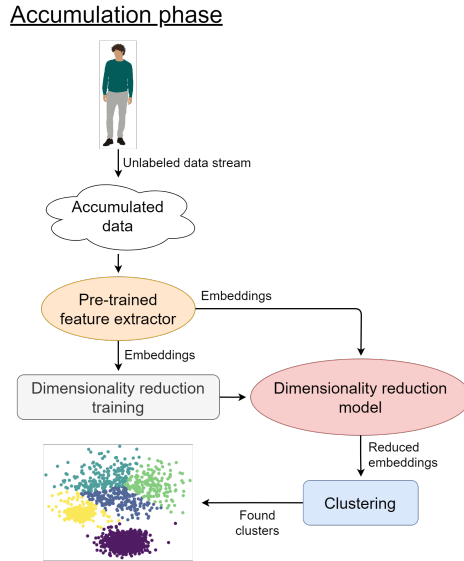
Fig. 2: The *accumulation phase* of *SelfAct*

obtain more than one cluster for each activity, since users may perform the same activity in different ways (e.g., walking indoors and walking outdoors).
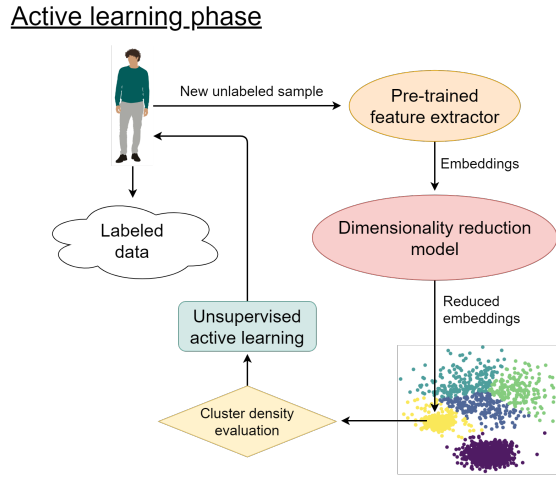
3.3   Active learning phase

The goal of the *active learning phase* is to select, in real-time, data samples that should be labeled by the user. These labeled samples are then used by *SelfAct* to fine-tune a personalized activity classifier.

As shown in Figure 3, in the *active learning phase*, each new unlabeled sample collected by the user's mobile device is provided to the pre-trained model, and the result is dimensionally reduced through the personalized dimensionality reduction model. Each new reduced embedding is then mapped to the closest cluster among the ones derived during the *accumulation phase* (e.g., the cluster with the closest centroid). The active learning strategy of *SelfAct* asks for a label only for those samples that can be considered as highly representative of the closest cluster (i.e., a specific activity pattern) in order to minimize the number of queries that are prompted to the user.

More specifically, an active learning process is started by evaluating whether a new reduced embeddings sample s improves the density of its cluster or not. *SelfAct* assigns to each cluster c a threshold T_c, which reflects the density level of the cluster. This threshold is automatically computed as the average distance between each pair of points within the cluster.

Intuitively, a new sample s is highly-representative of a cluster c if adding s to c increases the density of the cluster. In particular, we evaluate the average

Active learning phase

Fig. 3: The *active learning phase* of *SelfAct*

distance between each pair of points in c after adding s to the cluster. If this average distance is lower than T_c, then s is considered highly representative of c, and an active learning query is requested for the user.

Algorithm 2 shows the pseudo-code of the strategy adopted by *SelfAct* to decide if an active learning query is needed.

Algorithm 2 ActiveLearningNeeded

    Input: a new sample emb'_s, and its closest cluster $matchingClust$
    Output: a boolean value
1: $avg \leftarrow$ average distance between points in $matchingClust$
2: $avg' \leftarrow$ average distance between points in $matchingClust \bigcup \{emb'_s\}$
3: **if** $avg' < avg$ **then**
4:     **return** True
5: **end if**
6: **return** False

Regardless of whether the active learning query is triggered or not, a new sample s is always added to its closest cluster c, and T_c is updated accordingly. The active learning strategy of *SelfAct* assumes that the clusters found at the end of the *accumulation phase* are reliable. For this reason, as we will describe in Section 4, we empirically determined ACC_TH, by balancing the trade-off between recognition rate and a number of triggered questions.

3.4   Fine-tuning phase

Finally, during the *fine-tuning phase*, *SelfAct* relies on the data labeled through active learning at the previous phase to fine-tune the deep-learning model pre-trained during the *self-supervised phase*. In this way, *SelfAct* builds an activity classifier that is personalized for the specific user using the framework. Specifically, *SelfAct* dynamically adds to the *Feature Extraction Layers* of the deep-learning model pre-trained during the *self-supervised phase* a *Fine-Tuning Head* that includes classification layers. Since it is not possible to know in advance how many activities (and which) the user actually performed, the *Fine-Tuning Head* is created dynamically based on the number of activity classes in the labeled data obtained through active learning. During the fine-tuning process, the layers of both the *Feature Extraction Module* and the *Fine-Tuning Head* are unfrozen.

The *fine-tuning phase* should be triggered by considering specific conditions. For instance, it can be a periodic task (e.g., with a time periodicity or each time a certain amount of labels is collected). However, defining when fine-tuning should be started is out of the scope of this paper and it will be investigated in future work. Indeed, periodic fine-tuning would also require strategies to mitigate catastrophic forgetting.

4   Experimental evaluation

This section describes the experiments we carried out to evaluate *SelfAct*. We first introduce the two publicly available datasets we considered for our experiments. Then, we discuss details about our experimental setup: the data pre-processing, the specific models we used to implement *SelfAct*, and the evaluation methodology we adopted. Finally, we present the results of our evaluation.

4.1   Datasets

HHAR The Heterogeneity Human Activity Recognition (HHAR) dataset [33] includes accelerometer and gyroscope data from 9 users wearing a smartphone (in a pouch carried on the waist) and a smartwatch. The dataset was collected considering 6 different activities: *biking*, *sitting*, *standing*, *walking*, *stairs up*, and *stairs down*. HHAR includes heterogeneous sampling rates, based on the specific devices worn by the user. In particular, the dataset includes 2 different smartwatch devices: LG Watch and Samsung Galaxy Gears with maximum sampling rates of $200Hz$ and $100Hz$, respectively. Moreover, HHAR contains 4 different smartphone models: Samsung Galaxy S3 mini (maximum sampling rate of $100Hz$), Samsung Galaxy S3 ($150Hz$), LG Nexus 4 ($200Hz$), and Samsung Galaxy S+ ($50Hz$).

MotionSense MotionSense [24] is a publicly available HAR dataset collected from 24 subjects carrying an iPhone 6s in their trousers' front pocket, and performing 6 different activities: *walking downstairs*, *walking upstairs*, *walking*, *jog-*

ging, *sitting*, and *standing*. The dataset includes both accelerometer and gyroscope data sampled at $50Hz$.

4.2   Experimental setup

In the following, we describe our experimental setup.

Machine learning models As a self-supervised learning model, in our experiments, we used a version of the *SimCLR* technique [12] adapted for HAR in previous works [15,34]. Consistently with the previous work, the *Feature Extraction Layers* consist of three 1D convolutional layers with 32, 64, and 96 filters and 24, 16, and 8 kernel sizes, respectively, followed by a global maximum pooling layer. Each couple of consecutive layers is separated by a dropout layer with a dropout rate of 0.1. The *Self-Supervised Head* consists of three fully connected layers with 256, 128, and 50 neurons, respectively. The self-supervised model is pre-trained for 50 epochs and a batch size of 512 with the SGD optimizer, considering a cosine decay for the learning rate. Moreover, we empirically determined the optimal transformations to apply to the unlabeled data during SimCLR pre-training, finding that, consistently with [34], the application of the rotation transformation only was the best option.

The *Fine-Tuning Head* consists of a fully connected layer containing 1024 neurons and a softmax layer for the final classification. Fine-tuning is performed by running 50 training epochs with a batch size of 1, using an early stopping technique that stopped the training process when validation loss did not improve for 5 consecutive epochs, and adopting the Adam optimizer.

As a dimensionality reduction model, we adopted the Uniform Manifold Approximation and Projection (UMAP) technique [26] since previous work demonstrated how its use is effective in considerably improving the performance of clustering algorithms [6]. In preliminary experiments, we observed that other dimensionality reduction approaches were less effective. For instance, we obtained very negative results with PCA, probably due to the fact that it is a linear dimensionality reduction that is not effective in capturing the non-linear nature of sensor data. On the other hand, UMAP is effective in preserving global data relationships and capturing non-linearity [26].

Moreover, since *SelfAct* considers a real-time scenario for active learning, it is crucial to train a dimensionality reduction model only once at the end of the accumulation phase, and then to map new data samples in the reduced space. These real-time constraints can be addressed by using UMAP, while t-SNE [23], for example, would not be appropriate. During our evaluation, we experimentally determined through a grid search approach the best hyperparameters for UMAP: 15 as the number of neighbors, and an euclidean minimum distance equal to 0.4.

Realistic and real-time constraints mainly guided the choice of the clustering algorithm of *SelfAct*. First of all, it is not realistic to assume knowing the exact number of different activity patterns that the user will perform while using *SelfAct*. Thus, it would not be appropriate to consider clustering algorithms that

require knowing the number of clusters in advance (e.g., K-means). Moreover, in real-time scenarios, we need to rely on an algorithm that (i) clusters the reduced embeddings accumulated during the *accumulation phase*, and then (ii) directly maps new data into the existing clusters during the *active learning phase*. For these reasons, in our experiments, we relied on the HDBSCAN clustering algorithm [25]. In particular, we empirically determined with a grid search process its hyperparameters: a minimum cluster size of 6, and a minimum number of samples of 10.

Pre-processing Since the version of SimCLR we are using only considers data from a single tri-axial sensor as input, we focus on the accelerometer data collected through the users' smartphones. Such data are segmented into fixed-length windows with a 50% of overlap. In particular, following the evaluation protocol proposed in [34], we considered windows of 400 samples for MotionSense. On the other hand, since HHAR includes data collected with different sampling rates, we experimentally determined the size of the windows, obtaining a window size of 800 samples. Due to unsatisfying recognition rates during preliminary experiments, in MotionSense we grouped the labels *walking downstairs* and *walking upstairs* into the label *walking stairs*, while in HHAR we grouped *stairs down* and *stairs up* into *stairs*. Indeed, these activities are particularly challenging to discriminate.

Baseline In our evaluation, we compared *SelfAct* with a *fully supervised* approach: a deep neural network with the same architecture of the network used by *SelfAct* during the *fine-tuning phase*, trained in a supervised way considering full availability of labeled training data.

Evaluation Methodology The *SelfAct* framework is designed for personalizing the HAR model for each user by leveraging a limited amount of labeled data. Hence, we performed our experiments considering the leave-one-subject-out cross-validation technique. At each iteration of the cross-validation process, data from one user is used as the test set (i.e., accumulation, active learning, and fine-tuning phases), while the remaining folds are used as the training set (i,e., self-supervised phase). At the end of the process, we averaged the results obtained at each iteration in terms of weighted F1 score.

Considering the *fully supervised* baseline, at each fold, 10% of the training set was used as the validation set. On the other hand, in order to simulate the *accumulation* and *active learning* phases of *SelfAct*, a certain percentage (i.e. *accumulation threshold*) of the test set was used to simulate the samples that are accumulated by *SelfAct* during the *accumulation phase*. The remaining samples of the test set were used to simulate the data stream processed during the *active learning phase*[1].

---

[1] This setting resembles a scenario in which practitioners must allocate both *accumulation* and the *active learning* phases within a given timeframe (e.g., one month) that is constrained by deployment or project requirements.

During fine-tuning, we used 10% of labeled data as the validation set. Finally, all the samples in the test set that were not labeled (i.e., the ones collected in the *accumulation phase* and the samples not labeled during the *active learning phase*) were used to assess the recognition rate of the fine-tuned model. We also compute the *active learning rate*, i.e., the ratio between the amount of triggered active learning questions and the total number of samples available during the *active learning phase*.

While it was straightforward to apply the leave-one-out cross-validation on HHAR, it was more challenging on MotionSense. Indeed, users in this dataset do not have enough data samples to obtain meaningful results from the *accumulation* and *active learning* phases. Hence, we created pseudo-users by grouping actual MotionSense users performing activities in a similar way (based on available meta-data like age, height, weight, etcetera). This resulted in 5 pseudo-users (with each pseudo-users including data from 4 or 5 MotionSense users), that we used to perform the leave-one-out cross-validation.

4.3   Results

In the following, we describe the results we obtained during our experimental evaluation.

Recognition rate Table 1 compares *SelfAct* and the *fully supervised* baseline in terms of F1 score and the average number of necessary labeled data to deploy such solutions on both datasets.

Table 1: Main results of *SelfAct*. The *accumulation phase* was performed on 75% of the test set. The table reports the average number of active learning queries prompted to the user by *SelfAct* (i.e., number of samples for fine-tuning), and the average number of labeled samples in the training set used to train *fully supervised*

| Dataset	Average number of active learning queries (fine-tuning samples)	*SelfAct* F1	Average supervised training set size	*Fully supervised* F1
HHAR	369	0.9734	25053	0.8792
MotionSense	143	0.9291	5304	0.9483

Such results were obtained by accumulating during the *accumulation phase* a number of samples that corresponds to 75% of the data available in the test set. Considering HHAR, *SelfAct* dramatically outperforms *fully supervised* in terms of F1 score (i.e., $\approx +10\%$). Moreover, this result is obtained considering only 369 labeled samples per user, which is a reduction of approximately 98% compared to the amount of labeled data required by the *fully supervised* baseline (i.e., $\approx 25k$ labeled samples on average at each iteration of the cross-validation).

On the other hand, considering MotionSense, our framework did not improve the recognition rates of *fully supervised*. This happens since the *active learning phase* of *SelfAct* is actually computed on the data of more users (see Section 4.2). Hence, with a few fine-tuning samples labeled with active learning, it is hard to cover all the possible ways in which an activity can be performed by different users. Nonetheless, the F1 score reached by *SelfAct* is only $\approx 2\%$ behind *fully supervised*, but relying only on 143 labeled samples per group of users included in the test set, instead of considering on average more than $5k$ labeled training samples. Hence, we believe that these results still highlight the effectiveness of *SelfAct*, which reaches recognition rates close to supervised approaches with a limited amount of labeled samples.
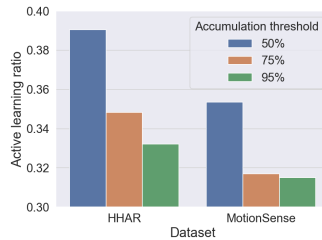
Impact of accumulation threshold Figure 4 depicts how the *accumulation threshold* affects the performance of *SelfAct*.
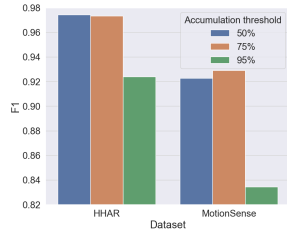
(a) Recognition rates of *SelfAct* by varying the *accumulation threshold*
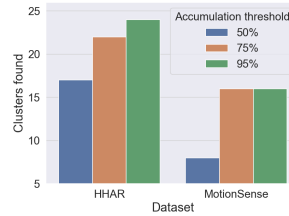
(b) Number of clusters derived by *SelfAct* at the end of the *accumulation phase* by varying the *accumulation threshold*

(c) Active learning rate during the *active learning phase* of *SelfAct* by varying *accumulation threshold*

Fig. 4: The impact of *accumulation threshold* on recognition rate, number of clusters, and active learning rate

By setting the *accumulation threshold* at 50% (i.e., 50% of data used for accumulation and 50% of data in the active learning phase) we get a high recognition rate, at the cost of having the highest active learning rate. Indeed, with this threshold, *SelfAct* finds a lower number of sparse clusters, thus requiring a higher number of samples during the active learning phase to increase their densities. This also implies a higher number of active learning queries. It is important to recall that the number of clusters should not necessarily coincide with the number of activity classes. Indeed, in HAR, it is well-known that the same activity can be performed in very different ways by the subject (e.g., walking can be performed indoors and outdoors). Hence, each activity class can be potentially represented by multiple activity clusters.

On the other hand, when the accumulation threshold is 95% (i.e., 95% of data used for accumulation and 5% of data in the active learning phase) the recognition rate is the lowest. This is due to the fact that this setting does not provide a sufficient amount of data in the active learning phase for properly fine-tuning the model.

Hence, we observed that setting the threshold to 75% (i.e., 75% of data used for accumulation and 25% of data in the active learning phase) led to a high recognition rate (that is similar to the one obtained with the threshold set to 50%) and, at the same time, a significantly reduced active learning rate. Indeed, this setting allows *SelfAct* to identify clusters that better reflect activity patterns and that can be effectively used for the active learning phase.

In general, our results show that increasing the accumulation threshold decreases the active learning rate. This is due to the fact that our unsupervised learning strategy is based on cluster density. Indeed, the higher the amount of accumulated data and the denser the clusters.

5   Conclusion and Future Work

In this paper, we presented *SelfAct*: a framework for personalized sensor-based HAR. Thanks to a novel combination of self-supervised learning and active learning, *SelfAct* requires only a limited amount of labeled data from target users to fine-tune a model pre-trained with unlabeled data only. While the results obtained so far are promising, we plan to investigate several aspects in future work.

First, even though *SelfAct* does not assume knowing in advance the activities performed by the target user, in this work the pre-trained feature extractor and the fine-tuned activity classifier are trained considering the same activity classes. In future work, we want to truly assess the capability of *SelfAct* for activity discovery and domain adaptation. A challenge will be to distinguish outliers from new activities.

A drawback of *SelfAct* is that it relies on a fixed threshold to decide when switching from *accumulation phase* to *active learning phase*. In future work, we will investigate automatic methods based on the clusters' density (e.g., computing the DBCV score [27] of the clusters).

Another limitation of *SelfAct* is that, while it requires limited labeled data samples to obtain high recognition rates, the active learning query rate is relatively high. This aspect has a negative impact on usability. We will investigate more sophisticated active learning strategies aiming at further minimizing the number of required samples. For instance, an interesting strategy is to establish a budget of queries for each cluster. Nonetheless, these results are also probably due to the fact that the considered datasets are small, and we could not evaluate *SelfAct* with large amounts of data for the accumulation and active learning phases. We expect that, in real-world scenarios where it is possible to collect a large amount of unlabeled data, the active learning rate would be significantly lower.

SelfAct also assumes that it is possible to store and process locally unlabeled and labeled datasets. However, in real-world scenarios, unlabeled data may be collected for a significantly long time. For instance, the accumulation phase may last several days. We will investigate solutions based on trusted edge devices with storage and computational capabilities. Users' devices should be only in charge of running the fine-tuned classifier.

Acknowledgment

References

1. Abdallah, Z.S., Gaber, M.M., Srinivasan, B., Krishnaswamy, S.: Streamar: incremental and active learning with evolving sensory data for activity recognition. In: 2012 IEEE 24th International Conference on Tools with Artificial Intelligence, vol. 1, pp. 1163–1170. IEEE (2012)
2. Abdallah, Z.S., Gaber, M.M., Srinivasan, B., Krishnaswamy, S.: Activity recognition with evolving data streams: A review. ACM Computing Surveys (CSUR) **51**(4), 1–36 (2018)
3. Abedin, A., Motlagh, F., Shi, Q., Rezatofighi, H., Ranasinghe, D.: Towards deep clustering of human activities from wearables. In: Proceedings of the 2020 ACM International Symposium on Wearable Computers, pp. 1–6 (2020)
4. Adaimi, R., Thomaz, E.: Leveraging active learning and conditional mutual information to minimize data annotation in human activity recognition. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies **3**(3), 1–23 (2019)
5. Ahmed, A., Haresamudram, H., Ploetz, T.: Clustering of human activities from wearables by adopting nearest neighbors. In: Proceedings of the 2022 ACM International Symposium on Wearable Computers, pp. 1–5 (2022)

6. Allaoui, M., Kherfi, M.L., Cheriet, A.: Considerably improving clustering algorithms using umap dimensionality reduction technique: a comparative study. In: Image and Signal Processing: 9th International Conference, ICISP 2020, Marrakesh, Morocco, June 4–6, 2020, Proceedings 9, pp. 317–325. Springer (2020)
7. Almaslukh, B., AlMuhtadi, J., Artoli, A.: An effective deep autoencoder approach for online smartphone-based human activity recognition. Int. J. Comput. Sci. Netw. Secur 17(4), 160–165 (2017)
8. Baevski, A., Hsu, W.N., Xu, Q., Babu, A., Gu, J., Auli, M.: Data2vec: A general framework for self-supervised learning in speech, vision and language. In: International Conference on Machine Learning, pp. 1298–1312. PMLR (2022)
9. Bouchabou, D., Nguyen, S.M., Lohr, C., LeDuc, B., Kanellos, I.: Using language model to bootstrap human activity recognition ambient sensors based in smart homes. Electronics 10(20), 2498 (2021)
10. Chatterjee, S., Chakma, A., Gangopadhyay, A., Roy, N., Mitra, B., Chakraborty, S.: Laso: Exploiting locomotive and acoustic signatures over the edge to annotate imu data for human activity recognition. In: Proceedings of the 2020 International Conference on Multimodal Interaction, pp. 333–342 (2020)
11. Chen, K., Zhang, D., Yao, L., Guo, B., Yu, Z., Liu, Y.: Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities. ACM Computing Surveys (CSUR) 54(4), 1–40 (2021)
12. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning, pp. 1597–1607. PMLR (2020)
13. Cruciani, F., Cleland, I., Nugent, C., McCullagh, P., Synnes, K., Hallberg, J.: Automatic annotation for human activity recognition in free living using a smartphone. Sensors 18(7), 2203 (2018)
14. Cui, Y., Hiremath, S.K., Ploetz, T.: Reinforcement learning based online active learning for human activity recognition. In: Proceedings of the 2022 ACM International Symposium on Wearable Computers, pp. 23–27 (2022)
15. Haresamudram, H., Essa, I., Plötz, T.: Assessing the state of self-supervised human activity recognition using wearables. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 6(3), 1–47 (2022)
16. Hassan, I., Mursalin, A., Salam, R.B., Sakib, N., Haque, H.Z.: Autoact: An auto labeling approach based on activities of daily living in the wild domain. In: 2021 Joint 10th International Conference on Informatics, Electronics & Vision (ICIEV) and 2021 5th International Conference on Imaging, Vision & Pattern Recognition (icIVPR), pp. 1–8. IEEE (2021)
17. Hiremath, S.K., Nishimura, Y., Chernova, S., Plötz, T.: Bootstrapping human activity recognition systems for smart homes from scratch. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 6(3), 1–27 (2022)
18. Hossain, H.S., Roy, N.: Active deep learning for activity recognition with context aware annotator selection. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1862–1870 (2019)
19. Jain, Y., Tang, C.I., Min, C., Kawsar, F., Mathur, A.: Collossl: Collaborative self-supervised learning for human activity recognition. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 6(1), 1–28 (2022)
20. Khaertdinov, B., Ghaleb, E., Asteriadis, S.: Contrastive self-supervised learning for sensor-based human activity recognition. In: 2021 IEEE International Joint Conference on Biometrics (IJCB), pp. 1–8. IEEE (2021)

21. Kwon, Y., Kang, K., Bae, C.: Unsupervised learning for human activity recognition using smartphone sensors. Expert Systems with Applications **41**(14), 6067–6074 (2014)
22. Ma, H., Zhang, Z., Li, W., Lu, S.: Unsupervised human activity representation learning with multi-task deep clustering. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies **5**(1), 1–25 (2021)
23. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)
24. Malekzadeh, M., Clegg, R.G., Cavallaro, A., Haddadi, H.: Protecting sensory data against sensitive inferences. In: Proceedings of the 1st Workshop on Privacy by Design in Distributed Systems, pp. 1–6 (2018)
25. McInnes, L., Healy, J., Astels, S.: hdbscan: Hierarchical density based clustering. J. Open Source Softw. **2**(11), 205 (2017)
26. McInnes, L., Healy, J., Melville, J.: Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426 (2018)
27. Moulavi, D., Jaskowiak, P.A., Campello, R.J., Zimek, A., Sander, J.: Density-based clustering validation. In: Proceedings of the 2014 SIAM international conference on data mining, pp. 839–847. SIAM (2014)
28. Presotto, R., Civitarese, G., Bettini, C.: Semi-supervised and personalized federated activity recognition based on active learning and label propagation. Personal and Ubiquitous Computing **26**(5), 1281–1298 (2022)
29. Ronald, M., Poulose, A., Han, D.S.: isplinception: An inception-resnet deep learning architecture for human activity recognition. IEEE Access **9**, 68985–69001 (2021)
30. Saeed, A., Ozcelebi, T., Lukkien, J.: Multi-task self-supervised learning for human activity detection. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies **3**(2), 1–30 (2019)
31. Sanabria, A.R., Zambonelli, F., Ye, J.: Unsupervised domain adaptation in activity recognition: A gan-based approach. IEEE Access **9**, 19421–19438 (2021)
32. Soleimani, E., Nazerfard, E.: Cross-subject transfer learning in human activity recognition systems using generative adversarial networks. Neurocomputing **426**, 26–34 (2021)
33. Stisen, A., Blunck, H., Bhattacharya, S., Prentow, T.S., Kjærgaard, M.B., Dey, A., Sonne, T., Jensen, M.M.: Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition. In: Proceedings of the 13th ACM conference on embedded networked sensor systems, pp. 127–140 (2015)
34. Tang, C.I., Perez-Pozuelo, I., Spathis, D., Mascolo, C.: Exploring contrastive learning in human activity recognition for healthcare. arXiv preprint arXiv:2011.11542 (2020)
35. Trabelsi, D., Mohammed, S., Chamroukhi, F., Oukhellou, L., Amirat, Y.: An unsupervised approach for automatic activity recognition based on hidden markov model regression. IEEE Transactions on automation science and engineering **10**(3), 829–835 (2013)
36. Wang, J., Chen, Y., Hao, S., Peng, X., Hu, L.: Deep learning for sensor-based activity recognition: A survey. Pattern recognition letters **119**, 3–11 (2019)
37. Weiss, G.M., Lockhart, J.: The impact of personalization on smartphone-based activity recognition. In: Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence (2012)