Future Generation Computer Systems 163 (2025) 107552

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

FeL-MAR: Federated learning based multi resident activity recognition in IoT enabled smart homes

Abisek Dahal [a,*], Soumen Moulik [a], Rohan Mukherjee [b]

[a] Department of Computer Science and Engineering, National Institute of Technology Meghalaya, Shillong, India
[b] Department of Management Information Systems and Analytics, International Management Institute, Kolkata, India

ARTICLE INFO

Keywords:
Activity recognition
Federated learning
Smart homes
Privacy and security

ABSTRACT

This study explores and proposes the use of a Federated Learning (FL) based approach for recognizing multi-resident activities in smart homes utilizing a diverse array of data collected from Internet of Things (IoT) sensors. FL model is pivotal in ensuring the utmost privacy of user data fostering decentralized learning environments and allowing individual residents to retain control over their sensitive information. The main objective of this paper is to accurately recognize and interpret individual activities by allowing them to maintain sovereignty over their confidential information. This will help to provide a services that enrich assisted living experiences within the smart homes. The proposed system is designed to be adaptable learning from the multi-residential behaviors to predict and respond intelligently to the residents needs and preferences promoting a harmonious and sustainable living environment while maintaining privacy, confidentiality and control over the data collected from sensors. The proposed FeL-MAR model demonstrates superior performance in activity recognition within multi-resident smart homes, outperforming other models with its high accuracy and precision while maintaining user privacy. It suggest an effective use of FL and IoT sensors marks a significant advancement in smart home technologies enhancing both efficiency and user experience without compromising data security.

1. Introduction

In the era of Internet of Things (IoT) the deployment of sensor networks within smart homes has garnered significant attention. The advent of Federated Learning (FL) framework has revolutionized the way Machine Learning (ML) models are trained, especially in the context of IoT in smart homes. The integration of FL framework in smart homes utilizing IoT sensors represents a significant stride in the research community. An FL framework allows multiple devices to collaboratively learn a shared ML model while keeping the data localized addressing major privacy and security concerns [1]. This decentralized approach is crucial in smart homes where sensitive data is involved. This ensures privacy as personal data does not leave the residents' premises and is suitable for real-time applications due to reduced latency [2]. Security challenges in FL is paramount. Research focuses on developing robust models within the FL framework to secure the IoT ecosystem against cyberattacks. The adoption of FL in smart homes necessitates proactive computation strategies to optimize resource utilization and service response times [3]. Context-aware policies learned from multiple smart homes via Federated Multi-Task Learning illustrate FL's adaptability to different conditions essential for personalized services are discussed in [4]. ProtoHAR [5] framework utilizes FL to train a global model on a central server while preserving data privacy. It ensures the flow of global activity prototype knowledge among different clients to correct local representations and optimizes user-specific classifiers to enhance their discriminative power for classification. Multimodal FL integrates data from various sources and improves the learning efficiency and accuracy in activity recognition, which is highlighted in [6]. The researchers in [7] present the T-FedHA framework, which combines a distributed trust management model with hierarchical asynchronous federated learning to tackle the challenges of device trustworthiness and heterogeneity in IoT environments. Their method can help in managing device trust. However the extraction of meaningful insights from such extensive datasets while ensuring residents privacy remains a formidable challenge. In this light, The primary objective of the proposed work is to explore the potential of FL framework in Human Activity Recognition (HAR) in an IoT-enabled multi-resident environment, where multiple sensors capture an abundance of data related to residents daily activities and their interactions with the surrounding environment. The researchers exploited HAR using Smartphone sensor such as accelerometers, gyroscope,etc to capture human activities in detailed in [8].

* Corresponding author.
E-mail addresses: abiseknitskm@gmail.com (A. Dahal), mouliksoumen@gmail.com (S. Moulik), r.mukherjee@imi-k.edu.in (R. Mukherjee).

https://doi.org/10.1016/j.future.2024.107552
Received 30 June 2024; Received in revised form 18 September 2024; Accepted 6 October 2024
Available online 11 October 2024

This research leverages a comprehensive dataset that encapsulates the daily routines and activities of multiple residents residing in two distinct households House A and House B. This dataset is rich in sensor data collected at a high temporal resolution offering detailed insights into residents' interactions with various IoT devices and household items. Each sensor reading is meticulously tagged with a specific activity label enabling the development of models capable of recognizing and categorizing a wide range of resident activities. To address the intricate challenges posed by multi-resident activity recognition we have designed and implemented federated learning frameworks for both households. Within each household affording us the opportunity to tailor our recognition capabilities to the unique behavioral patterns of each individual. Subsequently we have conducted federated learning rounds wherein models are trained on local data preserving data privacy on each resident's device. Following this we have aggregated the models knowledge culminating in global models for each resident that exhibit heightened recognition accuracy. Further, we integrate Elliptic Curve Cryptography (ECC) [9] with FL framework to enhance privacy in sensitive activity classification data shared across the network.

The specific contributions of this paper focus on the following areas -

1. *Application of FL in Smart Home Environment*: In FL framework, a model is trained across multiple decentralized edge devices or servers holding local data samples without exchanging them. This is extremely useful in smart homes for classification and recognition tasks. The use of FL ensures a collaborative act between distributive clients, which makes this approach suitable for multi-home multi-resident HAR. The learning mechanism allows the processing and analysis of data directly on the local devices, preserving data privacy. This method not only speeds up the process but also reduces the bandwidth needed for transmitting large data sets to a central server.
2. *Use of a Deep Learning (DL) model in federated framework*: We have applied an Long Short-Term Memory (LSTM) based model for activity recognition in multi-home multi-resident. The model promises improved accuracy in complex environments where activities are interleaved among residents. A comparative analysis with other algorithms is also provided, for both the global server and the local clients, i.e., different houses.
3. *Incorporation of ECC Encryption Technique for Privacy*: FL framework ensures that sensitive data such as daily routines and personal preferences are processed locally thereby maintaining privacy. To further enhance privacy in smart homes the paper also incorporates an ECC encryption technique that provides a secure method of encrypting and decrypting the model information shared during the FL process. This is crucial in smart home applications where privacy is a major concern as it ensures that any data shared between devices and the central server during the learning process is protected against potential cyber threats.

2. Related work

In recent years researchers are paying attention to various aspects of HAR, starting from HAR of single subject to multi-resident HAR, using different classification methods. A category-wise brief outline of the related works is summarized below.

2.1. Sensing technologies in HAR

The challenges of identifying human activities from video sequences or still images are presented in [10]. TransTM a novel Time-streaming Multiscale Transformer method for RFID-based HAR was introduced in [11]. It recognizes both individual activities and human-to-human interactions without the need for preprocessing. The author in [12] explores HAR in smart home scenarios focusing on the challenges associated with understanding the diverse behaviors of residents. These challenges arise due to varying sensor data and recognition methods. Additionally, the study highlights the importance of recognizing daily behaviors for diagnosing chronic diseases and evaluating treatments. The concept of load-aware continuous-time optimization for dynamic resource allocation and real-time adaptability in multi-agent systems is discussed in [13]. Further multi-agent resource allocation using continuous-time optimization and stochastic learning automata, achieving real-time adaptability and dynamic load balancing is mentioned in [14]. The use of sensors in home environments is emphasized as a means to provide objective measurements of health status in contrast to subjective methods such as questionnaires. Smart suspender integrated with strain sensors, diverging from the conventional reliance on smartphone accelerometers proposed in [15]. Their approach captures body movements periodicity by reducing noise and enabling non-localized measurements ultimately enhancing the accuracy and efficiency of HAR. The work in [16] suggests that a combination of accelerometer and gyroscope features is optimal for HAR.

2.2. Traditional machine learning based classification in HAR

The researchers referenced in [17] used Support Vector Machine (SVM) as a classifier utilizing features from the time domain, frequency domain, and a combination of both. In [18] utilized a Random Forest (RF) approach, which integrates the Bootstrap Aggregation Method (Bagging) and random selection of nodes for Decision Tree construction. The final decision output of the RF model is determined by a majority vote among the various decisions rendered by each tree within the forest. Their methodology necessitates a substantial volume of labeled data for effective training and implementation. In [19] the author applied the K-Nearest Neighbors (K-NN) classification method to distinguish among six types of human activities documented in the WISDM dataset. This analysis utilized features from both the time and frequency domains, derived from data collected by three-axial accelerometers in smartphones. The author in [20] proposed an algorithm called Gaussian Multi-resident Tracking which possesses the ability to handle multi-resident sensor data association in an unsupervised manner, eliminating the need for specific information such as floorplans, sensor locations or the number of residents. Their approach enables the accurate recognition of activities in smart homes with multiple residents providing a robust foundation for behavior modeling and tracking. They have separates a data stream into multiple tracks to identify activities associated with different residents. Researcher in [21] proposed a novel hybrid selection and training pipeline methodology combining nearest neighbor and evolutionary computing for instance selection in smartphone sensing-based HAR.

2.3. Deep learning based classification in HAR

A discussion on HAR through Transfer Learning (TL) across ten datasets comparing TL models with general and user-specific models is featured in [29]. These TL models have achieved over 90% F1 score surpassing both base and user-specific models and have significantly reduced training time, even with limited user data. The author in [30] examines the potential of synthetic data in enhancing Machine Learning classifiers specifically in the context of HAR using wearable device sensors. The GAN model proposed in the study emerges as the most promising for enhancing classifier performance while maintaining alignment with the original data distribution and behavioral characteristics. In addition to their use of a limited evaluation metric, we cannot rely on synthetic data despite its promising potential. An efficient CNN-Long Short Term Memory (LSTM) model for raw sensor data in an uncontrolled environment by leveraging the CNN-LSTM layer to extract spatial and temporal features without pre-processing or augmentation [16]. Their model efficiently handles imbalanced classes. The challenge of effectively extracting continuity

Table 1
Summary of datasets and federated approaches considered in related existing works.

Works	Multi-house?	Subject information	Activity information	Data source	Remarks on federated framework
Sozinov et al. [22]	No	No. of subjects: 9	No. of Activities: 6; Basic activities like sitting, standing, walking etc.	Accelerometer data captured from 13 different smartphones and smartwatches. Apart from IMU, no other sensing is considered.	Framework is not originally federated, rather federated clients are created by partitioning the data.
Xiao et al. [23]	No	Four datasets used: WISDM, UCIHAR2012, OPPORTUNITY, and PAMAP2. For WISDM subject information not mentioned. For the remaining three 30, 4, 9 subjects are considered, respectively.	No. of activities for the datasets - WISDM: 6, UCIHAR2012: 6, OPPORTUNITY: 4, PAMAP2: 11	Except OPPORTUNITY, all other datasets only contain IMU data (mostly accelerometer data) captured from Android-based smartphones.	Framework is not originally federated, rather the data is segmented to create local data for local training.
Ouyang et al. [24]	No	Four datasets - UWB, IMU, Depth and HARBox are created, with no. of subjects as 8, 7, 9, and 121 respectively.	No. of activities - IMU: 3, HARBox: 5. UWB is for a simple 2-class classification	HAR related data are captured only through smartphone-based accelerometer	Although the hardware setup resembles a clustered federated framework, the data segmentation is done on the centrally collected data.
Ek etet al. [25]	No	Dataset used: RealWorld; No. of subjects: 15	No. of activities: 8	Accelerometer and gyroscope time-series data obtained through Android smartphones only	Framework is not originally federated. Each subject of the dataset is considered as a client.
Yu et al. [26]	No	Dataset used: RealWorld and HAR-UCI. Details provided in Ek et al. and Xiao et al. respectively.	Details provided in Ek et al. and Xiao et al. respectively.	Details provided in Ek et al. and Xiao et al. respectively.	Framework is not originally federated. Each subject is considered as federated client.
Li et al. [27]	No	Leaving out gesture recognition, 5 datasets are used for HAR, among which IMU, UWB, WISDM, and HARBox are already discussed. The additional MobiAct dataset considers 57 subjects.	No. of activities in MobiAct dataset: 6.	Accelerometer and gyroscope based data, except UWB, which is doing only a simple 2-class classification, i.e. whether human movement occurred or not.	Framework is not originally federated. Data is partitioned randomly, and that is also differently done for different datasets.
Wu et al. [28]	Partially Yes	MobiAct dataset with 57 subjects.	MobiAct dataset with 6 activities.	accelerometer and gyroscope based data	Framework is not originally federated. Centrally collected data is partitioned to simulate multi-house scenario.
Proposed Work	Yes Fully Multi resident	ARAS dataset, two houses two distinct residents in each House, 21 subjects	Daily day to day activities, 27 different activities	Various sensors are placed in different locations at home.	Fully federated framework.

activity features while reducing reliance on labels in HAR within multiview IoT network scenarios is discussed in [31]. They introduced method that combines supervised activity feature extraction with unsupervised encoder–decoder modules. A binarized neural network and BinaryDilatedDenseNet for HAR using sensor data is highlighted in [32]. They tries to addresses the challenges of high latency and memory requirements in traditional server/cloud-based processing by implementing edge computing.

Convolution network MRFC-Net and auto-encoder network MRFC-VAE-Net multiresolution fusion convolution network used DAASG has been developed in [33]. It highlights the efficiency and advanced capabilities of the methods in dealing with unknown classes and improving feature representation. They had developed a dataset called 'Daily-Abnormal Activity of Special Groups' which include the routine surveillance of specific populations including prisoners and elderly individuals. A transformer-based model with a Swin-Transformer backbone that enhances the behavioral feature extraction from multi-scale information was discussed in [34]. A novel hierarchical semi-synchronous communication approach named SHFL to enhance FL incorporating a weighted aggregation method and a CNN-based anomaly detection in milling machine is highlighted in [35].

2.4. Convolution based classification in HAR

The classification method proposed in [11] takes raw RFID RSSI data as input that combines a multiscale convolutional hybrid transformer to capture behavioral features. DCapsNet [36] is a deep neural network architecture that merges convolutional layers with capsule networks for activity and gait recognition using sensor data. It utilizes convolutional layers for extracting temporal features and scalar representations from input data followed by CapsNet to capture equivariance with magnitude and orientation. However DCapsNet exhibits some limitations in gait recognition due to its higher directional orientation. The integration of Convolutional Neural Networks (CNNs) with the Convolution Block Attention Module (CBAM) for visual data and Convolutional Long Short Term Memory (ConvLSTM) for time-sensitive multi-source sensor information is discussed in [37]. This approach demands substantial computational resources and relies on multimodal data which limits its suitability for edge computing or scenarios with single modal data commonly found in real world applications. Author in [38] proposed a Temporal convolutional neural network (TCNN) which can efficiently analyzes and recognizes human activities using short video inputs. Their architecture integrate 3D convolutional and convolutional LSTM layers which leverages spatio-temporal features to achieve superior real-time classification.

2.5. Miscellaneous classification in HAR

The author in [39] discuss about the recognition of multiple resident types which is enhanced by leveraging Markov Logic Networks (MLN) which combine First-order Logic (FoL) and Markov Networks (MN) to capture complex relationships between activities and types of residents. Incorporating models such as gender, age bracket, job and family roles. Their proposed method improves the accuracy of inferring resident types and enables personalized services based on multi-label results. Their study tackles the challenge of distinguishing between the performers of activities in multi-resident scenarios by utilizing high-dimensional features including time sequences, duration's and periods.

Synthesis: Researchers have explored various aspects of machine learning and deep learning based classification for categorizing the

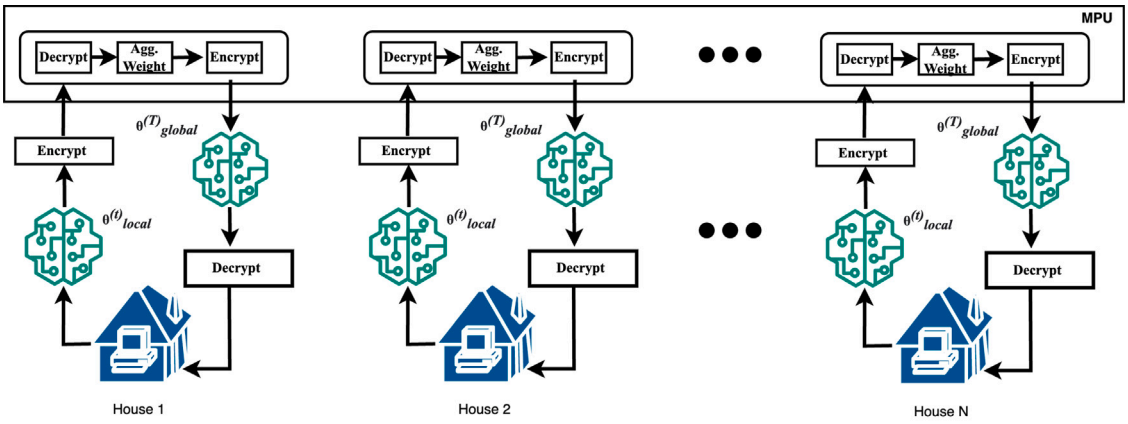


Fig. 1. Proposed system model.

activities performed by individuals or groups. However, these methods are often insufficient for real-world scenarios as they deal with simple activities. Also, the training and evaluation of these models require extensive amount data and computational capacities. Most of the activities in studies are conducted in controlled or lab-based environments. There is still a lack of effective model fusion for sensor data, as many proposed systems function perfectly only under specific conditions. Additionally, there is a scarcity of multi-sensor datasets. Moreover, many models involve collecting and training datasets on a central server. Individual activity data is both crucial and sensitive, and it must be handled with the utmost priority. There is a risk of data leakage from central servers. Ideally data should be managed within an individual's premises and models should be trained locally. This approach will help preserve data privacy, ensuring that data remains within the individual's control and limits which is achieved through federated learning. A summary of datasets and federated learning approaches for human activity recognition considered in the literature survey are mentioned in Table 1. The table highlights the various datasets used for evaluating human activity recognition. Additionally, it provides an overview of different federated learning techniques employed, their specific use case observed in distributed environments.

3. Fel-MAR system model

In this section, a detailed and comprehensive mathematical framework of our FL approach is presented. In an IoT-enabled smart home system consisting of N houses, denoted as $HouseA$, $HouseB$, ..., $HouseN$, sensors are deployed to collect resident activity data. Each house is equipped with a Local Processing Unit (LPU), which processes the collected data through local computations $\theta_i^{(t)}$local and stores the results in its respective local database. Local models are trained at each LPU and then sent to the Main Processing Unit (MPU). The MPU aggregates these local models into a combined model. It compares the aggregated local model $\theta_i^{(t)}$local with the global model $\theta_i^{(t)}$global, and performs operations to generate an updated model $\Delta\theta_i^{(t)} \leftarrow \theta^{(t)}_{\text{local}} - \theta^{(t)}_{\text{global}}$. This updated model is then sent back to each LPU for further processing in subsequent rounds. Privacy within the system is preserved by only sending updates of the local models to the MPU rather than the entire models or the raw data. The system model showing two houses involved in the process is depicted in Fig. 1. For readers' convenience the Fel-MAR system model description is divided into the following subsections - data description, proposed LSTM-based model, communication protocol of proposed model, FL process, privacy-preserving aggregation, enhancing data privacy with ECC in FL.

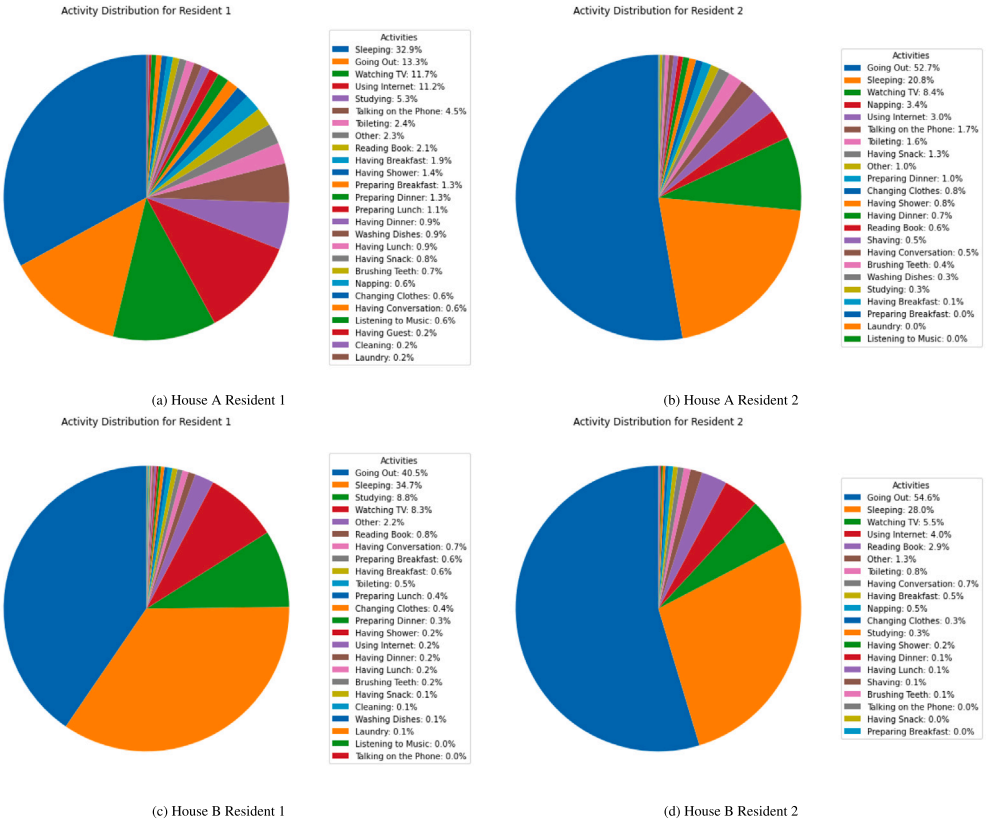
3.1. Data description

We used the publicly available ARAS [40] dataset to conduct this experiment. The datasets comprises 30 days of sensor data from two households 'House A' and 'House B' with each house inhabited by two residents. Data is collected in secondly interval. The data includes 20 columns of various sensor readings like photocells, IR, force, distance, contact, sonar, and temperature sensors, and the last two columns contain activity labels for each resident. The data is collected at a high granularity of one-second intervals, providing detailed information about the residents' interactions with various parts of the house. This allows the model to capture both long-term trends and short-term changes in activity patterns. This data represents a detailed time-series dataset capturing the minute-to-minute activities and environmental conditions within the households. The pie chart depicted in Fig. 2(a), 2(b) and 2(c), 2(d) illustrates the distribution of activities for two different residents in two different houses over the span of a month. The corresponding lengths indicate the total amount of time spent on each activity over a month. The corresponding values are mentioned in percentages representing the proportion of time spent on various activities throughout the months. In 'House A' and 'House B', these sensors are distributed across different rooms, such as kitchens, living rooms, bedrooms, and bathrooms, to capture a wide variety of daily activities (e.g., cooking, cleaning, sleeping, etc.). The dataset contains 27 different activity labels that cover these daily activities. The dataset is temporally dense, with readings captured every second, allowing for fine-grained activity recognition. This level of granularity provides a robust dataset for recognizing short and long-duration activities with high accuracy. The high frequency of data collection allows the model to capture not only the occurrence of events but also the transitions between them, which is essential for distinguishing between different activities in a multi-resident setting. Sensor activation patterns, such as frequent kitchen sensor usage during meal times, were key to recognizing overlapping activities.

Each house indexed by i, is treated as a client and holds a unique dataset D_i, systematically represented as:

$$D_i = \{(x_{i,j}, y^{(1)}_{i,j}, y^{(2)}_{i,j})\}_{j=1}^{N_i} \tag{1}$$

where N_i is the number of samples in house i. Each sample contains a sensor readings vector, $x_{i,j} \in \mathbb{R}^N$, corresponding to the N different sensors placed within the house. The last two elements, $y^{(1)}_{i,j}$ and $y^{(2)}_{i,j}$, are one-hot encoded activity labels for Resident 1 and Resident 2 respectively, representing the activities performed by the residents. The sensor activation count of respective houses are illustrated in Figs. 3 and 4. The insights gained from sensor activation count within a home illuminate patterns in routines zones of activity and resident behaviors at specific times. For instance, when the refrigerator sensor is active between 07:00 and 10:00 h and the CO2 sensor in the kitchen is active



(a) House A Resident 1 (b) House A Resident 2

(c) House B Resident 1 (d) House B Resident 2

Fig. 2. Monthly activity distribution of residents in House A and House B. The top row displays the distribution for residents 1 and 2 in House A, while the bottom row shows the distribution for residents 1 and 2 in House B.

during these times. We can infer that a resident is likely using the fridge or the kitchen. Identifying peak activity periods and analyzing the duration and frequency of sensor activations, we can pinpoint opportunities for optimizing energy and resource use. Furthermore, these insights enable us to detect anomalies that may signal health or safety concerns. This will significantly enhance living conditions by ensuring that automation supports the lifestyles of residents in both an efficient and effective manner.

3.2. Proposed LSTM based model

As depicted in Fig. 1 House A and House B independently train machine learning models on their local data. Post-training each houses transmit their updated model to a central server which then aggregate model weights. This central aggregation results in an enriched model that is then redistributed back to each house for further refinement. The cycle repeats several times, allowing the model to progressively learn from the collective data while maintaining individual data privacy. The process begins with the independent collection of data at two locations House A and House B. Each house then pre processes its data locally with LPU, ensuring the information is formatted correctly and is suitable for applying machine learning. If the data is deemed valid, each house proceeds to train a local model on its dataset. This is a crucial step as it allows for the development of models that are tailored to the specific characteristics of the data at each house. Post-training, the performance of each local model is evaluated. If the models meet predefined performance criteria and their weights are encrypted using the Algorithm 2 (page 10) a method that ensures the secure transmission of these weights to a central server. Upon receiving the encrypted weights the central server decrypts them and performs an aggregation process. This involves averaging the weights from House A and House B to create an updated global model. This global model theoretically benefits from learning diverse patterns from both datasets. The updated aggregated model is then re-encrypted using same with Algorithm 2 (page 10) and sent back to both houses. Upon receipt each house decrypts the model and integrates it into its local setup. This process is iterative meaning it repeats several times. Each iteration aims to enhance the model accuracy and generalizability by learning from the aggregated experiences of both locations. After a predetermined number of iterations or once the model reaches a satisfactory level of performance a final evaluation is conducted. This step assesses the effectiveness and accuracy of the federated model. Post-evaluation, the process concludes, and the final model can be deployed in practical applications, ensuring it is robust and privacy-preserving. Dataset with sensor readings and activity labels, each input vector at time t is represented as $x_t \in \mathbb{R}^{22}$, where 22 is the number of features (20 sensor readings + 2 activity labels) Each input x_t to the LSTM represents the sensor data at time t. The LSTM layers process this data, taking into account not just the current inputs but also what they have learned from previous inputs. It is crucial for understanding activities as the context sequence of sensor readings is important. An LSTM unit within an Recurrent Neural Network (RNN) is adept at processing time-series data due to its ability to retain information over long periods. The LSTM consists of several components each with a specific function. The forget gate decides which information is discarded from the cell state.

$$f_t = \sigma(W_f \cdot [h_{(t-1)}, x_t] + b_f) \tag{2}$$

where σ is the sigmoid function, W_f is the weight matrix for the forget gate, $[h_{(t-1)}, x_t]$ is the concatenation of the previous hidden state and the current input, and b_f is the bias. The input gate and candidate layer determine what new information is added to the cell state.

$$i_t = \sigma(W_i \cdot [h_{(t-1)}, x_t] + b_i) \tag{3}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{(t-1)}, x_t] + b_C) \tag{4}$$

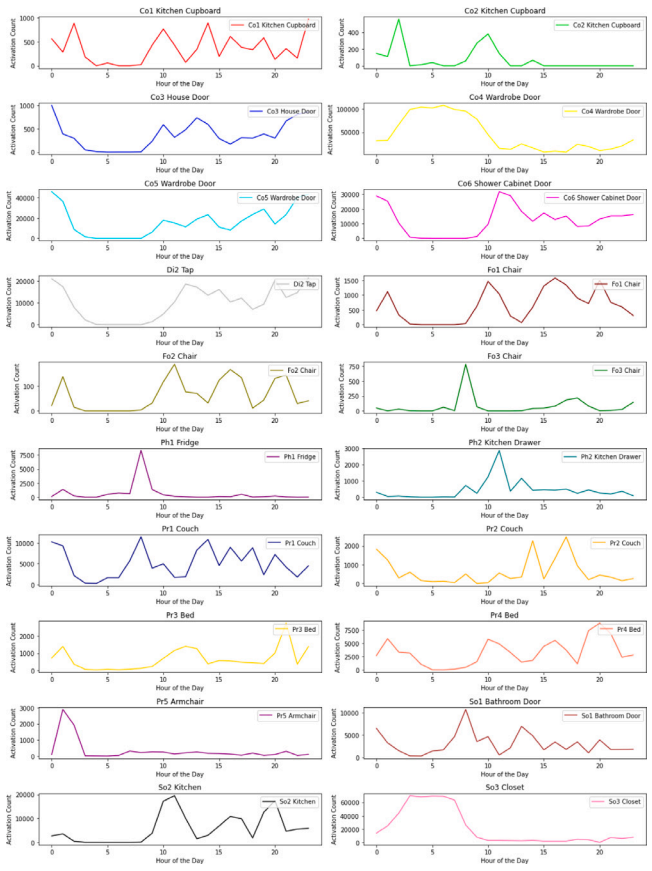


Fig. 3. Sensors activation count of House A.

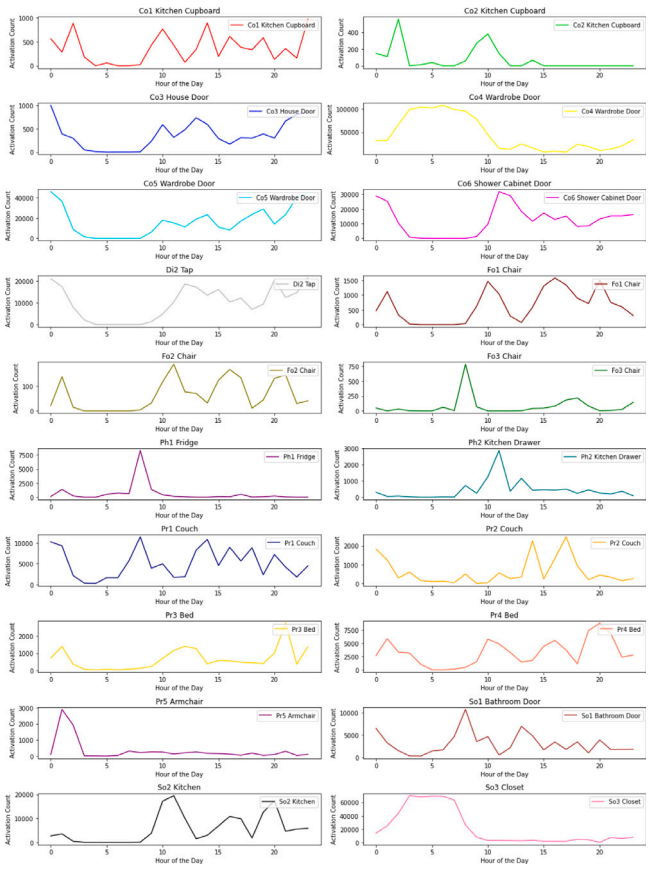


Fig. 4. Sensors activation count of House B.

Here, W_i and W_C are weight matrices, and b_i and b_C are biases. The cell state is updated by combining the old state with new information.

$$C_t = f_t * C_{(t-1)} + i_t * \tilde{C}_t \tag{5}$$

The output of the LSTM layers is then processed by further neural network layers, culminating in the prediction of the resident's activity. This is achieved by mapping the LSTM outputs to one of the 27 activity labels through a softmax layer (see Fig. 5).

The communication function C defines the protocol for bi-directional information flow,including sensor data and model parameters between clients and the server.

$$C : \text{LPU}_i, \text{MPU} \rightarrow \text{Data Transmission} \tag{6}$$

The communication operates in discrete rounds, denoted by t, where $t = 1, 2, \ldots, T$ and each round involves the transmission of model parameters and sensor data. Data transmission includes the amalgamation of model parameters and sensor data. The process flowchart is illustrated in Fig. 6.

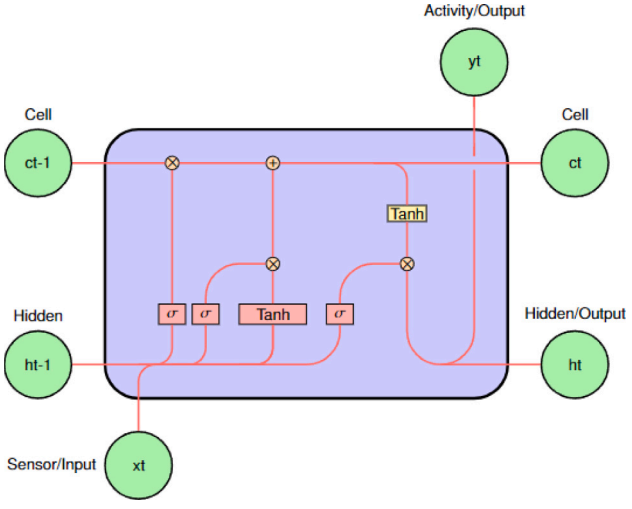


Fig. 5. Architecture of an LSTM unit with sensor input and activity output.

3.3. Communication protocol of proposed model

Our FL framework is designed to minimize the global loss function $L(\theta_{\text{global}})$ by aggregating updates from N distributed house (for example in our case $N = 2$), each holding a local dataset D_i with the aim of training a global model θ_{global}. The main task is to reduce communication costs while ensuring minimal impact on the model's performance. We implement model and update compression techniques to address communication efficiency which are foundational to our protocol. Model compression including quantization, is represented as

$$Q(\theta) = \text{round}\left(\frac{\theta}{q}\right) \times q, \tag{7}$$

where $Q(\theta)$ denotes the quantized model parameters, θ the original parameters and q the quantization step. It will significantly reduces the data size required for transmission. Similarly, update compression through gradient sparsification is applied by transmitting only gradients that exceed a certain threshold τ, hence

$$S(g) = \begin{cases} g & \text{if } |g| \geq \tau \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

where, $S(g)$ is the sparsified gradient after applying sparsification and g is original gradient computed during the training process. τ is threshold

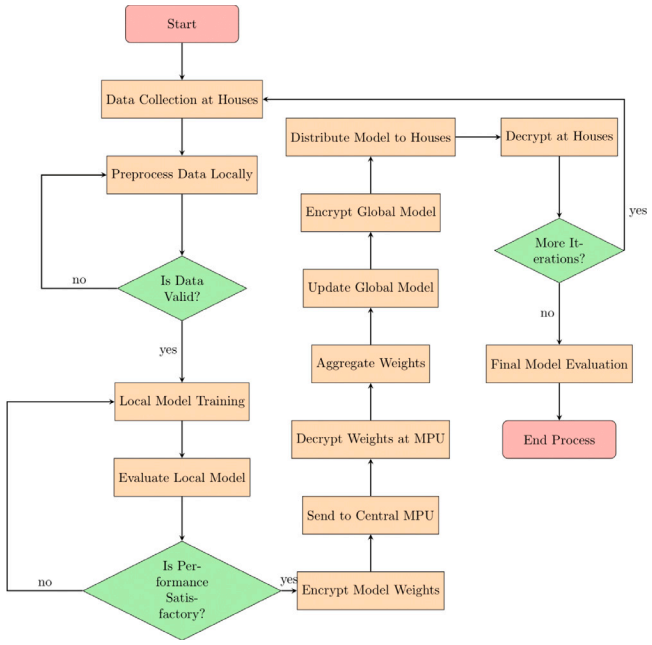


Fig. 6. Process flowchart of FeL-MAR Model.

for sparsification below which gradients are set to zero. Further we used an optimized version of Federated Averaging (FedAvg), where the global model update is determined by

$$(\theta_{\text{global}}^{t+1}) = \sum_{i=1}^{N} \frac{n_i}{n} \theta_{i^t} \tag{9}$$

where $\theta_{\text{global}}^{t+1}$ is global model parameters after aggregation at round $t+1$. N is total number of houses participating in the training process.

n_i is number of data samples held by the ith device, n is total number of data samples across all devices. θ_i^t is the model parameters from the ith device at round t. Leveraging weighted averaging of local models based on their respective data contributions. This method significantly enhances communication efficiency by reducing the frequency and volume of data exchange. We formulate an optimization problem aiming to minimize the global loss $L(\theta_g)$ subject to a communication cost constraint $C(\theta_g^{t+1} - \theta_g^t) \leq \epsilon$, ensuring that the efficiency gains do not detrimentally affect the model's accuracy. Performance degradation is carefully evaluated to ensure that $|L(Q(\theta_g)) - L(\theta_g)| \leq \delta$ remains within acceptable bounds, where δ denotes an acceptable level of performance loss due to compression. Where $C(\cdot)$ is the function measuring the communication cost for transmitting model updates. ϵ is the constraint on the maximum allowed communication cost per round. L$(\cdot)$ is the loss function evaluating the model performance, δ is an acceptable level of performance loss due to compression. We address the trade-off between communication cost and model accuracy by employing 8-bit quantization and gradient sparsification with a threshold of 0.01. It will significantly reduce the size of updates transmitted, thereby lowering communication overhead. It is noted that, quantization can introduce minor precision loss, and sparsification may discard gradients below the threshold, ensuring that accuracy loss remains within 1%–2% maintaining model effectiveness while optimizing communication efficiency.

3.4. FL process

In the proposed FL framework for multi-residential environments, the process begins with the initialization of a global model on the central server. This global model, denoted as θ_{global}, is distributed to all participating clients (e.g., smart homes) at the start of each communication round t.

Each client c_i, after receiving the global model, performs local training on its dataset D_i, which consists of sensor data and corresponding activity labels. The local model is updated based on the client's data, and the difference between the local and global models is computed as $\Delta\theta_i^{(t)}$. This update $\Delta\theta_i^{(t)}$ is transmitted back to the server, ensuring that no raw data is exchanged, thereby preserving privacy.

Upon receiving the updates from all clients, the server aggregates these updates to form a new global model $\theta_{\text{global}}^{(t+1)}$, using a weighted average of the local updates. The updated global model is then sent back to the clients for the next round of training. This iterative process continues for a predefined number of rounds T, until the model converges or no further significant updates are detected.

The final global model is evaluated on test datasets to assess its accuracy in recognizing multi-resident activities while ensuring data privacy throughout the process. The sequence diagram of FL process is depicted in Fig. 7.

Algorithm 1 FL Process for Multi-Residential Environments

Input: Global model $\theta_{\text{global}}^{(0)}$, Local datasets D_i, Number of rounds T.
Output: Updated global model $\theta_{\text{global}}^{(T)}$.
for $t = 1, 2, ..., T$ **do**
  for each client c_i in parallel **do**
    Receive the global model: $\theta_{\text{global}}^{(t)} \leftarrow$ Server
    Local Training: $\theta_{\text{local}}^{(t)} \leftarrow$ Train on D_i using $\theta_{\text{global}}^{(t)}$
    Compute Local Updates: $\Delta\theta_i^{(t)} \leftarrow \theta_{\text{local}}^{(t)} - \theta_{\text{global}}^{(t)}$
    Send Local Updates to Server: Server $\leftarrow \Delta\theta_i^{(t)}$
  end for
  Privacy-Preserving Aggregation:
  Aggregate Local Updates with Noise Addition:
  $\theta_{\text{global}}^{(t+1)} \leftarrow \theta_{\text{global}}^{(t)} + \eta \sum_{i=1}^{C} w_i \Delta\theta_i^{(t)} + \mathcal{N}(0, \sigma^2)$
end for
Evaluation:
Evaluate the final global model $\theta_{\text{global}}^{(T)}$ using test datasets from each house to assess its performance.

3.5. Privacy-preserving aggregation

To protect the sensitive data within the local updates, a privacy-preserving aggregation mechanism is employed. This method ensures that the global model $\theta_{\text{global}}^{(t+1)}$ does not reveal any individual client's contribution. Specifically, differential privacy is applied during the aggregation by adding Gaussian noise $\mathcal{N}(0, \sigma^2)$ to the updates. This noise addition ensures that the probability distribution of the model updates remains consistent, regardless of whether any particular data point is included or excluded from a client's dataset.

$$\theta_{\text{global}}^{(t+1)} = \theta_{\text{global}}^{(t)} + \eta \sum_{i=1}^{C} w_i \Delta\theta_i^{(t)} + \mathcal{N}(0, \sigma^2) \tag{10}$$

The parameter σ controls the trade-off between privacy and accuracy; a higher value of σ increases privacy at the cost of model precision. The privacy guarantee is quantified using the privacy budget ϵ and the privacy loss parameter δ. Lower values of ϵ and δ provide stronger privacy guarantees, as they limit the ability to distinguish between two datasets that differ by a single data point. Through this privacy-preserving aggregation technique, the system ensures that sensitive data from individual residents remains confidential while still enabling the global model to benefit from the collective knowledge of all clients. The principle of indistinguishability inherent in Differential Privacy guarantees that the presence or absence of any single data point does not significantly affect the output maintaining the confidentiality of individual data points.

$$\text{Indistinguishability: } P[f(D) + \text{Noise}] \approx P[f(D') + \text{Noise}] \tag{11}$$

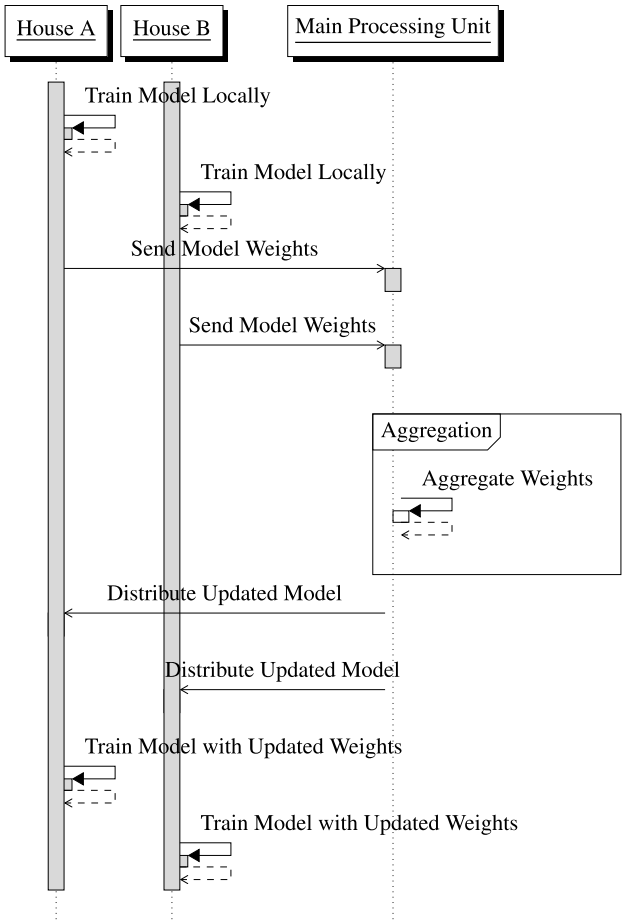


Fig. 7. Sequence diagram of FL process with two houses.

This approach ensures that while the global model benefits from the aggregated knowledge of all local models the information specific to residents remains protected.

3.6. Enhancing data privacy with ECC

In a our FL system comprising of House A, House B, and MPU as mentioned in 1, **Elliptic Curve Cryptography (ECC)** plays a pivotal role in securing the transmission of model weights. The system's security is anchored in the properties of an elliptic curve, typically defined by the equation [9]:

$$y^2 = x^3 + ax + b \mathrm{mod} p \tag{12}$$

where a and b are constants that uniquely define the curve's characteristics. with $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$, $a = -3$, and b as specified by the secp256r1 standard. Each client and the server generate a 256-bit private key, with the public key obtained by scalar multiplication with the curve's base point G. Each entity, including House A, House B, and MPU, initializes the process by generating their respective ECC key pairs. This involves each party selecting a private key, a large random number, and calculating the corresponding public key through scalar multiplication of the elliptic curve's base point **G** with their private key.

As *House A* completes a training iteration, it encodes its model weights as a point M on the elliptic curve. A random integer k is chosen, and the weights are encrypted as:

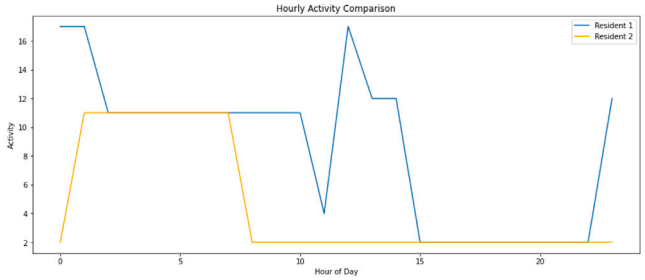
$$C_1 = k \times G \tag{13}$$

$$C_2 = M + k \times Q_{\mathrm{MPU}} \tag{14}$$

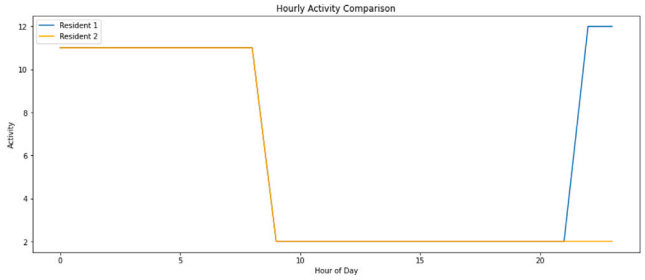
where Q_{MPU} denotes the public key of the *MPU*. The pair (C_1, C_2) is then securely transmitted to the *MPU*. Upon reception, the *MPU* uses its private key to decrypt the weights, aggregates them, and subsequently re-encrypts the updated model using the public keys of {House A, House B, ..., House N}. This process ensures that each client receives the updated global model securely. The suggest ECC approach in our scenarios not only ensures the confidentiality of the model weights during transmission but also maintains the overall integrity and privacy of the FL process. We chose ECC for its efficiency, as it provides strong security with smaller key sizes compared to traditional cryptography methods like RSA, which reduces communication overhead. Moreover, ECC was implemented using python libraries cryptography and ecdsa, ensuring an efficient and secure encryption process. Upon receiving the encrypted weights, the central server decrypts them, performs an aggregation process, and averages the weights from both House A and House B to create an updated global model. This global model benefits from learning diverse patterns from both datasets while maintaining strong privacy guarantees throughout the FL process. The computational overhead of ECC primarily stems from the scalar multiplication operation, which has a time complexity of $O(n^2)$ for a 256-bit key. Despite this, the encryption and decryption times are on the order of milliseconds, introducing minimal delays to the overall system performance. We choose ECC over alternatives like RSA and AES due to its smaller key size, which reduces communication overhead. RSA requires much larger keys for 2048+ bits for equivalent security, resulting in higher transmission costs

4. Result and discussion

We conducted an extensive simulation in the Jupyter environment utilizing 15 GB RAM and an 8 GB DDR4 graphic environment while



(a) Activity Schedule of House A



(b) Activity Schedule of House B

Fig. 8. Activity Schedule of houses with two residents.

Algorithm 2 Secure Model Weight Transmission in FL using ECC

Input: Model weights from House A and House B, ECC base point G

Output: Aggregated and updated model weights
Key Generation:
for each participant (House A, House B, MPU) **do**
  Generate private key d
  Compute public key $Q = d \times G$
end for
Encrypt Model Weights (Local Nodes):
for each house **do**
  Encode model weights as point M on curve
  Choose random integer k
  Compute $C_1 = k \times G$
  Compute $C_2 = M + k \times Q_{\text{MPU}}$
  Transmit (C_1, C_2) to MPU
end for
Decrypt and Aggregate (MPU):
Receive (C_1, C_2) from each house
for each received pair **do**
  Compute $M = C_2 - d_{\text{mpu}} \times C_1$
end for
Aggregate weights to form M_{agg}
Re-Encrypt and Distribute (MPU):
for each house **do**
  Encrypt M_{agg} using house's public key
  Send encrypted M_{agg} back to house
end for
Final Decryption (Local Nodes):
for each house **do**
  Decrypt received M_{agg} using private key
end for

employing Python libraries. Fig. 8 in show the activity of R1 and R2 from house A and House B which show that both the residents from House A 3:00 to 6:00 h performing the similar activity Sleeping that the common pattern and consistent behavior suggests a shared routine during these hours. Also both the resident of house almost share the similar activity throughout the day on an average. The hyperparameters used is mentioned in Table 5.

Table 2
Performance evaluation of ML models running in global server considering non-federated centralized framework.

Model	P	R	A	F1	RMSE	MAE
SVM	89.5	90.1	90.5	89.8	0.31	0.20
LR	90.0	90.5	90.6	90.3	0.31	0.19
MLP	90.2	90.7	90.8	90.5	0.30	0.18
RF	89.8	90.4	90.6	90.1	0.31	0.20
DT	89.6	90.2	90.3	89.9	0.32	0.21
NB	90.1	90.6	90.7	90.4	0.30	0.18
LSTM	*89.5*	*90.5*	*90.8*	*91.2*	*0.30*	*0.18*

We evaluate the effectiveness and performance of our model using a standard evaluation metric which is calculated across all classes. The precision is calculated by dividing the number of true positives (TP) by the sum of true positives (TP) and false positives (FP). Mathematically,

$$P = \frac{TP}{TP + FP} \tag{15}$$

$$R = \frac{TP}{TP + FN} \tag{16}$$

$$F1 = \frac{2 * PR}{P + R} \tag{17}$$

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{18}$$

We have utilized models such as SVM, LR, MLP, RF, DT, NB and LSTM to assess the accuracy with which each model classifies the activity defined in the datasets. The evaluation metrics are presented in Table 1, which combines datasets from both houses without Federated Learning. Table 2 displays the evaluation metrics of a Federated Learning Model using SVM, LR, MLP, RF, DT, NB and Fel-MAR (or the LSTM with federated framework). This is a specifically defined model for federated learning that employs a weighted average function to integrate custom evaluation metrics on the server side for various models within a federated learning framework.

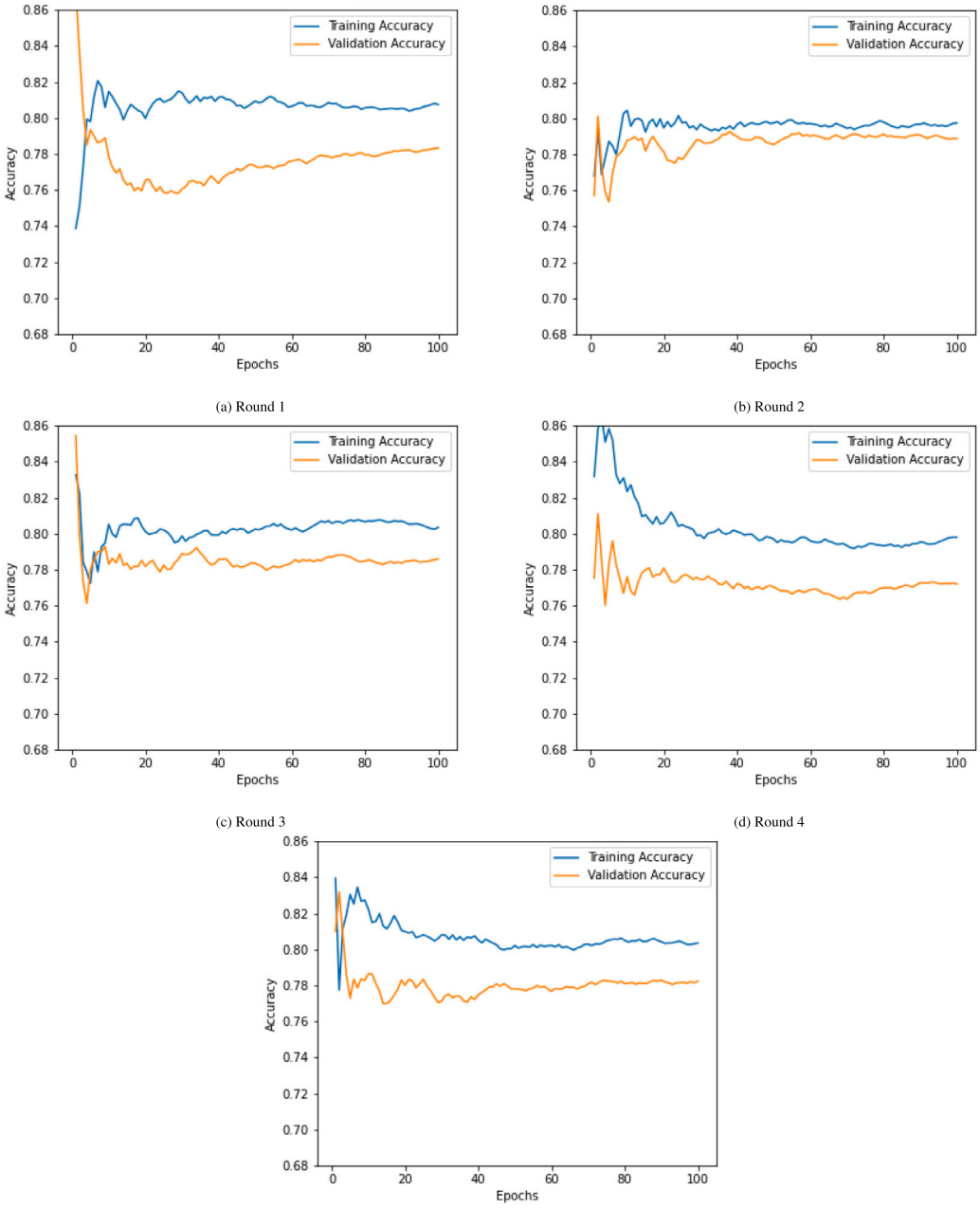


Fig. 9. FeL-MAR model accuracy across five rounds of global server.

Table 3
Performance evaluation of ML models running in global server considering federated framework.

Model	P	R	A	F1	RMSE	MAE
SVM	87.0	88.0	86.7	87.6	0.39	0.22
LR	88.1	88.3	88.2	88.2	0.38	0.21
MLP	88.5	88.9	88.6	88.7	0.37	0.20
RF	87.9	88.2	87.0	88.1	0.38	0.22
DT	87.6	87.8	87.7	87.7	0.39	0.23
NB	88.2	88.4	88.3	88.3	0.38	0.21
FeL-MAR	*89.0*	*89.5*	*89.6*	*89.3*	*0.16*	*0.10*

Starting with the Support Vector Machine (SVM) it demonstrates a solid balance between precision and accuracy both hovering around 89%–91%. However its recall and F1-score, while respectable are slightly lower than its precision and accuracy indicating a marginally less effective performance in identifying all activity. When integrated into a federated learning framework the accuracy of SVM model observed a decline to approximately 86%–87%. This reduction highlights the challenges posed by federated learning environments model generalization across diverse and decentralized datasets which can impact the overall model effectiveness. Logistic Regression (LR) shows a consistent performance across precision, recall and F1-score, all approximately 90%. This indicates a well-balanced classifier that is equally adept at identifying positive instances and avoiding false positives. Its accuracy slightly above 90% although not the highest still reflects a reliable predictive ability. Conversely within the federated learning framework LR model experienced a noticeable drop in performance across all evaluated metrics. Precision, recall, and F1-score were

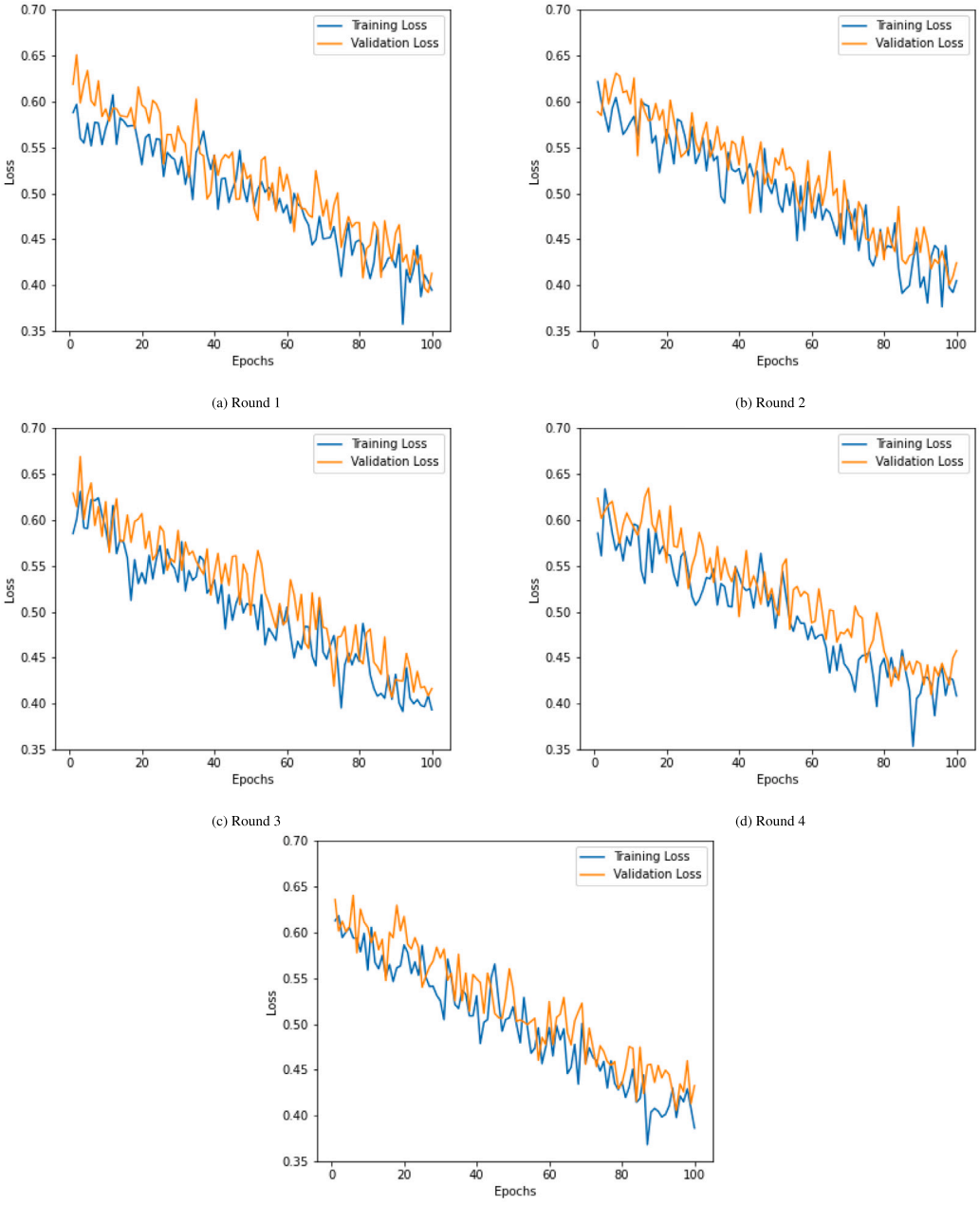


Fig. 10. FeL-MAR model loss across five rounds of global server.

uniformly reduced to approximately 88%. The decline is similar to that observed with the SVM classifier underscores the inherent challenges of federated learning such as data heterogeneity and communication constraints which can adversely affect model performance. Random Forest (RF) renowned for its robustness exhibits a similar pattern to LR with its metrics closely aligned around the 90.6% mark. This uniformity across precision, recall and F1-score underscores RF's capability in handling diverse data features effectively. Within the federated learning framework, however, there is a noticeable decline in RF's performance. The model's accuracy drops to 87%, with precision at 87.9%, recall at 88.2% and F1-score of 88.1%. Despite the decrease, these results still reflect a relatively high level of performance, indicating RF's resilience and adaptability to the federated learning environment.

The Decision Tree (DT) classifier often appreciated for its simplicity shows a slightly lower performance compared to others especially in

Table 4
Performance evaluation of ML Models running in local clients (House A and House B) in federated framework.

Model	House A				House B			
	A	P	R	F1	A	P	R	F1
SVM	86.8	88.2	86.5	87.1	86.8	88.2	86.5	87.1
LR	86.1	86.9	87.6	88.2	86.1	86.9	87.6	88.2
MLP	85.1	86.9	87.6	88.2	85.1	86.9	87.6	88.2
RF	86.6	86.9	88.8	87.7	86.6	86.9	88.8	87.7
DT	88.6	86.6	85.6	87.9	88.6	86.6	85.6	87.9
NB	86.6	88.3	88.5	87.3	86.6	88.3	88.5	87.3
Fel-MAR	*90.3*	*88.3*	*87.9*	90.1	*89*	*88.3*	*89.1*	*88.8*

Table 5
Hyperparameters for LSTM Model and Federated Learning Process.

Hyperparameter	Value	Justification
LSTM Model Hyperparameters		
Optimizer	Adam	Efficient and adaptive for time-series dataset
Learning Rate	0.0005	Provides smoother convergence and minimizes oscillations
Loss Function (both residents)	Categorical Crossentropy	Standard used by many researchers for multi-class classification tasks
Metrics	Accuracy	For evaluating classification tasks
LSTM Units	100	Captures temporal dependencies without overfitting
Dropout Rate	0.3	Helps in reducing overfitting and balancing regularization
Dense Output Units (per resident)	Unique activity classes	Matches 27 activity classes for correct classification
Batch Size	128	Frequent updates for better convergence in limited resources
Number of Epochs	50	Allows the model to capture data patterns without overfitting
Validation Split	20%	Standard split to ensure proper training and validation
Federated Learning Hyperparameters		
Number of Local Epochs	10	More local training improves local models before global updates
Federated Averaging Rounds	40	Reduces communication overhead, maintains model accuracy
Client Participation Rate	100% (both houses)	Full client participation ensures faster convergence
Optimizer (local)	Adam	Efficiently handles sparse gradients in distributed learning
Learning Rate (local)	0.0005	Matches the learning rate for consistency across clients
Batch Size (local)	64	Frequent updates improve local training effectiveness
Communication Frequency	Every 10 local epochs	Reduces communication costs while maintaining global updates
Aggregation Method	Federated Averaging (FedAvg)	Weighted by local data size for proportional updates

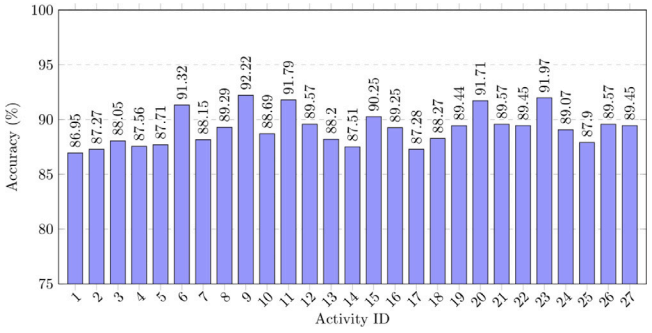


Fig. 11. Activity-wise recognition accuracy for Resident 1 in House A.

terms of precision and recall. These metrics along with its F1-score suggest a modest capability in distinguishing between classes, which could be due to the model's tendency to overfit or its simpler decision boundaries. Naive Bayes (NB) known for its efficiency in large datasets, demonstrates a good balance with precision and recall both exceeding 90%, aligning closely with its F1-score. Conversely, within the federated learning framework NB performance slightly decreases with the model achieving an accuracy of 88.3%, precision at 88.2%, recall at 88.4%, and F1-score at 88.3%. Despite the decline, NB still demonstrates a good level of performance, showcasing its effectiveness even in decentralized and distributed learning settings. This indicates that NB maintains a steady trade-off between identifying all relevant instances and maintaining accuracy (see Table 4).

Model With FL shows reduced model performance for classification of activities as compared to centralized methods as shown in Tables 2 and 3. This is mainly because it deals with varied and scattered data and faces limits in computational power and the need to protect privacy which make harder for the model to learn effectively from such a diverse range of data. The standout performer is the proposed FeL-MAR model which surpasses all others with impressive accuracy and precision both above 89%. Its recall and F1-score are equally commendable indicating an exceptional ability to correctly identify positive instances while maintaining a high overall accuracy. Furthermore its lower RMSE and MAE values suggest that its predictions are not only accurate but also close to the actual values indicating high reliability. The graph mentioned in Figs. 9 and 10 illustrate the training and validation accuracy of the FeL-MAR model. The training accuracy shows a general indication of good performance on the training dataset over time. The validation accuracy follows a similar trend that the model generalizes well on the unseen data. Loss Vs Epoch graph illustrates the training and validation loss of the FeL-MAR model. Both training and validation losses decrease over time, which is a positive indicator of model learning. The convergence of training and validation losses suggests that the model is not over-fitting and is learning general patterns.

Table 3 provides a further detailed performance evaluation of the federated framework. The performance of the used DL model, named as Fel-MAR, is compared with other ML models, separately for both the houses, and presented in this table. In addition to this, Fig. 11 illustrates activity-wise accuracy for Resident 1 in House A, for 27 different activities. Together from Table 3 and Fig. 11, it is evident that Fel-MAR does not compromise with the performance, while ensuring added benefits of federated framework.

5. Conclusion

In conclusion, FeL-MAR emerges as a significant step forward in smart home technology particularly in multi-resident settings to intelligently recognize and adapt to various activities while upholding user privacy and reducing reliance on centralized data processing. The proposed system not only efficiently recognizes varied resident activities but also respects the privacy of the individual residents. Furthermore also we get the significant accuracy while emphasizing privacy preservation during data processing locally.

In the future, we will concentrate on evaluating the effectiveness of FeL-MAR in various scenarios. This includes different user behaviors, activities and house configurations as well as the scalability of FeL-MAR Model. As smart homes become more interconnected with numerous IoT devices. It is essential to ensure that FeL-MAR can scale efficiently while maintaining high levels of performance and privacy. We will also incorporate the feedback mechanisms for human activity data generated within smart environments. This approach will enhance the applicability of such data and incorporate context-aware policies, ensuring the privacy of data remains under house control.

CRediT authorship contribution statement

Abisek Dahal: Writing – review & editing, Writing – original draft, Resources, Methodology, Investigation, Formal analysis, Data curation,

Conceptualization. **Soumen Moulik:** Writing – review & editing, Writing – original draft, Supervision, Formal analysis, Conceptualization. **Rohan Mukherjee:** Writing – review & editing, Writing – original draft, Validation, Supervision, Conceptualization.

Funding

The authors acknowledge that they received no funding for conducting this research.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] K. Skianis, A. Giannopoulos, P. Gkonis, P. Trakadas, Data aging matters: Federated learning-based consumption prediction in smart homes via age-based model weighting, Electronics 12 (14) (2023) 3054.
[2] S. Lee, D.H. Choi, Federated reinforcement learning for energy management of multiple smart homes with distributed energy resources, IEEE Trans. Ind. Inform. 18 (1) (2020) 488–497.
[3] P.M.S. Sánchez, A.H. Celdrán, T. Schenk, A.L.B. Iten, G. Bovet, G.M. Pérez, B. Stiller, Studying the robustness of anti-adversarial federated learning models detecting cyberattacks in iot spectrum sensors, IEEE Trans. Dependable Secure Comput. (2022).
[4] 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.
[5] D. Cheng, L. Zhang, C. Bu, X. Wang, H. Wu, A. Song, ProtoHAR: prototype guided personalized federated learning for human activity recognition, IEEE J. Biomed. Health Inf. (2023).
[6] Y. Zhao, P. Barnaghi, H. Haddadi, Multimodal federated learning on iot data, in: 2022 IEEE/ACM Seventh International Conference on Internet-of-Things Design and Implementation, IoTDI, IEEE, 2022, pp. 43–54.
[7] Y. Cao, D. Liu, S. Zhang, T. Wu, F. Xue, H. Tang, T-FedHA: A trusted hierarchical asynchronous federated learning framework for internet of things, Expert Syst. Appl. 245 (2024) 123006.
[8] V. Dentamaro, V. Gattulli, D. Impedovo, F. Manca, Human activity recognition with smartphone-integrated sensors: A survey, Expert Syst. Appl. (2024) 123143.
[9] N. Koblitz, A. Menezes, S. Vanstone, The state of elliptic curve cryptography, Des. Codes Cryptogr. 19 (2000) 173–193.
[10] M. Vrigkas, C. Nikou, I.A. Kakadiaris, A review of human activity recognition methods, Front. Robot. AI 2 (28) (2015).
[11] Y. Liu, W. Huang, S. Jiang, B. Zhao, S. Wang, S. Wang, Y. Zhang, TransTM: A device-free method based on time-streaming multiscale transformer for human activity recognition, Def. Technol. (2023).
[12] Y. Li, G. Yang, Z. Su, S. Li, Y. Wang, Human activity recognition based on multienvironment sensor data, Inf. Fusion 91 (2023) 47–63.
[13] LiFen Gu, Amin Mohajer, Joint throughput maximization, interference cancellation, and power efficiency for multi-IRS-empowered UAV communications, Signal Image Video Process. 18 (5) (2024) 4029–4043.
[14] Q. Wang, W. Li, A. Mohajer, Load-aware continuous-time optimization for multi-agent systems: Toward dynamic resource allocation and real-time adaptability, Comput. Netw. 250 (2024) 110526.
[15] N. Mani, P. Haridoss, B. George, Smart suspenders with sensors and machine learning for human activity monitoring, IEEE Sens. J. (2023).
[16] N.A. Choudhury, B. Soni, An efficient and lightweight deep learning model for human activity recognition on raw sensor data in uncontrolled environment, IEEE Sens. J. (2023).
[17] A. Jain, V. Kanhangad, Human activity classification in smartphones using accelerometer and gyroscope sensors, IEEE Sens. J. 18 (3) (2017) 1169–1177.
[18] N. Twomey, T. Diethe, X. Fafoutis, A. Elsts, R. McConville, P. Flach, I. Craddock, A comprehensive study of activity recognition using accelerometers, Informatics 5 (2) (2018) 27, http://dx.doi.org/10.3390/informatics5020027.
[19] Andrey Ignatov, Real-time human activity recognition from accelerometer data using convolutional neural networks, Appl. Soft Comput. (ISSN: 1568-4946) 62 (2018) 915–922, http://dx.doi.org/10.1016/j.asoc.2017.09.027.
[20] T. Wang, D.J. Cook, Multi-person activity recognition in continuously monitored smart homes, IEEE Trans. Emerg. Top. Comput. 10 (2) (2022) 1130–1141, http://dx.doi.org/10.1109/TETC.2021.3072980.
[21] A.K. Panja, A. Rayala, A. Agarwala, S. Neogy, C. Chowdhury, A hybrid tuple selection pipeline for smartphone based human activity recognition, Expert Syst. Appl. 217 (2023) 119536.
[22] 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, ISPA/IUCC/BDCloud/SocialCom/SustainCom, IEEE, 2018, pp. 1103–1111.
[23] 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.
[24] X. Ouyang, Z. Xie, J. Zhou, G. Xing, J. Huang, ClusterFL: A clustering-based federated learning system for human activity recognition, ACM Trans. Sensor Netw. 19 (1) (2022) 1–32.
[25] 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.
[26] 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 Trans. Mobile Comput. 22 (6) (2021) 3318–3332.
[27] Y. Li, X. Wang, L. An, Hierarchical clustering-based personalized federated learning for robust and fair human activity recognition, in: Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol. 7, (1) 2023, pp. 1–38.
[28] Q. Wu, X. Chen, Z. Zhou, J. Zhang, Fedhome: Cloud-edge based personalized federated learning for in-home health monitoring, IEEE Trans. Mob. Comput. 21 (8) (2020) 2818–2832.
[29] S.O. Bursa, O.D. Incel, G.I. Alptekin, Personalized and motion-based human activity recognition with transfer learning and compressed deep learning models, Comput. Electr. Eng. 109 (2023) 108777.
[30] M.D. de Souza, C.R.S. Junior, J. Quintino, A.L. Santos, F.Q. da Silva, C. Zanchettin, Exploring the impact of synthetic data on human activity recognition tasks, Procedia Comput. Sci. 222 (2023) 656–665.
[31] R. Yuan, J. Wang, The human continuity activity semi-supervised recognizing model for multi-view IoT network, IEEE Internet Things J. (2023).
[32] F. Luo, S. Khan, Y. Huang, K. Wu, Binarized neural network for edge intelligence of sensor-based human activity recognition, IEEE Trans. Mob. Comput. 22 (3) (2021) 1356–1368.
[33] J. Li, H. Xu, Y. Wang, Multi-resolution fusion convolutional network for open set human activity recognition, IEEE Internet Things J. (2023).
[34] T. Chen, L. Mo, Swin-fusion: swin-transformer with feature fusion for human action recognition, Neural Process. Lett. 55 (8) (2023) 11109–11130.
[35] M.G. Herabad, Communication-efficient semi-synchronous hierarchical federated learning with balanced training in heterogeneous IoT edge environments, Internet Things 21 (2023) 100642.
[36] A. Sezavar, R. Atta, M. Ghanbari, DCapsNet: Deep capsule network for human activity and gait recognition with smartphone sensors, Pattern Recognit. 147 (2024) 110054.
[37] M.M. Islam, S. Nooruddin, F. Karray, G. Muhammad, Multi-level feature fusion for multimodal human activity recognition in internet of healthcare things, Inf. Fusion 94 (2023) 17–31.
[38] Y.A. Andrade-Ambriz, S. Ledesma, M.A. Ibarra-Manzano, M.I. Oros-Flores, D.L. Almanza-Ojeda, Human activity recognition using temporal convolutional neural network architecture, Expert Syst. Appl. 191 (2022) 116287.
[39] Q. Li, W. Huangfu, F. Farha, T. Zhu, S. Yang, L. Chen, H. Ning, Multi-resident type recognition based on ambient sensors activity, Future Gener. Comput. Syst. 112 (2020) 108–115.
[40] H. Alemdar, H. Ertan, O.D. Incel, C. Ersoy, ARAS human activity datasets in multiple homes with multiple residents, in: 2013 7th International Conference on Pervasive Computing Technologies for Healthcare and Workshops, IEEE, 2013, pp. 232–235.

Abisek Dahal, currently pursuing his Ph.D. in the Department of Computer Science and Engineering at the National Institute of Technology Meghalaya, India. He Completed B.Tech and M.Tech in Engineering from National Institute of Technology Sikkim. His areas of expertise and research interests includes Machine Learning, Human Activity Recognition, Federated Learning, Internet of Things, Edge Computing, Machine Learning and Wireless Network.

Soumen Moulik received the Ph.D. degree in computer science from the Indian Institute of Technology Kharagpur, India. During his Ph.D., he has made significant contributions on the topic of health severity-based QoS provisioning in wireless body area network. He is currently an Assistant Professor in the Department of Computer Science and Engineering at National Institute of Technology, Meghalaya, India. His research interests include networking and communication aspects of wireless body/personal area networks, Internet-of-Things.

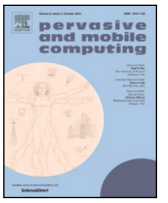
Rohan Mukherjee has completed his MS and Ph.D. from Indian Institute of Technology Kharagpur in the domain of visual information processing and system design. He has also post-doctoral research experience in cutting-edge approaches of Machine Learning and AI. He is currently working as an Associate Professor in Department of Management Information Systems and Analytics at International Management Institute Kolkata. Prior to joining IMI Kolkata, he has worked at IIM Bodh Gaya, IIM Jammu and Goa Institute of Management. Rohan has published consistently in reputed journals and IEEE conferences in his area of expertise. His current research interest includes data analytics and machine learning and their application in business.

Pervasive and Mobile Computing 104 (2024) 101972

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc



Edge human activity recognition using federated learning on constrained devices☆

Angelo Trotta [a,*], Federico Montori [a,b], Leonardo Ciabattini [a], Giulio Billi [a], Luciano Bononi [a], Marco Di Felice [a,b]

[a] University of Bologna, Italy
[b] Advanced Research Center for Electronic Systems, Bologna, Italy

ARTICLE INFO

Keywords:
Human Activity Recognition (HAR)
Internet of Things (IoT)
Edge computing
Federated Learning

ABSTRACT

Human Activity Recognition (HAR) using wearable Internet of Things (IoT) devices represents a well investigated researched field encompassing various application domains. Many current approaches rely on cloud-based methodologies for gathering data from diverse users, resulting in the creation of extensive training datasets. Although this strategy facilitates the application of powerful Machine Learning (ML) techniques, it raises significant privacy concerns, which can become particularly severe given the sensitivity of HAR data. Moreover, the labeling process can be extremely time-consuming and even more challenging for IoT wearable devices due to the absence of efficient input systems. In this paper, we address both aforementioned challenges by designing, implementing, and validating edge-based Human Activity Recognition (HAR) systems that operate on resource-constrained IoT devices, which relies on the utilization of Self-Organizing Maps (SOM) for activity detection. We incorporate a feature selection process before training to reduce data dimensionality and, consequently, the SOM size, aligning with the resource limitations of wearable IoT devices. Additionally, we explore the application of Federated Learning (FL) techniques for HAR tasks, enabling new users to leverage SOM models trained by others on their respective datasets. Our federated Extreme Edge (EE)-aware HAR system is implemented on a wearable IoT device and rigorously tested against state-of-the-art and experimental datasets. The results demonstrate that our C++-based SOM implementation achieves a consistent reduction in model size compared to state-of-the-art approaches. Furthermore, our findings highlight the effectiveness of the FL-based approach in overcoming personalized training challenges, particularly in onboarding scenarios.

1. Introduction

The widespread adoption of wearable Internet of Things (IoT) devices, equipped with Inertial Measurement Units (IMU) sensors, has fueled the development of Human Activity Recognition (HAR) systems across diverse application domains, spanning healthcare, fitness, and smart mobility. In telerehabilitation systems, HAR solutions enable fine-grained monitoring of motor activities performed by remotely connected patients [2,3]. Within the fitness domain, they support the synthesis of advanced statistics by combining sports performance metrics with physiological responses [4,5]. Likewise, numerous smart city mobile applications leverage HAR techniques for tasks such as mobility trace collections and parking spot recommendations [6,7]. Despite the varied techniques

☆ A preliminary version of this paper has appeared in *IEEE SMARTCOMP 2023* [1].

* Corresponding author.
E-mail address: angelo.trotta5@unibo.it (A. Trotta).

https://doi.org/10.1016/j.pmcj.2024.101972
Received 22 December 2023; Received in revised form 29 May 2024; Accepted 3 August 2024
Available online 8 August 2024

employed for activity recognition, most of the existing works follow a common, cloud-centric architecture [8]: data gathered from different users are centralized, and a robust Machine Learning (ML) model is trained on-situ [5]. While this methodology leads to the creation of extensive datasets [9] and enhances the accuracy of ML models, it introduces potential concerns related to privacy and scalability. Indeed, HAR raw data, being inherently sensitive, may expose users to vulnerabilities even with anonymization techniques due to the possibility of re-identification attacks [10]. Furthermore, supervised ML techniques require manual labeling of raw IMU data, which can be particularly cumbersome for wearable IoT devices that lack efficient input systems.

To address privacy concerns, the adoption of edge computing has become a prevalent technique, enabling data processing near IoT devices [11]. However, leveraging edge computing for Human Activity Recognition (HAR) systems introduces novel challenges that must be considered. Indeed, wearable IoT devices, commonly utilized for HAR data collection, are inherently constrained in terms of computational and storage resources [12]. Consequently, many complex, supervised Deep Learning (DL) techniques proposed for HAR may be impractical to implement. Moreover, the restriction on data sharing poses a unique challenge, requiring users to train on their own data for all the activities they aim to monitor. While this approach may not always lead to a performance drop [13], it may significantly affect the application of HAR systems, given the cost and time challenges associated with the training phase, and specifically the aforementioned constraints of the labeling process.

In this paper, we address the design, implementation, and validation of an effective edge-friendly solution for HAR tasks. Our innovative methodology integrates established and novel data processing techniques to address the privacy and scalability challenges while enhancing activity detection accuracy. Specifically, our solution leverages unsupervised Self-Organizing Maps (SOM) to minimize manual data labeling [14]. Its implementation on resource-constrained IoT devices is further facilitated by a feature selection phase aimed at balancing the SOM size with HAR accuracy. Additionally, we tackle the onboarding challenge, occurring when new users join the HAR task without prior training. In this regard, we explore the application of Federated Learning (FL) techniques on top of SOM models [15,16], in order to allow users to benefit from HAR knowledge shared by other peers without accessing raw data, hence preserving their privacy. Our HAR technique is implemented on a real IoT wearable prototype (ESP32 M5Stack), encompassing the full architecture with edge data management functionalities. The C++ implementation of SOM ensures a model size reduction of up to 8 times compared to state-of-the-art solutions [17]. We validate our approach using the popular UCI HAR dataset and a newly collected dataset through our prototype when worn by different users. In both cases, our approach demonstrates comparable performance to state-of-the-art DL architectures, with only an 8% difference. The FL solution enhances personalized learning for onboarding scenarios and remains comparable to centralized models while avoiding the need to share raw user data.

Specifically, four main contributions are provided in this paper:

- We design and develop an Extreme Edge (EE) Human Activity Recognition (HAR) system based on the Self Organizing Map (SOM) technique, complemented by a feature selection phase. We investigate the trade-off considerations involving SOM size, input feature dimensions, and HAR accuracy.
- We extend the application of the solo ML technique to a distributed scenario, accommodating multiple users eager to participate in the same HAR task without sharing their raw sensor data. To achieve this, we explore the integration of Federated Learning (FL) techniques with the SOM model, incorporating weight exchange and updates.
- We implement the proposed solution on a prototype IoT device, integrated into an edge–cloud framework; the prototype has been used to build a small-case HAR training set which is made available online for further exploration.
- We compare the performance of the EE-HAR system against state-of-the-art DL models based on centralized learning, This evaluation encompasses both our dataset and the UCI dataset. Furthermore, we assess the enhancements brought about by the FL approach in our proposed solution.

Compared to our previous work [1], this extended version includes several significant enhancements and novel contributions. We have designed a FL approach specifically for HAR systems in extreme edge scenarios that builds on SOMs, handling the limited computational resources and data privacy concerns inherent in these environments. Additionally, we have implemented a three-layer architecture comprising extreme edge devices, edge nodes, and a cloud node, facilitating improved data processing, model training, and system scalability. Our performance analysis includes a detailed examination of the onboarding scenarios where new, untrained users are introduced into the system. This is extremely important, as in real HAR scenarios it is unlikely that everyone participate in the FL training phase at the same time, as in most of the FL benchmarks in literature. Moreover, in this paper we have published a new HAR dataset featuring multiple subjects, enhancing the reproducibility of our experiments and providing a valuable resource for the research community.

The rest of the paper is structured as follows. Section 2 reviews the state of the art of HAR techniques focusing on edge-computing solutions. Section 3 presents the proposed HAR architecture and the interactions among components. Section 4 focuses on the learning techniques. Section 5 introduces the extension to distributed environments based on FL. Section 6 details the prototype implementation on the target IoT device. Section 7 describes the performance evaluation utilizing both the UCI and our dataset. Section 8 concludes the work.

2. Related works

HAR emerged in the academic domain during the 1980s, gaining prominence in the past two decades, notably within sectors such as elder care, healthcare, and ambient intelligence scenarios [18]. HAR approaches broadly fall into two categories: knowledge-based and data-driven. The former explores semantic correlations among event types and activity classes [19], while the latter entails the

collection and subsequent analysis of sensory data. Activity recognition can be viewed as a pattern-matching problem, and various DL techniques have been suggested to address it [20]. Despite the availability of various public HAR datasets [21,22], the proliferation of novel applications underscores the need for finely tuned and personalized datasets. The challenge is accentuated in supervised DL models, which necessitate labeled data for dataset creation. In order to alleviate such challenge with HAR systems, researchers are delving into both unsupervised and semi-supervised classification methods [23].

In the domain of unsupervised approaches, the work presented in [24] introduces a cluster-based framework designed to label multidimensional inertial signals. This approach includes through two distinct phases: firstly, a recurrent auto-encoder extracts meaningful spatiotemporal features from the inertial data. Subsequently, in the second phase, these extracted features seamlessly integrate into an unsupervised clustering algorithm, facilitating the prediction of unlabeled signals. A comprehensive survey on the application of clustering techniques to inertial signals can be found in [25]. Among the semi-supervised approaches, Unsupervised Domain Adaptation (UDA) techniques have emerged as potent tools for distinguishing new users from existing ones and aligning their unlabeled data with features present in labeled datasets. In this landscape, the SALIENCE architecture, described in [23], employs discriminators at both the feature and sample label levels. Additionally, an Attention-based Neural Network is employed to identify sensors housing strongly discriminating features, prioritizing their significance in the learning process. Another study, presented in [26], introduces an architecture comprising a bidirectional Generative Adversarial Neural Network (Bi-GAN) coupled with Kernel Mean Matching (KMM), in order to facilitate the transfer of activity detection knowledge across datasets characterized by heterogeneous feature spaces.

FL represents a compelling approach for HAR, providing a dual advantage of effective learning and privacy preservation. In [16], the authors introduce a Federated Learning System for HAR (HARFLS), relying on federated averaging and incorporating a novel Perceptive Extraction Network (PEN). The study illustrates that even the least performing DL model within this framework outperforms the best performing ML technique in terms of average F1 score. In other solutions reported in [13,27,28], authors describe the integration of FL with semi-supervised learning and clustering methodologies, addressing challenges associated with unlabeled data and non-independently and identically distributed data.

In the following, we focus on the research studies that are most related to the contributions of our study. In [15], the authors introduce two federated models – a Deep Neural Network (DNN) and a softmax regression – customized to operate on vitalized edge devices with constrained resources. Moving on to [29], the GWEP method is presented as an FL approach rooted in model compression. This strategy employs joint quantization and model pruning, addressing the balance between efficiency and computational constraints to align with the capabilities of resource-constrained devices. In our previous study [12], we investigated the application of dynamic range quantization. This technique is implemented to reduce the size of the Convolutional Neural Network (CNN) loaded onto a microcontroller unit. [30] delves into the efficiency of various binarized neural networks (BNN) for classifying human activities, which are extreme examples of quantization: they constraint weights, and activation functions to binary values. Exploring the classification of daily gesture samples collected through an inertial ring and bracelet, [31], employs a spectrum of supervised and unsupervised ML methods. The study investigates the efficacy of unsupervised techniques, such as the K-Means and Gaussian Mixture Model, alongside their supervised counterparts. In [17] a SOM and a Convolutional Neural Network (CNN) collaborate on a microcontroller unit, proving the viability of feature extraction methodologies in resource-constrained environments.

Our paper differs from previous works in the following aspects: (*i*) we showcase the implementation of a novel SOM library for the Arduino Software Development Kit (SDK), resulting in a substantial reduction of the model size compared to [17]; (*ii*) we introduce a pre-processing mechanism, consisting of ANOVA-F feature selection, enabling further reduction in the amount of data processed on the EE; (*iii*) we meticulously assess the trade-off between SOM accuracy and model size across various configurations of both techniques; (*iv*) we implement FL as a privacy-aware method for onboarding new users into the HAR system.

3. System architecture for cooperative HAR

In this Section, we present the architecture of our HAR system, specifically designed for wearable, resource-constrained IoT devices. The system architecture includes a cooperative module to address the challenges of efficient and privacy-aware computing in environments with limited computational power, storage, and data processing capabilities. The architecture has been designed to fulfill the following requirements:

- *Flexibility*: users can define and add new activities dynamically.
- *Customizability*: the system adapts to different user needs, allowing for personalized training phases.
- *Ease of Use*: the system minimizes manual data labeling, simplifying the user experience.
- *Cooperation*: enables multiple users to contribute to the model training process, enhancing the overall system performance and accuracy.

To achieve these objectives, our architecture incorporates three types of computational nodes: the IoT wearable device (the EE device), the user's edge node, and an external server which includes a cooperative functionality for enhanced model training. The HAR process is divided into three phases: data gathering, model training, and HAR inference. These computational phases are allocated statically to a device: data gathering and HAR inference take place on the wearable device, while the training phase is conducted betweeen the central server and the user's edge node, depending on the type of training (single, federated or centralized, more on this in Section 5). The architecture is depicted in Fig. 1. Below, we provide detailed descriptions for each phase.

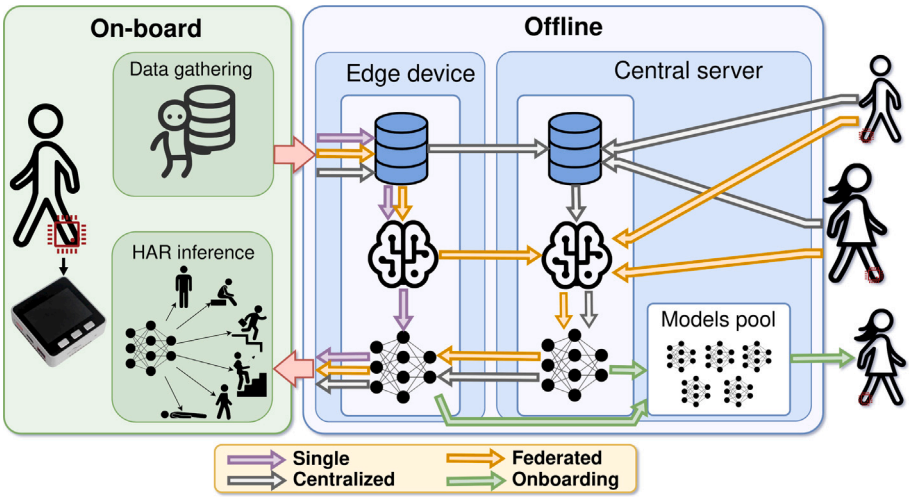


Fig. 1. The proposed HAR architecture. The purple path indicates the *single* approach; the gray path specify the *centralized* approach; the yellow path indicates the *federated* system. With the green arrows we specify the *on-boarding* process where new users use pre-trained model.

- *Data gathering*: in this initial phase, the IoT device actively samples raw sensor data while the user performs various activities. The system utilizes IMU sensors, including accelerometers, gyroscopes, and magnetometers, that operate at a predetermined sampling rate. During this stage, the system is designed to operate autonomously, eliminating the need for user intervention for data labeling. The raw data collected from the IMU sensors is then wirelessly transmitted to an edge node. Here, it is collected into unlabeled HAR datasets for subsequent model training phases.
- *Model training*: this phase is pivotal in developing the ML model using the HAR dataset gathered from the IoT devices. Initially, the IMU data undergoes a preprocessing stage where outliers and missing values are identified and removed. To enhance data quality, noise filtering techniques, particularly the Butterworth low-pass filter, are applied. Subsequently, feature extraction is carried out on the sensor data, focusing on both frequency and temporal domains. This process is elaborated in Section 7.1. The training of ML models is conducted using the preprocessed datasets. Our approach utilizes the SOM technique, detailed in Section 4.1, which organizes the data into K distinct classes. The number of classes, K, is either predetermined or dynamically calculated from the data using methods like the elbow method [32]. At this stage, we assign semantic labels to each class, correlating them with specific human activities. Notably, the labeling occurs post the application of the unsupervised SOM method, differing from the example-based labeling in supervised learning. To enhance efficiency, we have incorporated a feature selection phase using the ANOVA-F technique (see Section 4.2). This step aims to minimize the number of input features for the SOM, thereby reducing the processing load.
  The model training phase offers two distinct methodologies: single and cooperative training:
  - *Single Training*: in this approach, individual users train the SOM model solely with their data, ensuring privacy as the data does not leave the user's edge device. This flow is depicted in Fig. 1 with purple arrows.
  - *Cooperative Training*: this method can be executed with two different methods:
    - *Data-Level Cooperation (Centralized Model)*: data from various users are collected on the external server to train a comprehensive SOM model. This centralized model benefits from the diverse data pool, potentially enhancing accuracy and generalizability. This is shown in Fig. 1 with gray arrows. We can see here that the data are collected in the central server and the ML model is trained using the aggregated dataset from all the cooperating users.
    - *Training-Level Cooperation (Federated System)*: here, each device independently trains a local model. Instead of raw data, only SOM model parameters are shared with the server, which then aggregates these to refine the global model. This federated approach maintains data privacy by keeping raw data localized. In Fig. 1 this flow is indicated with yellow arrows. Notably, here the raw data are not shared.

  The system can generate multiple SOM models for varying network sizes, with the optimal model being one that balances high accuracy with the storage constraints of the IoT device. Upon completion of the training phase, these trained SOM models are stored in a *models pool* on the central server, which acts as a central repository (green arrows in Fig. 1). This setup ensures that the models are readily available for deployment to users' IoT devices, facilitating the practical application of the HAR system in diverse real-world environments.
- *HAR inference*: post-training, the trained SOM model (either from local or cooperative training), along with the selected feature set, is selected from the *model pool* and it is automatically transferred back to the IoT device. This pool enables also the onboarding feature (see Section 5) where new users utilize a pre-trained model, avoiding the data gathering and model training

phases It is important to note that while this process ensures the efficient deployment of the model, the specific aspects of protection and security related to model transfer and deployment are not covered in this work, while our focus is primarily on the functionality and performance of the HAR system. The device processes real-time IMU data to extract features, which are then fed into the SOM for activity classification. Once the activities are classified, the HAR system communicates the results to the user. This communication can be facilitated through various output peripherals, such as a display screen, or transmitted to other devices for further use or analysis.

4. Learning models for the EE

In this Section, we present the ML and pre-processing techniques used in the model training module of Fig. 1. We rely on Self Organizing Maps (SOMs) for human activity classification and the ANOVA-F technique for feature reduction.

4.1. Self-organizing map

The self-organizing map (SOM) [14] is a type of unsupervised learning algorithm that can be used for dimensionality reduction and clustering. SOMs are a form of artificial neural network that can map high-dimensional input data onto a lower-dimensional output space while preserving the topological properties of the input data. In our HAR system, we used SOMs as ML techniques to address the HAR task for the reasons below:

- They are lightweight and computationally efficient, making them well-suited for deployment on constrained devices with limited computational power and storage capacity [17]. This is in contrast to other ML/DL techniques which can have high storage requirements and long computational times. KNN, for example, requires storing all training data; FNN and SVM, on the other hand, have a complex implementation of transfer and kernel functions, respectively.
- They are well-suited for unsupervised learning, which is particularly useful for HAR systems as pointed out in Section 2. Indeed, SOMs can discover patterns or structures in the data without any labeled information, which can be useful for identifying features or patterns in the sensor data that are associated with different activities.
- They can be easily customized for specific sensor data formats or feature reduction methods, which is essential for deployment on constrained devices. By optimizing the SOM model size for the specific constraints of the device, we have a strategy to address the trade-off between performance, energy consumption and storage.

More in detail, a SOM is a distinct type of artificial neural network that employs competitive learning for training. In this approach, nodes compete for the opportunity to respond to specific subsets of input data, as opposed to error-correction learning methods (e.g., backpropagation with gradient descent) utilized by other artificial neural networks. Like most artificial neural networks, SOMs support two primary modes: training and mapping. The training phase uses an input data set, or "input space", to create a lower-dimensional representation known as the "map space". The mapping phase then employs this generated map to classify additional input data. Typically, the training process aims to represent an input space with p dimensions as a two-dimensional map space. An input space consists of p features, each representing a dimension. The map space is composed of *nodes* or *neurons*, which are organized in a two-dimensional rectangular grid of size $N \times N$. Each node n_i in the map space in the grid can be hence identified via its coordinates $\langle x_i, y_i \rangle$, with $0 < i < N$. The node n_i is linked to a weight vector w_i of p dimensions, that signifies the node's position in the input space. Although nodes in the map space remain stationary, the training process involves adjusting weight vectors towards the input data (by minimizing a distance metric such as *Euclidean* distance) without disrupting the topology derived from the map space. Once trained, the map can classify additional observations in the input space by identifying the node with the nearest weight vector (i.e., the smallest distance metric) to the input space.

During the initialization of the SOM algorithm, the neurons' weights are assigned random values. Then, iterating over the input data, for each training example a, the Best Matching Unit (BMU) is calculated:

$$\text{bmu} = \arg\min_i \{\|a - w_i\|\}$$

The weight of the BMU node, and its neighborhood, is then updated as follows:

$$w_i = w_i + \eta(t) \cdot h_{i,\text{bmu}}(t) \cdot (a - w_i)$$

Here, $\eta(t)$ is the learning rate at learning iteration t defined with a learning rate decay rule $\eta(t) = \eta_0 \cdot e^{(-t/\hat{\eta})}$, with η_0 and $\hat{\eta}$ as custom parameters. The neighborhood kernel function $h_{i,\text{bmu}}(t)$ defines the neighborhood size and is defined by:

$$h_{i,\text{bmu}}(t) = e^{-\frac{d^2_{i,\text{bmu}}}{2 \cdot \theta^2(t)}}$$

where $\theta(t) = \theta_0 \cdot e^{-t/\hat{\theta}}$ defines the neighborhood size decay rule, with θ_0 and $\hat{\theta}$ as custom parameters. The distance $d_{i,j}$ between two nodes is calculated as a Manhattan distance, i.e., $d_{i,j} = |x_i - x_j| + |y_i - y_j|$.

The critical factors for deployment on a constrained EE device are reflected by the number of weights w_i. The size of the SOM map is thus defined by the number of neurons, i.e., the grid $N \times N$, and the number of features p. Consequently, the total model size is $\mathcal{O}(N^2 \cdot p)$. In the following Section, we introduce a feature reduction mechanism to decrease the value of p. Additionally, in Section 7.4, we evaluate various SOM sizes to attain satisfactory classification performance while enabling model deployment on constrained EE devices.

4.2. ANOVA-F feature reduction

Feature reduction is an essential step in building efficient ML models for HAR. Indeed, complex ML models tend to use all possible features as inputs and internally disregard features that have little to no influence in discriminating the predicted class. Vice versa, for wearable devices with constrained resources, it is crucial to reduce the number of features *before* building the final model without compromising its accuracy. Several popular methods can be employed for feature reduction, such as Principal Component Analysis (PCA); however, they demand high computation and storage resources during the inference stage.

In the EE scenario, we chose an offline method that is able to reduce the number of features, i.e. the ANOVA-F technique, which can be conducted a priori and not during the inference phase. This technique is a statistical method that can identify the most significant features of the model [33]. In order to calculate the ANOVA-F value of a feature f we first calculate, for each of the K classes, the variance of that feature within such class, then we divide it by the variance of such feature over the whole dataset. More formally, The ANOVA-F value of feature f for class $c \in K$ is given by:

$$\mathrm{An}_c(f) = \frac{\sigma^2(f)_c}{\sigma^2(f)}$$

Ideally, a low value of $\mathrm{An}_c(f)$ means that f is highly discriminant for class c, because the values of f do not change significantly among the members of class c compared to how much they change normally in the dataset. Conversely, values of $\mathrm{An}_c(f)$ that approach 1.0 or above signify that feature f is likely to just add noise for class c. In order to come up with a single ANOVA-F value for each feature f, for this reason, an aggregation function over all instances of ANOVA-F values of f for all K classes is implemented. We conducted our experiments by selecting the *average* and the *minimum* as aggregation functions, which are defined as follows:

$$\mathrm{An}_{avg}(f) = \frac{\sum_{c \in C} \mathrm{An}_c(f)}{K},$$

$$\mathrm{An}_{min}(f) = min\{\mathrm{An}_c(f) | c \in K\}$$

The first associates each feature with an ANOVA-F value by taking into account its overall influence over all classes, while the second only considers the class for which the feature is most influential. Once calculated, we can rank all the p features in ascending order by their ANOVA-F value and keep only the ones ranked best. Therefore, we introduce the ANOVA-F Threshold (An_{thr}), which is the ANOVA-F value below which we keep all the features. This variable becomes a relevant parameter of our system, which depends on the available storage on the IoT device and has to be balanced with other parameters in order to achieve the best accuracy, such as the SOM size.

5. Collective context

The architecture we have detailed in Section 3 presents a versatile and adaptive framework for EE-friendly HAR systems. It incorporates SOM techniques and offers both single and cooperative training methodologies to cater to diverse user needs and privacy concerns. Here, by envisioning the usage of the device in a real context, we could imagine that the owner of a device is introduced in a scenario where a number of other individuals are using the same device. In such a scenario, assuming that each device hosts a model with a fixed shape, how do we train each of the models? We could, on the one hand, deliver a clean slate device so that the users shall perform the training process themselves. On the other hand, we could take advantage of the collective intelligence and perform the model training as a community of users, where all of them share their sensor data to train a single, general model. This however raises privacy concerns, which are unacceptable in certain use cases, unless adopting an FL approach.

More formally, in a collective context, we must select one from three different configurations, as outlined in Fig. 2.

Single. In this case, each device is delivered with an untrained model. Users are then committed to performing training in autonomy, generating a local model that is meant to be tailored to the users themselves. Given its specificity, if the training is conducted correctly, the model is likely to show a high accuracy on the activities of the owner and, since the model is local, it also maximizes her privacy.

Centralized. In this case, each device comes with a pre-trained model, which is generated by the collective training of a number of users who shared their training data and made up a sufficiently heterogeneous training dataset. As sensor data is completely disclosed, a central entity can interpret it as a single dataset and perform the training process only once, sharing the model with all participants. This solution allows users to avoid the local training process, however, in certain cases (i.e. healthcare) sharing plain sensor data with a central entity is not acceptable.

Federated. In this case, each device comes with an untrained model, and all users participate in an FL process mediated by a central entity. In this paper, we implement FL with SOM, which is architecturally similar to classic FL implementations on neural networks using FedAvg [34]. However, unlike the classic FedAvg, our aggregation strategy is unweighted, meaning that each user will have the same impact on the results regardless of the size of their dataset. The rationale behind this choice is to avoid excessively large datasets taking over, as they would cause the results to be too representative of a few users. This can be quite counterproductive in HAR systems, given their tendency to be personalized. This method resembles the Centralized structure, still preserving privacy and, presumably, acknowledging a small drop in performance. In this case, the owner of the device is still committed to performing the training process locally, however, the process results in a single model, which is not tied to a single individual.

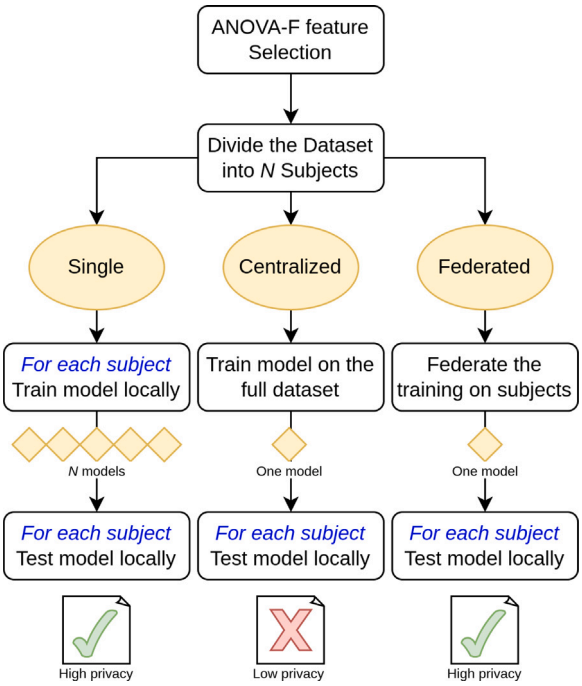


Fig. 2. Flow chart of possible application scenarios: single, centralized, and federated.

Onboarding process

The previous considerations outlined in this Section seem to privilege a *Single* approach, as HAR upon sensor data is strongly dependent on the individual who is performing the training [7]. However, it is likely that such a model would only suit the owner and cannot be efficiently reused. In fact, in a dynamic and realistic scenario, we should consider the onboarding issues. With "onboarding" we mean the process through which a new individual becomes part of the network and, in our case, is given a new device. Now, with a *Single* approach, this means that users would need to train the model from scratch, which, in certain cases, can be either a hassle or unfeasible, because maybe the users are not expert enough to conduct a meaningful training process. This is when a collective model (either through a *Centralized* or a *Federated* approach) comes in handy because it allows a newcomer to utilize a product off-the-shelf, versus performing a tedious and difficult configuration process. For this reason, as reflected in Section 7 we evaluate the collective approach, both in (i) an *offline* context, where all participants take part in the training process, and in (ii) an *onboarding* context when the system is at steady state and a new user, incapable of training a model, enters the system. More in detail, the onboarding process is conducted in the following way, with respect to the approaches:

Single The newcomer onboards by adopting a randomly chosen model from one of the existing users.
Centralized The newcomer onboards by adopting the centralized model.
Federated The newcomer onboards by adopting the federated model.

6. Implementation

The deployment of the HAR system depicted in Fig. 1 is based on the *M5Stack*,[1] which is a modular, stackable, and programmable device designed to support pervasive IoT applications. It is equipped with multiple IMU sensors, including an accelerometer, gyroscope, and magnetometer, making it ideal for HAR data collection tasks. The computational unit is based on the popular ESP32 microcontroller with 240 MHz dual-core processors and 320 kB of RAM memory. We choose this device to validate our HAR architecture due to its compact and energy-efficient profile; however, we emphasize that our firmware can be easily adapted for other IoT boards that support the Arduino SDK.

During the experiments, we positioned the device on users' ankles as this location provides optimal gait analysis, as reported by [35]. Generally speaking, the development of ML models, including the SOM, on micro-controllers is highly challenging and requires optimizing the code in order to fit the memory constraints, which, for the case of the M5Stack, is less than 400 kB. Traditional approaches involve quantization techniques [11], which aim to reduce the size of an ML trained out of the

[1] https://docs.m5stack.com/en/core/gray.

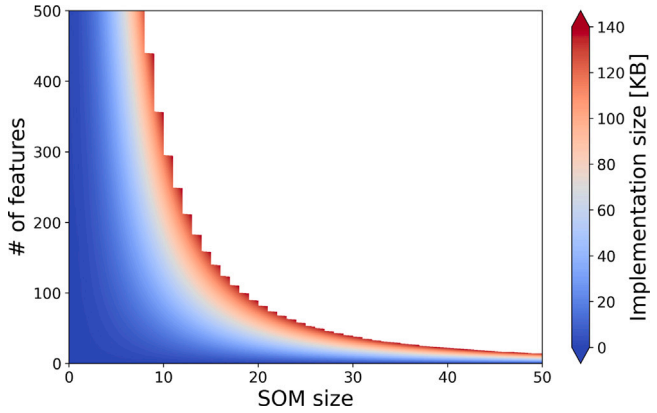


Fig. 3. Impact of the SOM size and the number of features on the deployed firmware. The color indicates how close the memory size of the system is to the limits imposed by the microcontroller. Areas not reached by a bar indicate that the system is not deployable on the device.

microcontroller by reducing the precision of floating point coefficients [12]. To this aim, frameworks like TensorFlow Lite [36] support the quantization process and the exportation of the model in a binary format; the latter is loaded by an interpreter running on the microcontroller. However, we were unable to follow such an approach as the generated code still exceeded the memory capacity of the M5Stack device.

To overcome these challenges, we opted to implement a SOM library from scratch in *C++* for microcontrollers, taking into account the hardware characteristics of the latter. The SOM library for the Arduino SDK is available online[2] and can be easily customized to support different microprocessors than the ESP-32 one.

Through our implementation, the size of the final deployed SOM model is restricted to several tens of kilobytes. The M5Stack has 320 kB of RAM memory, and after an initial code optimization phase that encompasses sensor reading and wireless communication, the maximum memory available for the SOM model is 140 kB only. As described in Section 4.1, the SOM size depends on two variables: the number of features and the map size. In Fig. 3 we depict the relationship between these two variables and the final size of the SOM model to be installed in the wearable device. In the Figure, only feasible configurations are shown. In Section 7.4 we investigate the trade-off between the number of input features, the map size, and the overall accuracy of the HAR task.

7. Performance evaluation

In this Section, we evaluate the performance of the proposed HAR system for resource-constrained IoT devices from different points of view. The evaluation is organized into five phases: (*i*) defining the HAR datasets, (*ii*) analyzing the classic supervised learning techniques on the HAR datasets to establish a reference baseline in a standalone context, (*iii*) assessing the effectiveness of the ANOVA-F technique for feature reduction, (*iv*) analyzing the standalone performance of the SOM on the two datasets, to evaluate the accuracy and memory size occupation for different map sizes, and (v) analyzing the collective approach in both offline and onboarding contexts.

7.1. Description of the datasets

We evaluated our proposed HAR system on two different datasets, one from the literature and a custom one, obtained by gathering data from a few volunteers through the *M5Stack* device.

Regarding the first, we used the well-known *University of California Irvine (UCI) HAR* dataset [21]. It was collected using the accelerometer and gyroscope sensors of a smartphone carried by 30 volunteers performing six different activities: *walking*, *walking upstairs*, *walking downstairs*, *sitting*, *standing*, and *lying down*. The dataset includes time-series data for the three axes of each sensor, as well as the corresponding activity label. The dataset has been widely used in research for developing and evaluating HAR models based on ML techniques. More in detail, each record in the dataset is provided with a vector composed of 561 features, calculated in the time and frequency domain. The features were sourced from the accelerometer and gyroscope 3-axial raw signals, which were sampled at a constant rate of 50Hz. To eliminate noise, a median filter and a 3rd-order low pass *Butterworth filter* with a corner frequency of 20Hz were employed. Another low-pass Butterworth filter with a corner frequency of 0.3Hz was used to split the acceleration signal into body and gravity acceleration signals. After that, the body's linear acceleration and angular velocity were used to derive *Jerk signals* in time. The magnitude of these three-dimensional signals was determined using the Euclidean norm. Finally, a Fast Fourier Transform (FFT) was applied to some of these signals.

[2] https://github.com/UniBO-PRISMLab/extreme-edge-som.

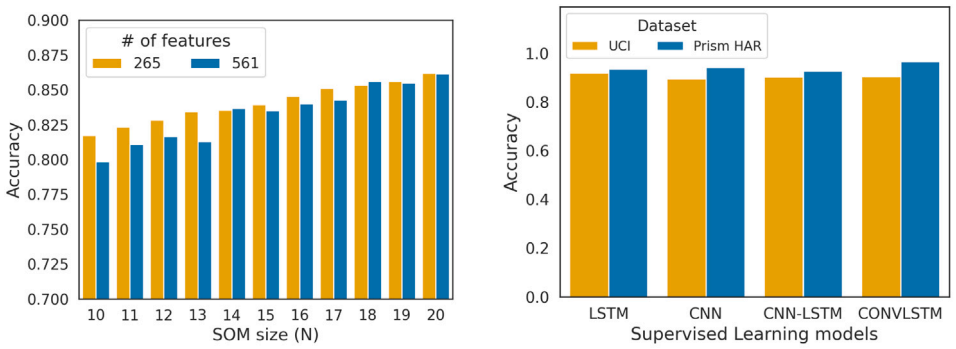


Fig. 4. 4(a) compares the accuracy of SOMs of different sizes for the UCI HAR dataset, when using all 561 features or a subset of 265 features only, excluding the frequency domain. 4(b) shows a comparison between different supervised techniques over the two datasets.

However, not all of these features are equally essential for accurate HAR systems. Some may be redundant or noisy, while others may be more informative. In [1], we assessed the performance of SOMs of various sizes over the UCI HAR dataset using different feature subsets, which suggests that the frequency-based features did not significantly improve the performance of our HAR system, at least not for the prediction models taken into account in this paper. An accurate result is shown in Fig. 4(a), which demonstrates that the prediction accuracy using the entire set of 561 features and the sole subset of 265 features, obtained by excluding frequency-domain features, behave quite similarly. Quite counterintuitively, results show that a smaller set of features tends to perform slightly better, which suggests that the full set of features is likely oversized for a simple model like our SOM. As a result, we decided to remove the frequency-based features and focus solely on time-domain and statistical features in our deployment. This results in a dataset with 10,299 rows with 265 features each.

The second dataset was built by the authors, aiming to resemble similar conditions to the ones observed in the UCI HAR dataset, in order to perform a meaningful comparison. We attached the M5Stack wearable device on the ankle of 8 different volunteers, who then performed the same six activities present in the UCI HAR dataset. Data was sampled with a rate of 41.29Hz and underwent the same procedures used on the UCI dataset, such as Butterworth filters and Jerk calculation. We then calculated the same selected 265 features, skipping the calculation of the frequency-based ones, obtaining a dataset with 16,976 rows and 265 features each. In our case, the dataset is intentionally unbalanced, with 10,916 rows associated with the most contributory subject, and 560 rows with the least contributory one. As the dataset is collected within the IoT-Prism Lab at the University of Bologna, we named the dataset **Prism HAR** and made it publicly accessible.[3]

7.2. Supervised learning for HAR

In this Section, we show the performance of state-of-the-art supervised learning methods for HAR applications. These techniques rely on labeled training data to build a model that can predict the activity performed by the user, while unsupervised learning techniques aim to discover patterns or structures in the data without any labeled information. Due to the constrained EE device used in our HAR system, in the following Sections, we will deeply focus on evaluating the performance of unsupervised learning techniques instead. This is because unsupervised learning techniques, such as SOMs, are lightweight and computationally efficient, making them well-suited for deployment on constrained devices. However, to have a meaningful comparison perspective, we analyze different supervised learning techniques for HAR applications as a baseline. Specifically, we evaluated the following techniques: long short-term memory (LSTM), convolutional neural network (CNN), CNN–LSTM, and convolutional LSTM (CONVLSTM).

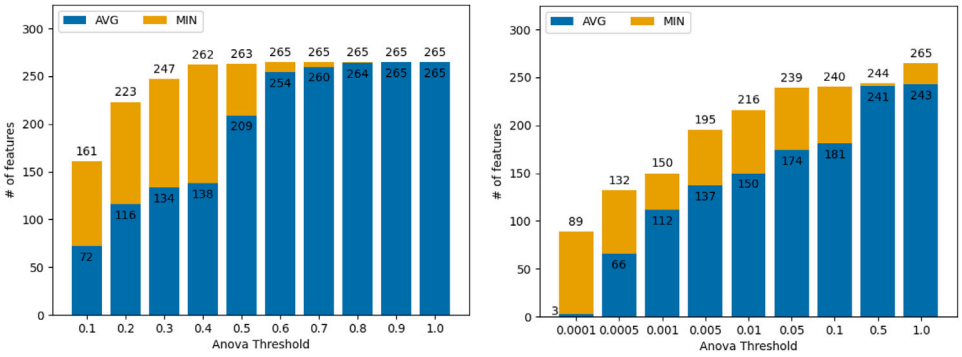
Fig. 4(b) shows that supervised learning techniques are very effective in classifying human activities, with all four models achieving high accuracy. The Figure depicts the results for both datasets. Here, the CNN–LSTM model achieved the highest overall performance, with an accuracy of 99% on our Prism HAR dataset. This evaluation shows also the effectiveness of placing the sensors in the right position, *i.e.*, the ankle, instead of using the sensors inside the smartphone. The results of the supervised learning techniques over the UCI HAR dataset are around 8% less accurate than the one executed on our dataset.

It is however necessary to remark that, while supervised learning methods are highly effective in the classification of HAR systems, they are not suitable for deployment in EE devices. Our M5Stack device, in fact, has 140 kB available memory only. On the contrary, the supervised learning models require significant computational resources and storage capacity.

7.3. Features reduction with ANOVA-F

In this Section, we present the results of the ANOVA-F feature reduction method on both the UCI HAR dataset and our custom dataset. We analyzed the two different variations of the ANOVA-F method described in Section 4.2: one using the average as the

[3] https://github.com/UniBO-PRISMLab/Prism-HAR-dataset.



(a) UCI HAR Dataset (b) Prism HAR Dataset

Fig. 5. Feature reduction using *AVG* and *MIN* for different value of An_{thr}.

aggregation function and another using the minimum as the aggregation function, in the following called *AVG* and *MIN*, respectively. The aim of this analysis was to identify the most relevant features for our ML model and reduce their number to minimize storage usage, as requested in previous Section 6. Our results show that both variations of the ANOVA-F method were effective in reducing the number of features used in our classification model. Figs. 5(a) and 5(b) display the outcomes of features reduction obtained by applying distinct An_{thr} on both the UCI HAR and Prism HAR datasets. It should be noted that the scales on the x-axes are different in the Figures (UCI dataset in Fig. 5(a) and Prism dataset in Fig. 5(b)). This disparity in scaling is because in the Prism dataset, setting the An_{thr} to 0.5 resulted in almost all of the input features being included without any feature reduction. Therefore, to evaluate the Prism dataset, we used exponential steps for the An_{thr}.

The results show that the variation using *AVG* as the aggregation function resulted in a greater reduction in the number of features compared to the *MIN* version, especially for low levels of the An_{thr}. Based on these results, we selected the *AVG* variation of the ANOVA-F method for the next evaluations in order to achieve a significant reduction in the number of features in input to our model.

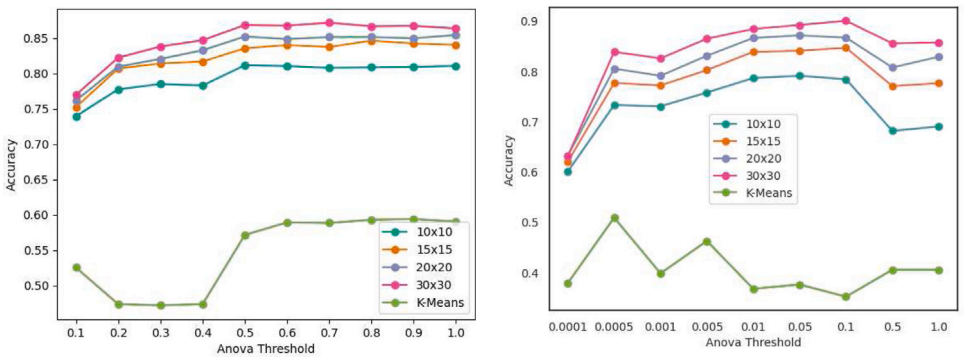
7.4. SOM for extreme edge devices

In this Section, we present the evaluation process undertaken to determine the optimal configuration for the SOM and the ANOVA-F threshold (An_{thr}) in our HAR model. Our primary goal was to achieve a satisfactory classification performance while maintaining a compact model size that is suitable for deployment on the M5Stack device. To ensure the robustness and generalizability of our model, we analyze the performance of SOMs with the legacy *K-Means* algorithm for comparison purposes. We conducted a series of experiments to assess the performance of our HAR model by varying the size of the SOM and the An_{thr}. The SOM sizes tested ranged from small (e.g., 10×10) to larger maps (e.g., 30×30), while the An_{thr} values were varied between 0.0001 and 1 for the Prism dataset and between 0.1 and 1 for the UCI dataset. This range of values was chosen to cover a wide spectrum of model complexities, thereby providing a comprehensive understanding of the trade-off between model size and performance.

Upon analyzing the results in Figs. 6(a) and 6(b) from both the UCI HAR dataset and the Prism HAR dataset, we observed that the HAR model's performance improved by increasing the SOM size. However, for sizes greater than 15×15, the increase in accuracy is not so evident. Regarding An_{thr}, we notice a decrease in performance for low values in both datasets and for high values only in the Prism dataset. Here, we can notice that the optimal performance is different for the two datasets. The UCI dataset performs well starting from An_{thr} of 0.5, i.e., from about 209 features (see Fig. 5(a)). The Prism dataset performs well with An_{thr} values between 0.0005 and 0.1, i.e., with a value between 66 and 181 features. It is clear that with larger SOMs and higher An_{thr}, and hence more features, the model would also demand more computational resources and storage, which may not be available on the M5Stack device.

Taking these considerations into account, we determined that the optimal configuration for our HAR model, when deployed locally, is a SOM size of 15×15 and An_{thr} of 0.005. For the deployment, we considered the results from our custom dataset. This configuration demonstrated a satisfactory balance between model accuracy and computational complexity, making it well-suited for deployment on our wearable device which has a limit of 140 kB.

To further evaluate the practical feasibility of deploying our HAR models on resource-constrained devices, we conducted energy consumption tests comparing the power usage of the SOM algorithm (with a 10×10 size) with the K-Means algorithm. This analysis is crucial as wearable devices have constrained battery capacities and additional energy consumption can impact their lifetime. The results depicted in Fig. 7 indicate that the SOM method consumes approximately 0.035mJ more per classification operation compared to the K-Means algorithm. Although there is a minor increase in energy consumption, this trade-off is justified by the significant improvement in classification accuracy provided by the SOM-based model.



(a) UCI HAR Dataset (b) Prism HAR Dataset

Fig. 6. The accuracy evaluated with *K-Means*, different SOM sizes by varying An_{thr} for the UCI dataset is shown in Fig. 6(a). The accuracy evaluated with *K-Means*, different SOM sizes by varying An_{thr} for our custom dataset is shown in Fig. 6(b).

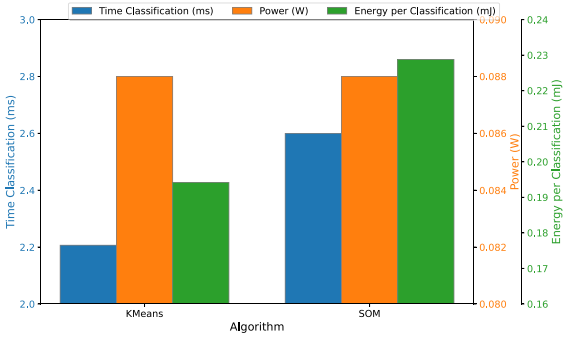
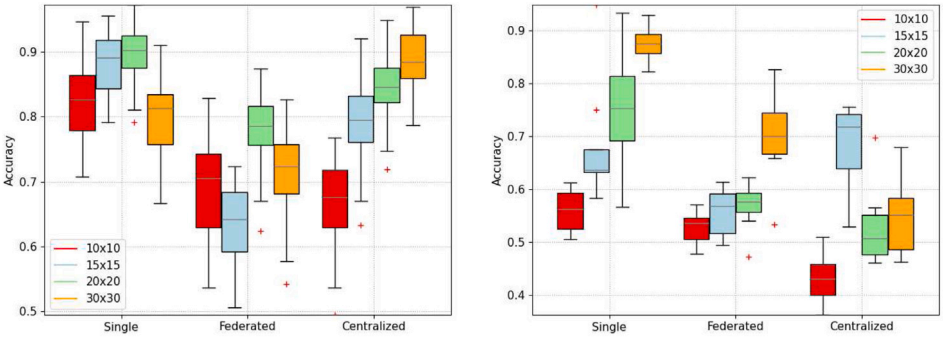


Fig. 7. Energy consumption comparison between K-Means and SOM per classification operation.



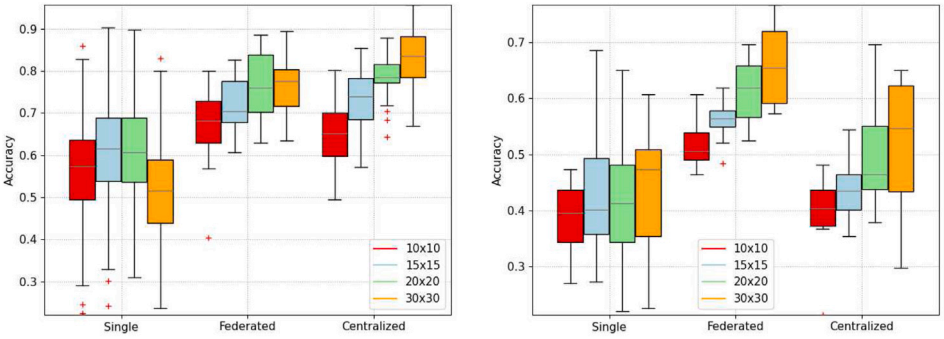
(a) UCI HAR Dataset (b) Prism HAR Dataset

Fig. 8. Evaluation of the collective offline approach.

7.5. Evaluating our SOM on the collective context

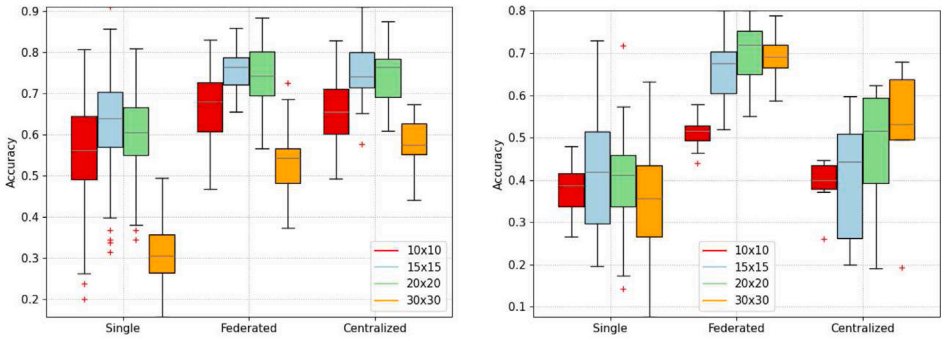
In this last stage of our performance evaluation, we aim to show how our HAR proposed system behaves in a real scenario, where it is evaluated on single users. In particular, the outcome of this performance evaluation demonstrates whether the results outlined in the previous section, which encompass an ideal case, would be impacted by a real scenario.

First of all, we evaluated our system under the three different configurations outlined in Section 5: Single, Centralized, and Federated considering an *offline approach*, thus assuming that every user participated in the training process and no user joins at a later stage. This is an assumption that serves as a baseline for the collective context, as we do not deem such a setup as realistic. The results of the evaluation are reported in Fig. 8 for both datasets and for SOM sizes of 10×10, 15×15, 20×20, and 30×30,



(a) UCI HAR Dataset (b) Prism HAR Dataset

Fig. 9. Evaluation of the collective onboarding approach.



(a) UCI HAR Dataset (b) Prism HAR Dataset

Fig. 10. Evaluation of the collective onboarding approach with the maximum number of features fitting the EE device.

compatibly with our standalone experiments. In this test, we used all 265 features, which means that the models are not necessarily deployable on an EE device. We will remove this assumption later on, here we wanted to focus the evaluation on the different collective configurations, without introducing further variables. Even though in this case the experiment shows slightly different trends on the two datasets, we still can state that the Single configuration yields the best results for almost all SOM sizes. This justifies our earlier statement, that HAR highly depends on the single individual, while a generalized model tends to have worse performance, as people have different movement patterns when they perform physical activities. Furthermore, we notice that the Centralized configuration performs better than the Federated one for the UCI HAR dataset, while it happens almost the opposite for the Prism HAR dataset across all experiments. This happens because the Prism HAR dataset is highly unbalanced, with ~65% of it attributed to a single user. This implies that the Centralized configuration will most likely train the model accounting greatly for that single user, while the others will have a negligible impact. The testing phase notably outputs the unweighted average accuracy across the users, resulting in the majority of them using a model that is mostly associated with someone else who will likely walk, sit, or run differently from them. Instead, the Federated configuration gives the same importance to all users regardless of the size of their contribution, showing that, even in the presence of fewer data points for the training process, this has a positive influence on the results.

Next, we evaluated the *onboarding approach* by assuming that one of the users is onboarding the system and cannot perform any training. We simulated such a scenario by performing a Leave-One-Out Cross Validation (LOOCV), which results in one user being removed from the training phase. Then, once the training phase is over, whatever the configuration, the user onboards and adopts a pre-trained model as explained in Section 5. In all cases, we repeated the LOOCV by leaving out each of the users in the dataset and then performing the onboarding. For the Single case, because the model adopted by the onboarding user is randomly chosen among all the others, we further repeated the experiment for each of the other users. Fig. 9 shows the results, which display an expected behavior: the Single configuration now has the worst performance because the onboarding user uses a model that is specifically trained on another user, that is, not generalized. Furthermore, as expected, the performance of both Federated and Centralized configurations is quite similar to the offline case, suggesting that both of them are minimally affected by a new user onboarding the system. It is also noticeable how, for smaller SOMs, the performance of the Federated approach increases with respect to its centralized counterpart, which leads to our next result.

Our last experiment repeats the LOOCV on both datasets, this time reducing the number of features from 265 to the maximum number that is acceptable for a SOM to be deployed into the EE system, always selecting the ones with the smallest ANOVA-f value. In particular, the number of considered features is as follows:

- SOM size: 10 - max features: 356 (we keep all 265 features here)
- SOM size: 15 - max features: 158
- SOM size: 20 - max features: 89
- SOM size: 30 - max features: 39

Results, shown in Fig. 10, have a similar trend to the previous test, with the difference that here the Federated configuration performs in line with the Centralized one for the UCI HAR dataset, while still performing better in the Prism HAR dataset. In fact, decreasing the number of features has a much more negative impact on the Centralized configuration, while the Federated one seems more resilient — this is more evident for the bigger models because the number of features drops more significantly. This suggests that, in a realistic scenario, when a new user onboards the systems and is given an off-the-shelf model, then a Federated configuration yields the best results, provided the model is sized appropriately to fit a constrained architecture.

8. Conclusions

In this paper, we presented the design, implementation, and validation of an Extreme Edge (EE)-aware Human Activity Recognition (HAR) system. The system aims to classify human activity on constrained and wearable IoT devices equipped with inertial sensors, by considering the unique challenges posed by the computational environment, such as the scarcity of hardware resources and the impracticability of a data labeling phase for some individuals. Our HAR system incorporates a feature selection mechanism, based on the ANOVA-F technique, to reduce the dimensionality of HAR features at the input stage. Then, it utilizes an unsupervised classification technique based on Self-Organizing Maps (SOMs) to enable data separation and effective activity classification. We developed a C++ SOM library for the Arduino SDK and validated our system through an M5Stack IoT wearable board. The experimental results take place in two phases and use two HAR datasets: the UCI HAR dataset and the Prism HAR dataset built through our IoT prototype. First, our standalone experiments demonstrated that the proposed SOM solution is capable of outperforming other unsupervised approaches and achieves performance close to state-of-the-art DL techniques while generating a model small enough to fit the limited memory capabilities of EE devices. Second, our collective experiments show the advantages of a Federated Learning (FL) approach when considering users that onboard the system without the capacity to train their model, while still considering privacy implications. From our experiments it is also evident that HAR systems are highly personalized, that is, if an individual can train his/her own HAR model, then this will likely perform better than any collective one. This implies that any user who can participate in the model training will not take advantage of the collective model, as he/she would be better off using the one trained locally without any external influence. However, because this assumes that the training is conducted properly, which is not a trivial task, then inexperienced users who are incapable of conducting extensive training would be better off taking advantage of the collective model, especially if they onboard the system at a later stage. In this latter case, we have shown that a centralized approach suffers from unbalanced datasets, while an unweighted FL solution makes sure that all contributing users have the same impact regardless of the size of their dataset and it is in general a better choice.

In the future, there might be a focus on enhancing the system through exploring different methods for selecting features and adapting our approach to suit various wearable devices and application areas.

CRediT authorship contribution statement

Angelo Trotta: Writing – original draft, Visualization, Validation, Project administration, Data curation. **Federico Montori:** Writing – original draft, Methodology, Formal analysis, Conceptualization. **Leonardo Ciabattini:** Writing – review & editing, Resources, Investigation. **Giulio Billi:** Visualization, Software, Data curation. **Luciano Bononi:** Supervision. **Marco Di Felice:** Writing – review & editing, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

We have shared the link in the text of the paper.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used ChatGPT in order to improve language and readability. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

Acknowledgments

The work has been supported by the PRIN PNRR 2022 project SORTT (“An IoT-Serviced and Ontology-based Remote human digital Twin for physioTherapy: a feasibility study”, project id: P2022H74YP) and by the PRIN 2022 project I-TROPHYTS (“IoT and humanoid RObotics for autonomic PHYsio-Therapeutic monitoring, coaching and supervising in smart Spaces: a feasibility study”, project id: 20224TAETP).

References

[1] A. Trotta, F. Montori, G. Vallasciani, L. Bononi, M. Di Felice, Optimizing IoT-based human activity recognition on extreme edge devices, in: 2023 IEEE International Conference on Smart Computing, SMARTCOMP, IEEE, 2023, pp. 41–48, http://dx.doi.org/10.1109/SMARTCOMP58114.2023.00023.

[2] Y. Chen, L. Yu, K. Ota, M. Dong, Robust activity recognition for aging society, IEEE J. Biomed. Health Inf. 22 (6) (2018) 1754–1764, http://dx.doi.org/10.1109/JBHI.2018.2819182.

[3] M.A. Khatun, M.A. Yousuf, S. Ahmed, M.Z. Uddin, S.A. Alyami, S. Al-Ashhab, H.F. Akhdar, A. Khan, A. Azad, M.A. Moni, Deep CNN-LSTM with self-attention model for human activity recognition using wearable sensor, IEEE J. Transl. Eng. Health Med. 10 (2022) 1–16, http://dx.doi.org/10.1109/JTEHM.2022.3177710.

[4] Y.-L. Hsu, S.-C. Yang, H.-C. Chang, H.-C. Lai, Human daily and sport activity recognition using a wearable inertial sensor network, IEEE Access 6 (2018) 31715–31728, http://dx.doi.org/10.1109/ACCESS.2018.2839766.

[5] D. Bouchabou, S.M. Nguyen, C. Lohr, B. LeDuc, I. Kanellos, A survey of human activity recognition in smart homes based on IoT sensors algorithms: Taxonomies, challenges, and opportunities with deep learning, Sensors 21 (18) (2021) URL https://www.mdpi.com/1424-8220/21/18/6037.

[6] R. Salpietro, L. Bedogni, M. Di Felice, L. Bononi, Park here! a smart parking system based on smartphones’ embedded sensors and short range communication technologies, in: 2015 IEEE 2nd World Forum on Internet of Things, WF-IoT, 2015, pp. 18–23, http://dx.doi.org/10.1109/WF-IoT.2015.7389020.

[7] A.R.M. Forkan, F. Montori, D. Georgakopoulos, P.P. Jayaraman, A. Yavari, A. Morshed, An industrial IoT solution for evaluating workers’ performance via activity recognition, in: 2019 IEEE 39th International Conference on Distributed Computing Systems, ICDCS, IEEE, 2019, pp. 1393–1403.

[8] G. De Leonardis, S. Rosati, G. Balestra, V. Agostini, E. Panero, L. Gastaldi, M. Knaflitz, Human activity recognition by wearable sensors: Comparison of different classifiers for real-time applications, in: 2018 IEEE International Symposium on Medical Measurements and Applications, MeMeA, IEEE, 2018, pp. 1–6.

[9] G. Alam, I. McChesney, P. Nicholl, J. Rafferty, Open datasets in human activity recognition research—Issues and challenges: A review, IEEE Sens. J. 23 (22) (2023) 26952–26980, http://dx.doi.org/10.1109/JSEN.2023.3317645.

[10] J. Dai, B. Saghafi, J. Wu, J. Konrad, P. Ishwar, Towards privacy-preserving recognition of human activities, in: 2015 IEEE International Conference on Image Processing, ICIP, IEEE, 2015, pp. 4238–4242.

[11] D. Xu, T. Li, Y. Li, X. Su, S. Tarkoma, T. Jiang, J. Crowcroft, P. Hui, Edge intelligence: Architectures, challenges, and applications, 2020, http://dx.doi.org/10.48550/ARXIV.2003.12172, URL https://arxiv.org/abs/2003.12172.

[12] A. Ghibellini, L. Bononi, M. Di Felice, Intelligence at the IoT edge: Activity recognition with low-power microcontrollers and convolutional neural networks, in: 2022 IEEE 19th Annual Consumer Communications & Networking Conference, CCNC, 2022, pp. 707–710, http://dx.doi.org/10.1109/CCNC49033.2022.9700665.

[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] T. Kohonen, The self-organizing map, Proc. IEEE 78 (9) (1990) 1464–1480, http://dx.doi.org/10.1109/5.58325.

[15] 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 (ISPA/IUCC/BDCloud/SocialCom/SustainCom), IEEE, 2018, pp. 1103–1111.

[16] 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.

[17] P.-E. Novac, A. Castagnetti, A. Russo, B. Miramond, A. Pegatoquet, F. Verdier, Toward unsupervised human activity recognition on microcontroller units, in: 2020 23rd Euromicro Conference on Digital System Design, DSD, IEEE, 2020, pp. 542–550.

[18] E. Kim, A. Helal, D. Cook, Human activity recognition and pattern discovery, IEEE Pervasive Comput. 9 (2010) http://dx.doi.org/10.1109/MPRV.2010.7.

[19] D. Riboni, F. Murru, Unsupervised recognition of multi-resident activities in smart-homes, IEEE Access 8 (2020) http://dx.doi.org/10.1109/ACCESS.2020.3036226.

[20] F. Demrozi, G. Pravadelli, A. Bihorac, P. Rashidi, Human activity recognition using inertial, physiological and environmental sensors: A comprehensive survey, IEEE Access 8 (2020) 210816–210836, http://dx.doi.org/10.1109/ACCESS.2020.3037715.

[21] D. Anguita, A. Ghio, L. Oneto, X. Parra, J.L. Reyes-Ortiz, A public domain dataset for human activity recognition using smartphones, Comput. Intell. Mach. Learn. ESANN (2013).

[22] S. Zhang, Y. Li, S. Zhang, F. Shahabi, S. Xia, Y. Deng, N. Alshurafa, Deep learning in human activity recognition with wearable sensors: A review on advances, Sensors 22 (4) (2022) http://dx.doi.org/10.3390/s22041476, URL https://www.mdpi.com/1424-8220/22/4/1476.

[23] L. Chen, Y. Zhang, S. Miao, S. Zhu, R. Hu, L. Peng, M. Lv, SALIENCE: An unsupervised user adaptation model for multiple wearable sensors based human activity recognition, IEEE Trans. Mob. Comput. (2022) 1, http://dx.doi.org/10.1109/TMC.2022.3171312.

[24] H. Amrani, D. Micucci, P. Napoletano, Unsupervised deep learning-based clustering for human activity recognition, in: 2022 IEEE 12th International Conference on Consumer Electronics, ICCE-Berlin, 2022, pp. 1–6, http://dx.doi.org/10.1109/ICCE-Berlin56473.2022.9937141.

[25] P. Ariza Colpas, E. Vicario, E. De-La-Hoz-Franco, M. Pineres-Melo, A. Oviedo-Carrascal, F. Patara, Unsupervised human activity recognition using the clustering approach: A review, Sensors 20 (9) (2020) http://dx.doi.org/10.3390/s20092702.

[26] A.R. Sanabria, F. Zambonelli, J. Ye, Unsupervised domain adaptation in activity recognition: A GAN-based approach, IEEE Access 9 (2021) 19421–19438, http://dx.doi.org/10.1109/ACCESS.2021.3053704.

[27] R. Presotto, G. Civitarese, C. Bettini, Semi-supervised and personalized federated activity recognition based on active learning and label propagation, Pers. Ubiquitous Comput. 26 (5) (2022) 1281–1298, http://dx.doi.org/10.1007/s00779-022-01688-8.

[28] R. Presotto, G. Civitarese, C. Bettini, Federated clustering and semi-supervised learning: A new partnership for personalized human activity recognition, Pervasive Mob. Comput. 88 (2023) 101726, http://dx.doi.org/10.1016/j.pmcj.2022.101726, URL https://www.sciencedirect.com/science/article/pii/S1574119222001390.

[29] P. Prakash, J. Ding, R. Chen, X. Qin, M. Shu, Q. Cui, Y. Guo, M. Pan, IoT device friendly and communication-efficient federated learning via joint model pruning and quantization, IEEE Internet Things J. 9 (2022) 13638–13650, http://dx.doi.org/10.1109/jiot.2022.3145865.

[30] F. Luo, S. Khan, Y. Huang, K. Wu, Binarized neural network for edge intelligence of sensor-based human activity recognition, IEEE Trans. Mob. Comput. 22 (3) (2023) 1356–1368, http://dx.doi.org/10.1109/TMC.2021.3109940.

[31] A. Moschetti, L. Fiorini, D. Esposito, P. Dario, F. Cavallo, Daily activity recognition with inertial ring and bracelet: An unsupervised approach, in: 2017 IEEE International Conference on Robotics and Automation, ICRA, 2017, pp. 3250–3255, http://dx.doi.org/10.1109/ICRA.2017.7989370.
[32] J. Han, M. Kamber, J. Pei, Data Mining Concepts and Techniques, Third Edition, Morgan Kaufmann Publishers, Waltham, Mass., 2012.
[33] X. Li, J. Lin, Linear time complexity time series classification with bag-of-pattern-features, in: 2017 IEEE International Conference on Data Mining, ICDM, IEEE, 2017, pp. 277–286.
[34] 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.
[35] A.R. Anwary, H. Yu, M. Vassallo, Optimal foot location for placing wearable IMU sensors and automatic feature extraction for gait analysis, IEEE Sens. J. 18 (6) (2018) 2555–2567, http://dx.doi.org/10.1109/JSEN.2017.2786587.
[36] R. David, J. Duke, A. Jain, V. Janapa Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, T. Wang, et al., Tensorflow lite micro: Embedded machine learning for tinyml systems, Proc. Mach. Learn. Syst. 3 (2021) 800–811.