

Received October 9, 2020, accepted October 31, 2020, date of publication November 5, 2020, date of current version November 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3036226

Unsupervised Recognition of Multi-Resident Activities in Smart-Homes

DANIELE RIBONI^{ID}, (Member, IEEE), AND FLAVIA MURRU^{ID}

Department of Mathematics and Computer Science, University of Cagliari, 09124 Cagliari, Italy

Corresponding author: Daniele Riboni (riboni@unica.it)

This work was supported in part by the Project ADAM under Grant Convenzione tra Fondazione di Sardegna e Atenei Sardi annualità 2018 CUP F74I19000900007, and in part by the POR FESR Sardegna 2014-2020 Project Match Information System and Technologies for the Evaluation of the Performance (MISTER).

ABSTRACT Several methods have been proposed in the last two decades to recognize human activities based on sensor data acquired in smart-homes. While most existing methods assume the presence of a single inhabitant, a few techniques tackle the challenging issue of multi-resident activity recognition. To the best of our knowledge, all existing methods for multi-inhabitant activity recognition require the acquisition of a labeled training set of activities and sensor events. Unfortunately, activity labeling is costly and may disrupt the users' privacy. In this article, we introduce a novel technique to recognize multi-inhabitant activities without the need of labeled datasets. Our technique relies on an unlabeled sensor data stream acquired from a single resident, and on ontological reasoning to extract probabilistic associations among sensor events and activities. Extensive experiments with a large dataset of multi-inhabitant activities show that our technique achieves an average accuracy very close to the one of state-of-the-art supervised methods, without requiring the acquisition of labeled data.

INDEX TERMS Activity recognition, multi-resident activities, hybrid reasoning, unsupervised reasoning.

I. INTRODUCTION

The ability to recognize the activities going on in smart-homes is a central requirement in several application domains, including healthcare and home automation [1]. As a consequence, a plethora of activity recognition methods have been proposed in the last decades to recognize activities based on sensor data and artificial intelligence algorithms [2]. However, most activity recognition methods assume the presence of a single person in the home, while it is a common situation for people to live together and to concurrently execute activities. Unfortunately, techniques for single-resident activity recognition are ineffective when multiple persons concurrently execute activities in the smart space. Indeed, multiple streams of sensor events generated by the execution of different activities by different persons are treated as a whole by the single-resident activity recognition system. In general, the resulting single stream of sensor events does not match any activity model, thus confusing the recognition system. This fact limits the applicability of single-resident methods to restricted scenarios or to specific user categories.

The associate editor coordinating the review of this manuscript and approving it for publication was M. Anwar Hossain^{ID}.

A few previous works have tackled the challenging issue of recognizing multi-resident activities [3]. Existing multi-resident techniques generally apply two steps-recognition. In the first step, "data association" is applied to identify the resident who triggered each sensor event. In the second step, the result of data association is used to recognize the activities of each resident. Activity recognition is generally achieved by probabilistic models, such as Hidden Markov Models [4] and Conditional Random Fields [5], or by other machine learning algorithms.

To the best of our knowledge, all existing multi-resident activity recognition systems are based on supervised learning: they rely on a labeled training set of sensor data and activities. Unfortunately, activity labeling by an external observer may undermine the inhabitants' privacy. Moreover, the acquisition of such datasets incurs high overhead. In the literature, it is reported that labeling one hour of single-user activities may require from 30 minutes to 10 hours, depending on the data acquisition modality, and on the level of detail of annotations [6], [7]. Of course, labeling is even harder when multiple activities are executed concurrently by multiple persons in a smart environment.

In this article, we propose a novel method to recognize multi-resident activities without the need of labeled

datasets. Our method relies on a weaker form of data association, which we name “resident separation”, that consists in determining whether a pair of observed sensor events was generated by the same resident or by different residents. We propose an unsupervised data mining method to perform resident separation based on a stream of sensor events acquired in the smart-home during the execution of everyday activities by a single resident. Then, we apply knowledge-based reasoning to recognize the multi-resident activities going on in the smart-home considering the mined resident separation model and the stream of observed sensor events. Unsupervised activity recognition is achieved exploiting semantic correlations among sensor events and activity classes, which are extracted by reasoning with an OWL 2 ontology [8].

We have implemented our algorithms, and experimented our system with a well-known dataset of multi-resident activities performed by 26 couples of individuals in a smart-home instrumented with several kinds of sensors. The results show that our unsupervised method for resident separation is accurate, correctly classifying 87.9% pairs of sensor events as either being generated by the same resident or by different residents. Moreover, our multi-resident activity recognition algorithm achieves an accuracy close to the one obtained by a state-of-the-art supervised method based on Hidden Markov Models, without requiring any data labeling.

The rest of the paper is structured as follows. Section II discusses related work. Section III introduces our multi-inhabitant activity recognition system. Algorithms for resident separation are illustrated in Section IV, while Section V explains the algorithms for activity recognition. Section VI reports experimental results, while Section VII concludes the paper and illustrates directions for future work.

II. RELATED WORK

Techniques for multi-resident activity recognition are generally composed by two parts: *data association*, and *activity recognition*. The goal of the former is to associate each sensor event to the resident who triggered it. The latter aims at recognizing the activity of each resident based on the result of data association. Some studies, including [9]–[12], focus on the problem of data association, while others consider this problem already solved, and directly focus on the recognition of activities [13]–[16].

Activity recognition systems may be classified in two categories: data-driven or knowledge-based ones [2]. To the best of our knowledge, all existing methods for recognizing multi-resident activities adopt a data-driven approach based on supervised learning [3]. On the contrary, in our work, we propose a hybrid technique combining data-driven *resident separation* (i.e., a weaker form of data association) and knowledge-based activity recognition. A strong point of our work is that it does not rely on labelled activity datasets, whose acquisition is costly and may negatively impact the resident privacy. Furthermore, we do not need any technique or device to perform data association, because we apply

resident separation based on streams of unlabeled sensor events.

Different techniques and tools have been proposed to perform accurate data association using cameras or wearable devices [17], [18]; however, those tools are often perceived as obtrusive by several people [19]. Hence, in our work, we aim at performing multi-resident activity recognition without the use of cameras or specific wearables.

Crandall *et al.* propose to use Hidden Markov Models (HMM) for resident identification [9]. In their model, the hidden states represent the possible residents, while observations represent the sensor events emitted as a consequence of the residents’ activities. Their supervised method achieved around 90% accuracy on different datasets. However, their method relies on a dataset of sensor events labeled with the resident that triggered the event, while our method does not require any labeled training set. Cook *et al.* propose the use of event density maps and Bayesian models to automatically track resident movements and sensors activations in sensor-dense environments such as smart-homes and smart-workplaces [10]. The result of resident tracking is later used to recognize resident interactions through supervised machine learning using HMM. The use of Conditional Random Fields was proposed by Hsu *et al.* for both data association and multi-resident activity recognition [11]. Chen and Tong propose the use of combined labels to represent the activity performed simultaneously and independently by multiple residents [12], and adopt supervised probabilistic models for activity recognition.

Singla *et al.* propose the use of HMM for multi-resident activity recognition assuming exact knowledge of data association [13]. With exact data association, a HMM is built for each resident, achieving 73.15% accuracy on activity recognition on the CASAS multi-resident dataset, which is the same dataset that we use in this work. Without data association, a single HMM is built for the two residents, and accuracy drops to 60.6%. Their results indicate that the ability to associate sensor events to different inhabitants is of foremost importance for multi-resident activity recognition. Exact knowledge of data association is also assumed by Ul Alam *et al.*, which mine association rules from labeled data to reduce the complexity of a Hierarchical Dynamic Bayesian Network to capture correlations among sensor events and activities [14]. Other researchers assumed exact knowledge of data association, and applied different supervised learning algorithms, including Recurrent Neural Networks used by Tran *et al.* in [15], and Incremental Decision Trees used by Prosegger and Bouchachia in [16].

Most existing techniques for multi-resident activity recognition do not explicitly consider the interaction among inhabitants. However, modeling interaction may increase recognition rates when the residents perform collaborative activities. For this reason, Chiang *et al.* introduce a binary “interaction feature” attribute stating whether two residents are in the same room [20]. Gu *et al.* use Emerging Patterns distinguishing individual and collaborative activities [21].

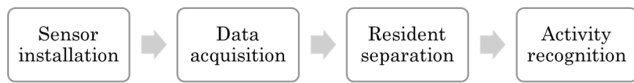


FIGURE 1. High-level flow chart of our multi-resident activity recognition framework.

However, all these methods require the acquisition of labelled training sets, and accurate tools or devices to associate each sensor event to the resident that triggered it. On the contrary, our technique does not require labeled data, and relies on the existing smart-home sensor infrastructure, without the need to wear or install additional hardware for data separation.

III. MULTI-INHABITANT ACTIVITY RECOGNITION SYSTEM

Before explaining our algorithm for resident separation, we provide a formal description of the problem. We denote by **event type** the pair:

$$event_type = \langle sensor, value \rangle \quad (1)$$

which indicates a possible value of a given sensor. For instance, the event type $\langle fridge_door_sensor, open \rangle$ indicates that the door of the fridge has been opened.

An actual occurrence of an event type is given by a **sensor event**, that is a tuple:

$$sensor_event = \langle timestamp, event_type, resident \rangle. \quad (2)$$

A sensor event indicates that a certain *event_type* occurred at a certain *timestamp* as a consequence of the action of a given *resident*. For simplicity, we denote that the sensor event was *triggered* by a given resident. For instance, the sensor event $\langle '2019-05-29 \quad 11:58:01.424', fridge_door_sensor, open, Alice \rangle$ indicates that the fridge door has been opened on 2019-05-29 at 11:58:01.424 due to an action of the resident Alice.

Resident separation consists in determining whether a pair of sensor events were triggered by the same (possibly anonymous) resident, or by different residents. The objective is to facilitate activity recognition when multiple residents are doing activities in the same environment. Note that this problem does not correspond to the *data association* problem described in Section II, which aims at identifying the specific individual that activated the sensor.

A. FRAMEWORK

In this work, we assume that two residents live together in a smart home. Through an infrastructure of sensors, the smart home system detects the basic actions of residents (e.g., opening the fridge, entering the kitchen), which generate sensor events. Sensor events are processed by an artificial intelligence module to recognize the residents' activities (e.g., cooking, cleaning).

The high-level scheme of our system is represented in Figure 1. The first step *Sensor installation* consists in setting up the sensor infrastructure. As in any smart-home system,

when a new sensor is installed, it is necessary to assign a semantics to the data it produces, in order to profitably use the data for activity recognition. For example, for the recognition of the 'cooking' activity, it is useful to know when the sensors connected to the stoves have been triggered by a resident. In some cases, it may also be necessary to specify the position of the sensor within the home. We assume that the infrastructure includes a network system to convey the data to an artificial intelligence module, deployed either in the home or on the cloud, for processing.

In the second step (*Data acquisition*), for a certain initial period the system acquires unlabeled sensor events to be used for resident separation. Those data are acquired when only a single resident is doing activities in the home. Different methods may be used to automatically detect the presence of a single person in the home, based on power consumption analysis or inexpensive sensors [22], but this aspect is out of the scope of this article. Single-resident sensor events are used by our system to build the model for resident separation. As shown in our experiments reported in Section VI, a few days of unsupervised data acquisition are sufficient to reliably build the model.

Once enough data has been acquired, our system processes the data to build the model for *Resident separation*. In particular, for each couple of sensor events $\langle se_1, se_2 \rangle$ generated at time t_1 and t_2 , respectively, the model states whether se_1 and se_2 were likely triggered by the same resident or by different residents. The model of resident separation is built offline. The model can be refined incrementally when new single-resident sensor events are available by re-executing the model construction algorithm.

Finally, considering the predictions of online resident separation, the *Activity recognition* algorithm is in charge of recognizing the current activities carried out in the home, as described in Section V. We denote by *activity class* an abstract activity (e.g., eating or working), and by *activity instance* the actual occurrence of an activity of a given class during a certain time period.

B. PROBLEM FORMULATION

We model the resident separation problem as a binary classification task, in which each record is a pair of sensor events:

$$record = \langle se_1, se_2 \rangle, \quad (3)$$

and the class of the record is 1 if se_1 and se_2 were triggered by the same resident; it is 0 otherwise. Consequently, in order to evaluate the accuracy of our resident separation algorithm, we define:

- True Positive (TP): records in which the sensor events are activated by the same resident, and correctly classified by our algorithm,
- False positive (FP): records in which the sensor events are activated by different residents, and misclassified,
- True negative (TN): records in which the sensor events are activated by different residents, and correctly classified,

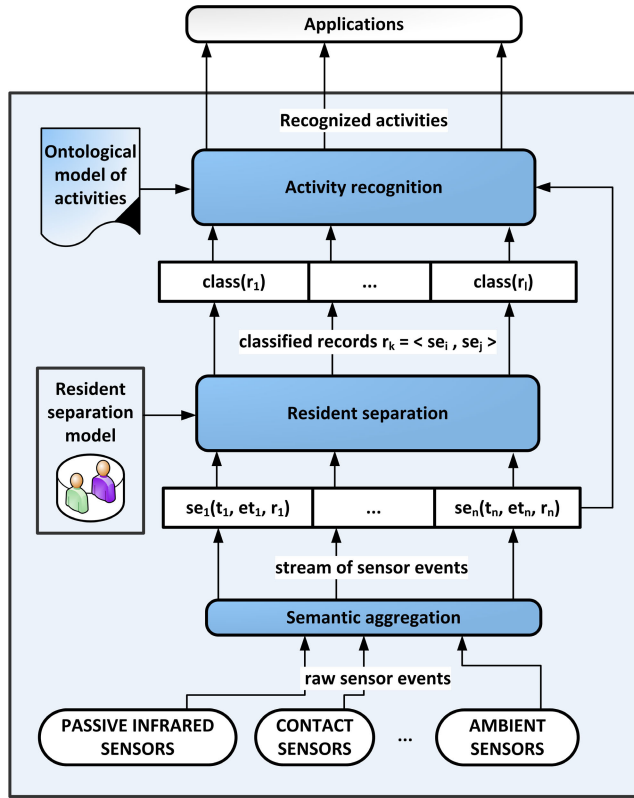


FIGURE 2. Architecture for multi-inhabitant activity recognition.

- False negative (FN): records in which the sensor events are activated by the same resident, and misclassified.

As explained before, in order to make our system feasible to real-world applications, we adopt an unsupervised approach. In fact, a supervised approach requires, in an initial period, the annotation of the activities performed by the inhabitants, but it is well known that activity annotation is costly and obtrusive. In an unsupervised system, on the other hand, no manual annotation is required: this brings benefits in terms of costs, comfort and usability of the system in a real-world context.

C. ARCHITECTURE

Figure 2 shows our architecture for multi-inhabitant activity recognition. The smart-home is instrumented with sensors to detect presence at certain locations, and interactions with items, devices and furniture. Raw sensor events are collected by the smart home infrastructure, and passed to the Semantic aggregation module. That module is in charge of applying simple preprocessing rules to transform raw data into sensor events. For example, if at time t the fridge door binary sensor S_{11} produces a raw value of type “1”, the module transforms the raw sensor data into the sensor event $\langle t, \text{fridge_door_sensor}, \text{open}, r \rangle$. The stream of sensor events is passed to the Resident separation module, that applies resident separation according to the trained model, as explained in Section IV. Finally, the stream of sensor events, as well as the classified sensor records, are used by the Activity

Algorithm Consecutive events(T):

Input: A temporal sequence of sensor events T

Output: A set R of records with associated statistics

```

1: for each couple of consecutive sensor events  $\langle se_1, se_2 \rangle \in P$  do
2:    $c = \langle se_1, se_2 \rangle$ 
3:    $c.occurrences = c.occurrences + 1$ 
4:    $R.add(c)$ 
5: end for
6: for each  $c = \langle se_1, se_2 \rangle \in R$  do
7:    $n_{se_1} = |\{se \in T : se = se_1\}|$ 
8:    $c.frequency = \frac{c.occurrences}{n_{se_1}}$ 
9: end for
10: return  $R$ 

```

FIGURE 3. Extraction of the resident separation model according to the consecutive events approach.

recognition module to recognize the activities occurring in the smart home based on an ontological model, as explained in Section V.

IV. RESIDENT SEPARATION ALGORITHMS

Since our algorithm is unsupervised, we use unlabeled records to extract statistics useful for our classification task. In particular, our intuition is that if two sensor events of given types often occur in consecutive temporal order (e.g., events of type *open_fridge_door* and *close_fridge_door*, respectively), or within a small time interval, they are likely triggered by the same person in order to perform a certain activity or a set of concurrent activities. On the contrary, if two sensor events of given types are rarely observed in temporal proximity (e.g., events of type *open_fridge_door* and *flush_the_toilet*, respectively), they are likely triggered by different residents executing different activities in the home. Hence, for building our resident separation model, we mine the training set of sensor events to extract statistical information about the co-occurrence of events of given sensor types, and use it to perform the classification of new records.

Formally, we represent the training set as a temporal sequence of sensor events:

$$T = \langle se_1, se_2, \dots, se_n \rangle,$$

where, for each couple (se_i, se_j) with $i < j$, se_i was generated before se_j . We denote by $type(se)$ the event type of the sensor event se .

We have devised different approaches and algorithms for building the resident separation model. The first approach is named **consecutive events**. With this approach, the model considers two statistical measures. The first measure is named **event type occurrences**. For each couple of event types $c = \langle et_i, et_j \rangle$, we count the number of consecutive events $\langle se_k, se_{k+1} \rangle$ in T of types et_i and et_j , irrespectively from their order:

$$n_c = \left| \left\{ \langle se_k, se_{k+1} \rangle \in T : \begin{aligned} &(type(se_k) = et_i \wedge type(se_{k+1}) = et_j) \\ &\vee (type(se_k) = et_j \wedge type(se_{k+1}) = et_i) \end{aligned} \right\} \right|. \quad (4)$$

Algorithm Temporally close events(T):**Input:** A temporal sequence of sensor events T ; a threshold ϵ (in seconds)**Output:** A set R of records with associated statistics

```

1: for each couple of sensor events  $\langle se_1, se_2 \rangle \in P$  whose time distance
   is less than  $\epsilon$  do
2:    $c = \langle se_1, se_2 \rangle$ 
3:    $c.occurrences = c.occurrences + 1$ 
4:    $R.add(c)$ 
5: end for
6: for each  $c = \langle se_1, se_2 \rangle \in R$  do
7:    $n_{se_1} = |\{se \in T : se = se_1\}|$ 
8:    $c.frequency = \frac{c.occurrences}{n_{se_1}}$ 
9: end for
10: return  $R$ 

```

FIGURE 4. Extraction of the resident separation model according to the temporally close events approach.

Then, when we observe a consecutive pair of sensor events $\langle se_k, se_{k+1} \rangle$ of types et_i and et_j , respectively, we compute the value of n_c , where $c = \langle et_i, et_j \rangle$. If the value of n_c is below a certain threshold τ , we classify those events as triggered by different residents; we classify them as triggered by the same resident otherwise.

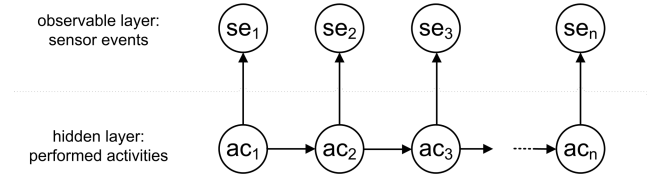
The second measure is named **event type frequency** f_c , and relies on the following formula:

$$f_c(r) = \frac{n_c}{n_{se_k}}, \quad (5)$$

where n_c is the number of occurrences of the event types c in T , and n_{se_k} is the number of occurrences of $type(se_k)$ events in T . Event type frequency is introduced because the number of occurrences of the couple of event types does not consider the numerousness of the individual types composing the couple. For example, we could have an event type et_1 with 3 occurrences only in T , which in all 3 cases is observed immediately before a sensor event of type et_2 . In this case, although the number of occurrences of events of those types is low, its event type frequency is high, possibly indicating that the two events are related to the execution of a certain activity. Then, in order to classify pairs of sensor events, we adopt the same method explained above. The algorithm pseudo-code for building the model is reported in Algorithm 3.

The second approach is named **temporally close events**. With this approach, we consider all the couples of sensor events generated within a time window of ϵ seconds, including non-consecutive sensor events. In fact, it is likely that sensors generated within a restricted time window were triggered by the activity of a single resident. We denote by $\tau(se)$ the UNIX timestamp of a sensor event. Given a couple of event types $c = \langle et_i, et_j \rangle$, we compute its **event type occurrences** measure n'_c according to the formula below:

$$n'_c = \left| \left\{ \langle se_l, se_k \rangle \in T : \begin{aligned} &(type(se_l) = et_i \wedge type(se_k) = et_j) \\ &\wedge |\tau(se_l) - \tau(se_k)| \leq \epsilon \end{aligned} \right\} \right|. \quad (6)$$

**FIGURE 5.** Hidden Markov Model for activity recognition.

The **event type frequency** measure is computed as:

$$f_c(r) = \frac{n'_c}{n_{se_k}}, \quad (7)$$

where n_{se_k} is the number of occurrences of $type(se_k)$ events in T . The classification method is the same used in the consecutive events approach.

V. RECOGNITION OF MULTI-INHABITANT ACTIVITIES

The Activity recognition module receives the stream of sensor events, as well as the results of the resident separation algorithm. The goal of activity recognition is to associate each sensor event to the class of the activity that generated it.

We propose a novel unsupervised methods for activity recognition, which relies on Hidden Markov Models (HMM). In the literature, HMMs have been extensively used for supervised activity recognition [2]. HMM is a statistical Markov model, characterized by joint probabilities among random variables representing hidden and observable states [23].

Figure 5 represents our formulation of HMM for activity recognition. The observable layer consists of the sensor events, while the hidden layer consists of the performed activities. We denote by A the set of activity classes, and by E the set of event types. The system is characterized by the following probability distributions:

- *emission probabilities* represent the probability that a sensor event type $se_i \in E$ is observed given that the current activity class is $ac_i \in A$;
- *transition probabilities* represent the probability of the current activity class being ac_{i+1} given that the previous activity class was ac_i ;
- *initial probabilities* represent the probability distribution of the first activity class ac_1 .

Based on an observed sequence of sensor events and on the HMM parameters, we use the Viterbi algorithm [24] to derive the most likely sequence of activities that may have generated the observations.

In the following, we present two approaches to instantiate the parameters of the HMM. The first, named *baseline*, is supervised, and it is commonly used for activity recognition [2]. The second is unsupervised, and it is based on automatic extraction of the HMM parameters through knowledge-based reasoning. We use the former as a baseline to evaluate the latter, which is the unsupervised HMM-based approach that we propose in this work.

A. BASELINE SUPERVISED HMM-BASED METHOD

In the baseline method, we extract the HMM parameters from a labelled dataset D of sensor data acquired during the execution of a set of activities.

The *emission probability matrix* EPM contains in each cell $[ac_i, se_j]$ the conditional probability of observing a sensor event type se_j given that the current activity class is ac_i . This value is computed according to the following formula:

$$EPM[se_j, ac_i] = \frac{\text{occurrences}(se_j, ac_i)}{\text{occurrences}(ac_i)}, \quad (8)$$

where we denote by $\text{occurrences}(se_j, ac_i)$ the number of times that se_j is observed during ac_i in D , while $\text{occurrences}(ac_i)$ is the number of times that ac_i occurs in D .

The *transition probability matrix* TPM is calculated for each cell $[ac_i, ac_j]$ according to the following formula:

$$TPM[ac_i, ac_j] = \frac{\text{occurrences}(ac_i, ac_j)}{\text{occurrences}(ac_i)}, \quad (9)$$

where we denote by $\text{occurrences}(ac_i, ac_j)$ the number of times that a sensor event emitted by ac_i is observed in D , followed by the observation of a sensor event emitted by ac_j . We denote by $\text{occurrences}(ac_i)$ the total number of sensor events emitted by ac_i in the dataset.

The *initial probability array* IPA contains for each activity class ac_i the probability that it is the first performed activity class. The formula is:

$$IPA[ac_i] = \frac{\text{occurrences}(ac_i)^{t1}}{N_{res}}, \quad (10)$$

where $\text{occurrences}(ac_i)^{t1}$ is the number of times that the first activity of a resident k in D is ac_i , and N_{res} denotes the number of residents in D .

B. UNSUPERVISED HMM-BASED METHOD

In order to avoid the acquisition of a labelled dataset of activities, we propose a novel technique to derive the HMM parameters using a knowledge-based approach. In particular, our method relies on **semantic correlations** [25], which represent probabilistic dependencies among event types and activity classes. Given an event type $et \in E$ and an activity class $ac \in A$ (where E is the set of event types and A is the set of activity classes), the semantic correlation function $SC : E \times A \rightarrow [0, 1]$ gives the probability of et being triggered by the execution of an activity of class ac . As a consequence, given any event type, SC is a probability distribution over all activity classes:

$$\forall et \in E \sum_{ac \in A} SC(et, ac) = 1. \quad (11)$$

By definition, semantic correlations correspond to the emission probabilities of our HMM. In order to derive them in an unsupervised fashion, we compute semantic correlations extending the knowledge-based method described in [26]. In particular, we re-use an OWL 2 ontology [8] modeling

activities, context, and sensors in the smart home. The ontology is available online.¹ In the ontology, activities are defined in terms of the **key objects** that are typically used during their execution.

Example 1: For instance, the ontology defines *PreparingHotMeal* as an activity that requires the usage of a *CookingInstrument*.

Moreover, the ontology is filled with instances of sensors in the smart home that are related to the usage of certain objects.

Example 2: Suppose that the smart home includes a power sensor monitoring the usage of the oven. Then, an instance of *Sensor* in the ontology is related by the property *detectsUsageOf* to an instance of the class *Oven*, which is a subclass of *CookingInstrument*.

Then, though ontological reasoning, we compute the correlations among sensor types and activity classes.

Example 3: Continuing the above example, the ontological reasoner determines that the oven power sensor is related to the activity “preparing hot meal”, since it detects the usage of a cooking instrument, which is a key object for that activity according to its ontological definition.

The method to derive semantic correlations among object sensors and activity classes is described in detail in [26]. In our extension, we also consider the presence of the user at certain locations as an indicator of a given activity. For instance, *PreparingMeal* is defined in our ontology as an activity that is typically executed in the kitchen. Hence, using the same method described above, we derive semantic correlations among presence sensors deployed at certain locations and activity classes.

We manually set transition probabilities based on common sense. Since a user normally performs the same activity for a given lapse of time before changing activity, we assign a higher probability p to transitions between the same activity class, and we uniformly distribute the remaining $(1 - p)$ probability to transitions to the other classes. However, depending on the set of considered activities, the transition matrix can be fine-tuned based on the typical order of activity execution. For instance, it is common that “eating” happens after “cooking”, while the contrary is unlikely.

Finally, we set the initial probability to the uniform distribution, since activity recognition may start at any time of the day and on every possible context condition; hence, we have no knowledge to set initial probability values based on common sense.

C. UNSUPERVISED HMM-BASED METHOD WITH RESIDENT SEPARATION

In the unsupervised HMM-based method with resident separation, we exploit the result of resident separation to assign each observed sensor event to an anonymous resident. Of course, the number of residents has in impact on the resident separation algorithm. For the sake of this work, we assume that there are two residents in the smart-home.

¹<http://sites.unica.it/domusafe/adlont/>

TABLE 1. List of activities in the CASAS dataset.

ID	Description	Actor(s)
1	Fill a medication dispenser and return items	Resident A
2	Hang up clothes in the hallway closet	Resident B
3	Move the couch and coffee table to the other side of the living room	Resident B asking help from Resident A
4	Sit on the couch and read a magazine	Resident B
5	Water plants	Resident A
6	Sweep the kitchen floor	Resident B
7	Play a game of checkers	Resident A and Resident B
8	Set out ingredients for dinner	Resident A
9	Set dining room table	Resident B
10	Sit on the couch and read a magazine	Resident A
11	Pay an electric bill	Resident A asking help from Resident B
12	Prepare a picnic basket	Resident A
13	Retrieve dishes	Resident B asking help from Resident A
14	Pack supplies in the picnic basket	Resident B
15	Pack food in the picnic basket	Resident A

The first event is assigned to an arbitrary “resident 0”. Then, if the record composed by the first and second events are classified as generated by the same resident, also the second event is assigned to “resident 0”. Otherwise, it is assigned to a different “resident 1”. We repeat the same procedure for the record composed by the second and third sensor events, and so on.

Finally, we separately apply the unsupervised HMM-based method described in Section V-B to each resident’s stream of sensor events.

VI. EXPERIMENTAL EVALUATION

In this section, we report our experiments about resident separation and multi-inhabitant activity recognition.

A. DATASET AND EXPERIMENTAL SETUP

In our experiments, we used a real-world dataset acquired and labeled by researchers of the Center for Advanced Studies in Adaptive Systems (CASAS) at Washington State University. The dataset is available online.²

The dataset was acquired in a smart-home instrumented with more than 60 sensors, including passive infrared motion sensors, temperature sensors, and sensors attached to doors, furniture, and items. Based on manual inspection, each sensor event was manually annotated with the resident that triggered it, and with the activity that he/she was performing.

The data were collected while two participants performed a set of fifteen scripted activities in a smart-home. The activities were executed by 26 pairs of residents. The considered activities are listed in Table 1.

Since our activity recognition method does not aim at recognizing the identity of the actor, we had to disregard certain activities. In particular, we disregarded activities 4 and 10 because they are identical, except for the actor’s identity. For the same reason, we disregarded activities 14 and 15, since they only differ on the actor and used tools, and the smart home does not provide any sensor to detect the

interaction with those tools. Finally, we disregarded activity 5 (watering plants), since the smart home does not provide enough sensors to recognize it. Indeed, the sensor infrastructure cannot detect events related to the interaction with the watering can or with plants, which are essential to detect that activity. Hence, in the following, we report on experiments performed using the remaining 10 activities only.

When evaluating the recognition techniques, we count the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The metrics used to evaluate the effectiveness of our algorithms are:

- *accuracy* is the percentage of correct predictions of the classifier and is defined as: $\frac{TP+TN}{TP+TN+FP+FN}$;
- *precision* = $\frac{TP}{TP+FP}$;
- *recall* = $\frac{TP}{TP+FN}$;
- *F₁ score* is the harmonic mean of precision and recall and is defined as: $\frac{2 \cdot TP}{2 \cdot TP + FP + FN}$.

B. RESIDENT SEPARATION

We model the resident separation problem as a binary classification problem, in which the positive class for an instance $\langle se_i, se_j \rangle$ is “same resident” (i.e., both se_i and se_j were triggered by the same resident), while the negative class is “different residents”.

In these experiments, we evaluated our resident separation algorithms introduced in Section IV. We applied cross-validation with 26 iterations, iteratively using one couple of residents data for test, and the other 25 couples data for training the model. We recall that, since our technique is unsupervised, while training the model for resident separation we do not consider the dataset labels (residents and activities), but only the unlabeled sensor events.

1) CONSECUTIVE EVENTS APPROACH

At first, we evaluated the *consecutive events* approach. The results of our algorithm based on *event type occurrences* are listed in Table 2. We evaluated the technique using increasing values of the τ threshold. The best results are obtained with

²<http://casas.wsu.edu/datasets/adlmr.zip>

TABLE 2. Results with consecutive events approach, event type occurrences.

Threshold τ	Errors	Correct	TP	TN	FP	FN	Accuracy
2	956	5118	4397	721	784	172	0.8426
4	755	5319	4305	1014	491	264	0.8757
6	767	5307	4207	1100	405	362	0.8737
8	765	5309	4131	1178	327	438	0.8741
10	829	5245	4049	1196	309	520	0.8635
12	843	5231	4007	1224	281	562	0.8612
14	858	5216	3981	1235	270	588	0.8587
16	883	5191	3945	1246	259	624	0.8546
18	946	5128	3863	1265	240	706	0.8443
20	998	5076	3802	1274	231	767	0.8357

TABLE 3. Results with consecutive events approach, event type frequency.

Threshold τ	Errors	Correct	TP	TN	FP	FN	Accuracy
0.005	950	5124	4426	698	807	143	0.8436
0.006	909	5165	4413	752	753	156	0.8503
0.007	887	5187	4385	802	703	184	0.854
0.008	818	5256	4374	882	623	195	0.8653
0.009	781	5293	4367	926	579	202	0.8714
0.01	772	5302	4360	942	563	209	0.8729
0.02	735	5339	4230	1109	396	339	0.879
0.03	747	5327	4152	1175	330	417	0.877
0.04	790	5284	4096	1188	317	473	0.8699
0.05	833	5241	4041	1200	305	528	0.8629
0.1	1038	5036	3744	1292	213	825	0.8291
0.15	1192	4882	3552	1330	175	1017	0.8038
0.2	1376	4698	3257	1441	64	1312	0.7735

TABLE 4. Results with temporally close events approach, event type frequency, 1 second time window.

Threshold τ	Errors	Correct	TP	TN	FP	FN	Accuracy
0.005	897	5177	4272	905	600	297	0.8523
0.006	806	5268	4243	1025	480	326	0.8673
0.007	792	5282	4223	1059	446	346	0.8696
0.008	790	5284	4217	1067	438	352	0.8699
0.009	806	5268	4196	1072	433	373	0.8673
0.01	815	5259	4177	1082	423	392	0.8658
0.05	1165	4909	3509	1400	105	1060	0.8082
0.1	1372	4702	3277	1425	80	1292	0.7741
0.15	2032	4042	2593	1449	56	1976	0.6655
0.2	2366	3708	2223	1485	20	2346	0.6105

$\tau = 8$, achieving an accuracy of 0.8741. In general, reducing the value of τ , we reduce the number of false negatives (i.e., consecutive sensor events generated by the same resident but wrongly classified), but we increase the number of false positives. Values of τ between 6 and 8 provide the best trade-off between false positives and false negatives.

We achieve slightly better results using the event type frequency measure. Results are reported in Table 3. The best results are obtained with values of threshold τ of 0.01 and 0.02. Even with event type frequency, reducing the τ value determines a reduction of false negatives and an increase of false positives.

2) TEMPORALLY CLOSE EVENTS APPROACH

Results obtained with the temporally close events approach are in line with those of the consecutive events one. In particular, the best results are obtained using the frequency-based measure; we do not report results achieved by the occurrence-based measure due to lack of space. Results using a time window of 1, 2, and 3 seconds are shown in Tables 4, 5 and 6, respectively.

TABLE 5. Results with temporally close events approach, event type frequency, 2 seconds time window.

Threshold τ	Errors	Correct	TP	TN	FP	FN	Accuracy
0.005	962	5112	4409	703	802	160	0.8416
0.01	816	5258	4344	914	591	225	0.8657
0.02	794	5280	4202	1078	427	367	0.8693
0.03	806	5268	4130	1138	367	439	0.8673
0.04	901	5173	4019	1154	351	550	0.8517
0.05	908	5166	3991	1175	330	578	0.8505
0.1	1068	5006	3628	1378	127	941	0.8242
0.15	1185	4889	3488	1401	104	1081	0.8049
0.2	1484	4590	3175	1415	90	1394	0.7557

TABLE 6. Results with temporally close events approach, event type frequency, 3 seconds time window.

Threshold τ	Errors	Correct	TP	TN	FP	FN	Accuracy
0.005	1041	5033	4467	566	939	102	0.8286
0.01	929	5145	4425	720	785	144	0.8471
0.02	820	5254	4324	930	575	245	0.865
0.03	811	5263	4265	998	507	304	0.8665
0.04	812	5262	4219	1043	462	350	0.8663
0.05	793	5281	4151	1130	375	418	0.8694
0.06	850	5224	4063	1161	344	506	0.8601
0.07	867	5207	4044	1163	342	525	0.8573
0.08	928	5146	3967	1179	326	602	0.8472
0.09	975	5099	3907	1192	313	662	0.8395
0.1	981	5093	3883	1210	295	686	0.8385
0.15	976	5098	3716	1382	123	853	0.8393
0.2	1095	4979	3591	1388	117	978	0.8197

Overall, the achieved accuracy is slightly lower than the one obtained with the consecutive events approach. This result may be due to the nature of the dataset, which contains scripted activities carried out in short time periods. When training the model with activities carried out in naturalistic environments for longer time periods, we expect to obtain higher accuracy using the temporally close events approach.

In the rest of the experiments, we adopt the consecutive events approach, using the event type frequency measure. Overall, with that method we achieved an accuracy close to 88%. Compared to other approaches, we consider this result as a positive one. For instance, the supervised HMM-based method for data association proposed in [9] achieves accuracy of around 90%. The accuracy of our method is slightly lower, but our method has the advantage of not requiring the acquisition of a training set of sensor events labeled with the resident that triggered the events.

C. ACTIVITY RECOGNITION

In the following experiments, we evaluate our activity recognition method based on resident separation. For each sensor event se , the objective is to identify the class of the activity that triggered it.

1) BASELINE SUPERVISED HMM-BASED METHOD

As explained in Section V-A, we use this supervised method as a baseline to evaluate our unsupervised technique. For the sake of this experiment, we extract the HMM parameters from the data, using the same cross validation approach explained in Section VI-B. However, the activities in the dataset were scripted imposing a fixed order of activity execution. This fixed order is not representative of a real-world situation,

TABLE 7. Unsupervised HMM-based method.

Activity	Precision	Recall	F_1	TP	FP	FN
1	0.8988	0.838	0.8673	595	67	115
2	0.7427	0.7494	0.7461	664	230	222
3	0.6245	0.9341	0.7486	326	196	23
6	0.4766	0.52	0.4973	417	458	385
7	0.8025	0.6876	0.7406	581	143	264
8	0.695	0.615	0.6526	139	61	87
9	0.4335	0.5022	0.4654	225	294	223
11	0.4951	0.5984	0.5419	301	307	202
12	0.7796	0.8611	0.8183	527	149	85
13	0.7595	0.4437	0.5601	319	101	400
Average accuracy = 0.6711						

in which humans perform most activities in variable order. Hence, extracting the TPM from the data would determine a strong bias, which would affect the results. For this reason, we manually set the TPM matrix with diagonal values d , uniformly distributing the remaining probability values among the other cells. We choose the value $d = 0.9$ because it achieves the highest recognition rates in our experiments.

We evaluate the supervised HMM approach without data association; i.e., considering the whole sensor data stream as triggered by a unique resident. The achieved average accuracy is 0.6962.

2) UNSUPERVISED HMM-BASED METHOD

In this experiment, we evaluate the unsupervised HMM-based technique described in Section V-B. The achieved results are reported in Table 7. Overall, the achieved accuracy is 0.6711. Compared with the supervised HMM-based method without data association, the unsupervised method is less accurate. This result is explained by the fact that semantic associations derived by ontological reasoning are less accurate than emission probabilities computed from the data. Indeed, semantic correlations represent generic relationships among sensor events and activities. On the contrary, emission probabilities extracted from the data are fine-tuned to the specific environment.

By closely inspecting the results, we notice that certain activities are particularly hard to recognize. The lowest F_1 score was achieved by activity 9 (set dining room table), probably because it is an activity involving several movements and items use, that are difficult to distinguish from other activity executions. On the contrary, activity 1 (fill medication dispenser) achieved the highest F_1 score, probably because it involves the usage of specific items, and the presence at a specific location.

3) UNSUPERVISED HMM-BASED METHOD WITH RESIDENT SEPARATION

Finally, we evaluate the HMM-based technique described in Section V-C, which exploits our resident separation method.

Results are shown in Table 8. Overall, the achieved accuracy is 0.7213. Hence, our unsupervised technique achieves higher accuracy than the baseline supervised HMM-based method. An additional benefit of our technique is that it does

TABLE 8. Unsupervised HMM-based method with resident separation.

Activity	Precision	Recall	F_1	TP	FP	FN
1	0.9357	0.9634	0.9493	684	47	26
2	0.761	0.8555	0.8055	758	238	128
3	0.5895	0.8023	0.6796	280	195	69
6	0.6393	0.4863	0.5524	390	220	412
7	0.862	0.8426	0.8522	712	114	133
8	0.6565	0.6681	0.6623	151	79	75
9	0.4636	0.625	0.5323	280	324	168
11	0.5503	0.5109	0.5299	257	210	246
12	0.7802	0.9281	0.8478	568	160	44
13	0.739	0.4451	0.5556	320	113	399
Average accuracy = 0.7213						

not need the acquisition of labelled datasets of activities and sensor events. Moreover, the introduction of resident separation into the unsupervised HMM-based method improves accuracy of 0.05.

Compared to other techniques applied to the same dataset, the accuracy obtained by our unsupervised method is close to the one obtained by the supervised HMM-based technique reported in [13]. Indeed, that method achieves 0.7315 accuracy assuming exact data separation. On the contrary, our method does not assume neither data separation, nor the existence of a labeled training set. Our method also outperforms the supervised technique based on Conditional Random Fields proposed in [11], which achieves 0.6416 accuracy.

By comparing the results of the unsupervised HMM-based method with and without resident separation, we notice that resident separation improves the F_1 score of most activities. The only activities that are negatively affected by resident separation are activities 3, 11, and 13. In fact, in all those activities, a resident asks help to the other resident to complete the task. Hence, they are group activities, which do not benefit from separating the actors. If we consider only the individual (i.e., non-group) activities in the dataset, the introduction of resident separation increases the accuracy from 0.7 to 0.77. These results show that the introduction of our unsupervised resident separation technique determines a relevant improvement of recognition rates.

VII. CONCLUSION AND FUTURE WORK

In this article, we tackled the challenging issue of recognizing multi-resident activities based on sensor data acquired from a smart-home infrastructure. We have proposed a hybrid method based on unsupervised data mining, and on ontological reasoning. To the best of our knowledge, this is the first effort to recognize multi-resident activities without the need of labeled training data. Experimental results show that our method achieves an accuracy close to the one of a state-of-the-art supervised technique. Indeed, our unsupervised method achieves 0.7231 average accuracy, while the state-of-the-art supervised technique achieves 0.7315 average accuracy.

Several research challenges remain open. First of all, our current resident separation method assumes the presence of

at most two persons in the home. Extending our method to more residents is not trivial, both for the technical issues introduced by the extension, and for the lack of extensively labeled datasets to evaluate the technique with more than two inhabitants. Another limitation of our resident separation framework is the lack of specific support for handling collaborative activities, in which multiple persons execute the tasks needed to perform a given activity. As future work, we will also investigate methods to explicitly model interaction among inhabitants, in order to effectively recognize collaborative activities.

REFERENCES

- [1] N. Davies, D. P. Siewiorek, and R. Sukthankar, "Activity-based computing," *IEEE Pervas. Comput.*, vol. 7, no. 2, pp. 20–21, Apr. 2008.
- [2] J. Ye, S. Dobson, and S. McKeever, "Situation identification techniques in pervasive computing: A review," *Pervas. Mobile Comput.*, vol. 8, no. 1, pp. 36–66, Feb. 2012.
- [3] A. Benmansour, A. Bouchachia, and M. Feham, "Multioccupant activity recognition in pervasive smart home environments," *ACM Comput. Surv.*, vol. 48, no. 3, p. 34, 2015.
- [4] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *IEEE ASSP Mag.*, vol. 3, no. 1, pp. 4–16, Jan. 1986.
- [5] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. ICML*, San Mateo, CA, USA: Morgan Kaufmann, 2001, pp. 282–289.
- [6] K. D. Feuz and D. J. Cook, "Real-time annotation tool (RAT)," in *Proc. AAAI Workshops*, Jun. 2013, pp. 2–10.
- [7] D. Roggen, A. Calatroni, M. Rossi, T. Holleczeck, K. Forster, G. Troster, P. Lukowicz, D. Bannach, G. Pirkil, A. Ferscha, J. Doppler, C. Holzmann, M. Kurz, G. Holl, R. Chavarriaga, H. Sagha, H. Bayati, M. Creatura, and J. D. R. Millan, "Collecting complex activity datasets in highly rich networked sensor environments," in *Proc. 7th Int. Conf. Netw. Sens. Syst. (INSS)*, Jun. 2010, pp. 233–240.
- [8] D. Riboni and C. Bettini, "OWL 2 modeling and reasoning with complex human activities," *Pervas. Mobile Comput.*, vol. 7, no. 3, pp. 379–395, Jun. 2011.
- [9] A. S. Crandall and D. J. Cook, "Using a hidden Markov model for resident identification," in *Proc. 6th Int. Conf. Intell. Environments*, Jul. 2010, pp. 74–79.
- [10] D. J. Cook, A. Crandall, G. Singla, and B. Thomas, "Detection of social interaction in smart spaces," *Cybern. Syst.*, vol. 41, no. 2, pp. 90–104, Apr. 2010.
- [11] K. Hsu, Y. Chiang, G. Lin, C. Lu, J. Y. Hsu, and L. Fu, "Strategies for inference mechanism of conditional random fields for multiple-resident activity recognition in a smart home," in *Proc. IEA/AIE in Lecture Notes in Computer Science*, vol. 6096, Berlin, Germany: Springer, 2010, pp. 417–426.
- [12] R. Chen and Y. Tong, "A two-stage method for solving multi-resident activity recognition in smart environments," *Entropy*, vol. 16, no. 4, pp. 2184–2203, Apr. 2014.
- [13] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, "Recognizing independent and joint activities among multiple residents in smart environments," *J. Ambient Intell. Humanized Comput.*, vol. 1, no. 1, pp. 57–63, Mar. 2010.
- [14] M. A. U. Alam, N. Roy, A. Misra, and J. Taylor, "CACE: Exploiting behavioral interactions for improved activity recognition in multi-inhabitant smart homes," in *Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2016, pp. 539–548.
- [15] S. N. Tran, Q. Zhang, V. Smallbon, and M. Karunanithi, "Multi-resident activity monitoring in smart homes: A case study," in *Proc. IEEE Int. Conf. Pervas. Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2018, pp. 698–703.
- [16] M. Prosserger and A. Bouchachia, "Multi-resident activity recognition using incremental decision trees," in *Proc. ICAIS in Lecture Notes in Computer Science*, vol. 8779, Berlin, Germany: Springer, 2014, pp. 182–191.
- [17] L. McCowan, D. Gatica-Perez, S. Bengio, G. Lathoud, M. Barnard, and D. Zhang, "Automatic analysis of multimodal group actions in meetings," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 3, pp. 305–317, Mar. 2005.
- [18] L. Wang, T. Gu, X. Tao, and J. Lu, "Sensor-based human activity recognition in a multi-user scenario," in *Proc. Am/In Lecture Notes in Computer Science*, vol. 5859, Berlin, Germany: Springer, 2009, pp. 78–87.
- [19] G. Demiris, D. P. Oliver, G. Dickey, M. Skubic, and M. Rantz, "Findings from a participatory evaluation of a smart home application for older adults," *Technol. Health Care*, vol. 16, no. 2, pp. 111–118, May 2008.
- [20] Y.-T. Chiang, K.-C. Hsu, C.-H. Lu, L.-C. Fu, and J. Y.-J. Hsu, "Interaction models for multiple-resident activity recognition in a smart home," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 3753–3758.
- [21] T. Gu, Z. Wu, L. Wang, X. Tao, and J. Lu, "Mining emerging patterns for recognizing activities of multiple users in pervasive computing," in *Proc. 6th Annu. Int. Conf. Mobile Ubiquitous Syst., Comput., Netw. Services*, Jul. 2009, pp. 1–10.
- [22] Z. Chen, C. Jiang, and L. Xie, "Building occupancy estimation and detection: A review," *Energy Buildings*, vol. 169, pp. 260–270, Jun. 2018.
- [23] P. Norvig and S. Russell, "Artificial intelligence. A modern approach," in *Prentice-Hall Series in Artificial Intelligence*. London, U.K.: Pearson, 2003.
- [24] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 2, pp. 260–269, Apr. 1967.
- [25] G. Civitarese, C. Bettini, T. Szytler, D. Riboni, and H. Stuckenschmidt, "NewNECTAR: Collaborative active learning for knowledge-based probabilistic activity recognition," *Pervas. Mobile Comput.*, vol. 56, pp. 88–105, May 2019.
- [26] D. Riboni, T. Szytler, G. Civitarese, and H. Stuckenschmidt, "Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning," in *Proc. ACM Int. Joint Conf. Pervas. Ubiquitous Comput.*, Sep. 2016, pp. 1–12.



DANIELE RIBONI (Member, IEEE) received the Ph.D. degree in computer science from the University of Milano. He is currently an Associate Professor with the Department of Mathematics and Computer Science, University of Cagliari. He was the Vice Technical Program Chair of the IEEE PerCom, in 2016 and the Program Chair of the 14th International Conference on Intelligent Environments, in 2018. His main research interests include knowledge management for mobile and pervasive computing, context awareness, activity recognition, and data privacy. He is the author of over 70 publications and his contributions appear in highly-ranked international journals and conference proceedings. He is a Principal Investigator of different projects on AI and healthcare.



FLAVIA MURRU received the bachelor's and master's degrees in computer science from University of Cagliari, in 2016 and 2019, respectively. She is currently working with IT company as a Software Developer. Her research interests include activity recognition, machine learning, web programming, and architectures.

...