Human Activity Discovery and Recognition Using Probabilistic Finite-State Automata

Kévin Viard, Maria Pia Fanti, *Fellow, IEEE*, Gregory Faraut, *Member, IEEE*, and Jean-Jacques Lesage, *Member, IEEE*

***Abstract*— Ambient assisted living and smart home technologies are a good way to take care of dependent people whose number will increase in the future. They allow the discovery and the recognition of human's activities of daily living (ADLs) in order to take care of people by keeping them in their home. In order to consider the human behavior nondeterminism, probabilistic approaches are used despite difficulties encountered in model generation and probabilistic indicators computing. In this article, a global method based on probabilistic finite-state automata and the definition of the normalized likelihood and perplexity is proposed to manage ADLs discovery and recognition. In order to reduce the computational complexity, some results about a simplified normalized likelihood computation are proved. A real case study showing the efficiency of the proposed method is discussed.**

***Note to Practitioners*—This article is motivated by the problem of the automatic recognition of activities that are daily performed by elderly or disabled people in a smart dwelling. The set of activities to be recognized is defined by a medical staff (e.g., to prepare meal, to do housework, to take leisure, etc.) and correspond to pathologies that have to be monitored by doctors (e.g., loss of memory, loss of mobility, etc.). The proposed method is based on a systematic procedure of offline construction of a model for each activity to be monitored (the activity discovering step). The online recognition of activities actually performed (the activity recognition step) is afterward based on these models of activities. Since the human behavior is nondeterministic, and may even be irrational, probabilistic activity models are built from a learning database. In the same way, probabilistic indicators are used for determining online the most probable activities actually performed. The efficiency of the proposed approach is illustrated through a case study performed in a smart living lab.**

***Index Terms*— Activity of daily living (ADL), activity discovery (AD), activity recognition (AR), normalized likelihood, probabilistic finite-state automata (PFA), smart home.**

Manuscript received January 16, 2020; revised March 30, 2020; accepted April 8, 2020. Date of publication May 8, 2020; date of current version October 6, 2020. This article was recommended for publication by Editor Q. Zhao upon evaluation of the reviewers' comments. *(Corresponding author: Kévin Viard.)*

Kévin Viard, Gregory Faraut, and Jean-Jacques Lesage are with the Université Paris-Saclay, ENS Paris-Saclay, LURPA, 94235 Cachan, France (e-mail: kevin.viard@ens-paris-saclay.fr; gregory.faraut@ens-paris-saclay.fr; jean-jacques.lesage@ens-paris-saclay.fr).

Maria Pia Fanti is with the Dipartimento di Ingegneria Elettrica e dell'Informazione, Politecnico di Bari, 70125 Bari, Italy (e-mail: mariapia.fanti@poliba.it).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2020.2989226

I. Introduction

According to several demography studies, the dependence rate of the world population is continuously increasing since 2010. In 2050, the part of the population aged 60 or more will rise to 30% in the majority of countries [1]. This societal evolution is becoming an important human and economic issue for next years. In fact, current health and welfare institutions will not be sufficient to treat this proportion of elderly people. Hence, severe pressure on the public healthcare sector and lack of adequate facilities are driving the way in which health services are delivered to the patients [2], [3]. Therefore, alternative solutions have to be found and rapidly developed in order to supply help and independence to people suffering from not too severe pathologies.

Smart homes, which integrate medical equipment and other ambient assisted living technologies, can play a lead role in revolutionizing the way in which healthcare services are being provided to the elderly people [4], [5]. Health at home systems is an efficient possible solution that consists of keeping old people at home as long as possible, thanks to an automatic monitoring of their daily life.

The activities of daily living (ADLs) analysis is one of the main investigation fields in health-assistive smart homes and smart environments [6]–[15]. To live independently at home, individuals need to be able to complete ADLs such as eating, dressing, cooking, drinking, and taking medicine. Current studies about ADLs mainly treat two problems: activity discovery (AD) and activity recognition (AR) [6]. In particular, AD is a technique employed to reduce the need for expert knowledge by using learning algorithms to discover activities in sensor events raw sequences [6]–[8]. In addition, the objective of AR is to detect the activity actually performed by the inhabitant [9]–15]. The generally accepted approach to AR is to design and/or use machine learning techniques to map a sequence of sensor events to a corresponding activity label.

Kim *et al.* [9] and van Kasteren *et al.* [12] describe all inhabitant activities by only one hidden Markov model (HMM), one of the most frequent models used in literature for AD and recognize each activity by applying the Viterbi algorithm [11]. Unfortunately, the complexity of the model drastically increases with the number of activities and sensors. Furthermore, the used model has no intermediary semantic levels between activities and sensors and the precision of the recognition is not guaranteed. In a previous work [8], each activity is modeled by only one HMM for discovery purpose and in [10] a new probabilistic indicator for AR, called normalized likelihood, is proposed.

Kellokumpu *et al.* [13] and Hongeng *et al.* [14], [15] present a system that recognizes a set of activities modeled by HMMs. Moreover, activities are classified by a probability that allows recognizing the performed activity as being the one, which is represented by the most probable model. Typically, the works focusing on recognition give few details about how the probabilistic models are built. By using a similar model, article [16] addresses the problem of recognizing ADLs in smart homes in a hidden semi-Markov model (HSMM) framework. In addition, comparing different kinds of HMMs, the authors show that the proposed novel form of HSMM, called Coaxian Hidden Semi-Markov Model, performs online activity classification and segmentation from a segmented training data. The performance of the model is compared with various counterparts by a likelihood computation that can be used only if the systems exhibit the same event sets. On the contrary, typically activities are linked to different events (i.e., sensors) because they occur in different home areas and are detected by using equipment located in different areas. A common drawback of these approaches is that they do not explain how the probabilistic model of scenarios is built and, typically, they are manually built by an expert.

This article proposes a novel approach for AR and AD in order to fill the gap that is present in the works of the related literature. Indeed, most contributions suggest AR methodologies, but they do not give precise information on the way activities are discovered and whether the recognition works well when there are variations during the performance of the activities.

The novelty of this article is threefold.

First, the activities are modeled by using probabilistic finite-state automata (PFA), a superclass of HMMs [17] that are powerful modeling techniques when the system states are partially or completely unknown. On the other hand, PFA models are chosen for three main reasons: 1) the structure of the PFA can be automatically deducted from the input data of AD, hence, the use of HMMs is not necessary; 2) the proposed framework allows automatically building the models of scenarios by overcoming the drawback of the Markov models; and 3) the PFA allows exploiting existing tools and consolidated theoretical results that give the possibility of automatically building the activity models (see for example the algorithms presented in [30] and [31]).

Second, in order to identify ADLs linked to different sets of events and having some nondeterministic variations, a new method is developed to allow distinguishing activities. To this aim, we adopt the perplexity evaluation by introducing the normalized likelihood to select the most probable activity. This methodology allows overcoming the limit of considering models sharing the same set of events. Moreover, in order to reduce the computational complexity some results simplifying the normalized likelihood computation are proved.

Third, the presented methods are applied to a real smart flat provided by the ENS Paris-Saclay (France) and experimental results are discussed for both AD and AR methodologies.

Finally, an important benefit of the proposed approach concerns the collection of the needed expert data in comparison with the methods of the related literature. Indeed, the proposed method allows building the AD module and performing the recognition by using short strings of data: hence, the approach can be suitably applied in real situations.

In this article, assumptions and related works are first presented in Section II and Section III proposes a formal statement of the problem. Sections IV and V present the AD and AR methods, respectively, and Section VI discusses a real case study. Finally, Section VII draws the conclusions.

II. Literature Review and Assumptions

In this section, we start by listing the main assumptions of the proposed framework, in relation to the results and the contributions of the related literature. The methodology proposed for solving the AR and AD problems is based on the following assumptions.

1) Only binary sensors are used for observing patient activities.
2) Human behavior is nondeterministic and may even be irrational.
3) The smart dwelling is supposed to be occupied by a single inhabitant.
4) The database which has been recorded during the learning period, and which is the input data of AD, does not include the knowledge of activities which have been actually performed.

The rationality of considering such assumptions is based on the following justifications that are discussed on the basis of the literature review.

Assumption 1: In most works of the related literature, the use of cameras is needed in at least one step of ADLs monitoring [18]–[20]. Indeed, cameras provide information of very high level of semantics for discovering activities. Nevertheless, they are often considered too intrusive and raise problems of acceptance by monitored people [21]. This is the main reason why cameras are not used in this article, but only binary sensors, such as motion detectors or door barriers. Such sensors are furthermore low cost, what is very interesting in the context of health at home, which is becoming a mass problem.

Assumption 2: Human behavior is nondeterministic and characterized by small changes every day. Therefore, ADL models which are not robust to minor variations, like data mining approach [7], [22], are nonconsidered to maximize the robustness and applicability of the proposed method.

Assumption 3: In order to assume that more inhabitants are in the dwelling, it is necessary to consider that each inhabitant wears a sensor that allows identifying himself [e.g., a radio-frequency identification (RFID) sensor] and therefore knowing who has generated which event. Such wearable sensors are very often used for AD and AR [20], but the efficiency of this kind of sensors strongly depends on the ability and the willingness of the patients to wear them every day, and sometimes during the night. As in the case of cameras, this sensor technology also raises some problems of acceptance and it is sometimes not compatible with the pathology of patients to be monitored (e.g., loss of memory).

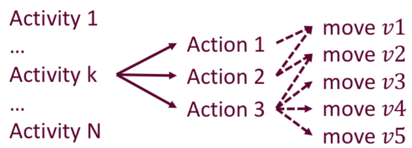

Fig. 1. Hierarchical decomposition of activities in actions and moves.

Hence, according to Assumption 1, only binary sensors are used for both AD and AR and it is necessary to assume that a single inhabitant is living in the smart home. This assumption is quite restrictive but allows proposing a complete solution for AD and AR that is based on the use of binary sensors only.

Assumption 4: In the main part of the works based on assumptions 1–3, a perfect knowledge of activities performed by the inhabitant during the learning period is needed [12], [23]. This information is in practice very difficult to obtain. In [23], the monitored patient indicates what activity is performed. Of course, the efficiency of this approach is compared with the ability and the willingness of the person to declare his activity: in general, numerous reported activities errors are introduced in the database [23]. In other works [19], experts are in charge of enriching the database by studying sensor logs or using cameras exclusively during the learning phase. This approach is expensive, intrusive, and not accurate. In both cases, the labeling step is difficult and unreliable: in this article, the AD does not require the knowledge of actually performed activities during the learning phase.

III. Framework and Notations for the AD and AR

In this section, we describe the main components of the proposed discovery and recognition framework that allows automatically building models representing activities, models that are used to recognize the ADL performed by the inhabitant. This framework has to consider the privacy of the inhabitant, the nondeterminism of the human behavior, and the different ways to perform the same ADL, i.e., the slight variations to perform an activity has to lead to a recognition of the activity. Hence, we describe the structure of the AD and AR tasks and the used notations for the definition of the PFA.

A. Framework of the AD and AR

ADLs are regularly performed by a person and can be decomposed into several actions [18], [24], [25]. For instance, "cooking" can be decomposed in "preparing pasta," "preparing a ready-cook dish," "ordering meal on the net," etc. Moreover, actions can be described as a sequence of elementary moves. The hierarchical decomposition of activities in actions and observable moves is represented in Fig. 1. Note that an observable move can be linked to several actions and can be observed by binary sensors.

The structure of the framework developed to perform AD and AR procedures is represented in Fig. 2 showing the main modules of the strategy: the AD and AR modules.

In some preliminary operations, the set of activities to be monitored are determined and the related actions and moves are singled out. On the basis of the considered moves, the sensors are chosen and positioned in the house.

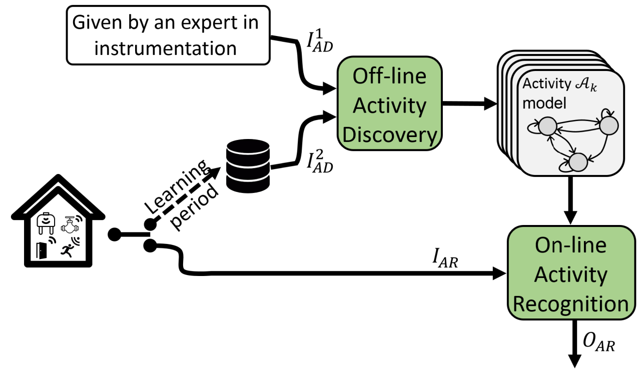

Fig. 2. Structure of the AD and AR.

The AD module is applied offline for a learning period with the objective of generating formal models of ADLs. To this aim, in coherence with assumptions presented in Section II, two sets of inputs denoted I^1_{AD} and I^2_{AD} are employed.

1) I^1_{AD} symbolizes the items that are provided by an expert.
   a) The set of activities to be monitored.
   b) The set of actions composing each activity.
   c) The moves connected with the actions and linked with the sensors in the dwelling.
   d) I^2_{AD} is a log obtained by observing an inhabitant life during a learning period, i.e., streaming data represented by sequences of events detected by the sensors.

Starting from such inputs, the AD module provides the set $A = \{\mathcal{A}_k\}$ of PFA models of the activities. Note that in this article, symbol $\in \mathcal{A}_k \in A$ is used to represent both the activity and the PFA modeling this activity.

The AR module works online to identify in real time the activity actually performed by the inhabitant. The inputs of this module are the following.

1) The observation I_{AR} provided by the binary sensors of the real-time behavior of the monitored person.
2) The set of the activity models $A = \{\mathcal{A}_k\}$ obtained by the AD.

The output O_{AR} of the AR module is the recognition of the activity performed by the inhabitant.

B. PFA: Notations and Definitions

In order to describe the nondeterministic behavior of the house inhabitant, the activities are modeled in a PFA framework as follows [17].

Definition 1: A PFA $\mathcal{A}_k$ is a tuple $\mathcal{A}_k = \langle Q_{\mathcal{A}_k}, \Sigma_{\mathcal{A}_k}, \delta_{\mathcal{A}_k}, I_{\mathcal{A}_k}, F_{\mathcal{A}_k}, P_{\mathcal{A}_k}\rangle$, where:

1) $Q_{\mathcal{A}_k}$ is a finite nonempty set of states q;
2) $\Sigma_{\mathcal{A}_k}$ is a nonempty alphabet of events e;
3) $\delta_{\mathcal{A}_k} \subseteq Q_{\mathcal{A}_k} \times \Sigma_{\mathcal{A}_k} \times Q_{\mathcal{A}_k}$ is a set of transitions;
4) $I_{\mathcal{A}_k} : Q_{\mathcal{A}_k} \to [0, 1]$: the initial-state probabilities;
5) $P_{\mathcal{A}_k} : \delta_{\mathcal{A}_k} \to [0, 1]$: the transition probabilities;
6) $F_{\mathcal{A}_k} : Q_{\mathcal{A}_k} \to [0, 1]$: the final-state probabilities.

Note that $I_{\mathcal{A}_k}$, $P_{\mathcal{A}_k}$, and $F_{\mathcal{A}_k}$ are functions such that [17]

$$\sum_{q \in Q_{\mathcal{A}_k}} I_{\mathcal{A}_k}(q) = 1 \tag{1}$$

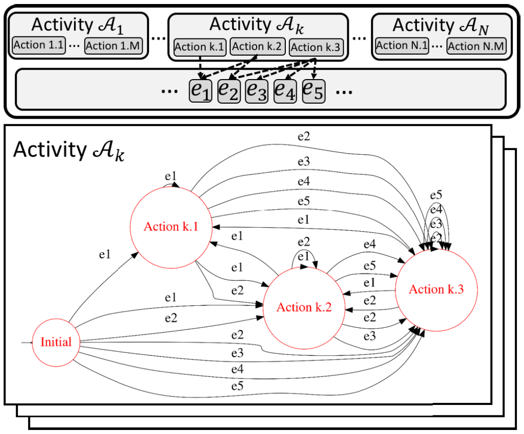

Fig. 3. Generation of model structure from the activities semantic description.

and

$$\forall q \in Q_{\mathcal{A}_k}, F_{\mathcal{A}_k}(q) + \sum_{e \in \Sigma_{\mathcal{A}_k}, q' \in Q_{\mathcal{A}_k}} P_{\mathcal{A}_k}(q, e, q') = 1. \quad (2)$$

When used in subscript of a symbol, $\mathcal{A}_k$ represent the PFA to which the symbol is linked.

Now, according to Definition 1, the PFA model of activity $\mathcal{A}_k$ is specified as follows.

1) $q_l \in Q_{\mathcal{A}_k}$ represents an action performed during the activity; transitions starting from this state represent the probabilities to switch from this action to another one.
2) $q_0 \in Q_{\mathcal{A}_k}$ is the initial dummy state that we consider as the initial condition of the activities where no action is performed.
3) An event $e_i \in \Sigma_{\mathcal{A}_k}$ describes a move that may occur in a state of activity $\mathcal{A}_k$.
4) A transition $(q_m, e_j, q_s) \in \delta_{\mathcal{A}_k}$ if starting from state q_m event e_j may occur and activity $\mathcal{A}_k$ reaches state q_s.
5) Each activity starts in the initial dummy state q_0 with probability equal to 1, it holds $I_{\mathcal{A}_k}(q_0) = 1$ and $I_{\mathcal{A}_k}(q_l) = 0$ for each $q_l \in Q_{\mathcal{A}_k}$.
6) $P_{\mathcal{A}_k}(q_m, e_j, q_s)$ is the probability, in activity $\mathcal{A}_k$, of observing event e_j and destination state q_s starting from state q_m.
7) $F_{\mathcal{A}_k}(q_m) = 0$ for each $q_m \in Q_{\mathcal{A}_k}$.

An example of PFA is illustrated in Fig. 3.

On the basis of Definition 1, several objects can be defined as follows.

1) $w = e_i e_j \ldots e_n$ is a sequence of observed events and the length of w (denoted $|w|$) corresponds to the number of events in the sequence.
2) $\theta = (q_l, e_i, q_m, e_j, q_s, \ldots, q_p, e_n, q_r)$ is a path of transitions for w in $\mathcal{A}_k$, i.e., the sequence of transitions $(q_l, e_i, q_m), (q_m, e_j, q_s), \ldots, (q_p, e_n, q_r) \in \delta_{\mathcal{A}_k}$ consistent with $w = e_i e_j \ldots e_n$.
3) $\mathcal{L}(w)$ denotes the language, i.e., a set of sequences of events generated from the observed sequence w.
4) $\Sigma^m_{\mathcal{A}_k}$ is the set of all possible sequences of length m which can be generated with symbols of the alphabet $\Sigma_{\mathcal{A}_k}$. For example, let $\Sigma_{\mathcal{A}_k} = \{e_1, e_2, e_3\}$ be an alphabet, then $\Sigma^3_{\mathcal{A}_k} = \{e_1e_1e_1, e_1e_1e_2, e_1e_1e_3, e_1e_2e_1, \ldots, e_3e_3e_3\}$ is the set of all possible sequences of length 3.
5) $w_{p_k} \subset w$ is the projection of sequence w on alphabet $\Sigma_{\mathcal{A}_k}$ of activity $\mathcal{A}_k$, where the projection of $w \in \Sigma^*$ on alphabet $\Sigma_{\mathcal{A}_k}$ is defined as follows [29]:

$$\text{Proj} : \Sigma^* \to \Sigma^*_{\mathcal{A}_k} \quad (3)$$

with

$$\text{Proj}(\varepsilon) := \varepsilon$$
$$\text{Proj}(e) := \begin{cases} e, & \text{if } e \in \Sigma_{\mathcal{A}_k} \\ \epsilon, & \text{if } e \in \Sigma \backslash \Sigma_{\mathcal{A}_k} \end{cases}$$
$$\text{Proj}(we) := \text{Proj}(w)\text{Proj}(e) \text{ for } w \in \Sigma^*, \quad e \in \Sigma_{\mathcal{A}_k}$$

where Σ is the set of system events, Σ^* and $\Sigma^*_{\mathcal{A}_k}$ represent the Kleene-closure [29] of Σ and $\Sigma_{\mathcal{A}_k}$, respectively, and ϵ is the empty symbol, i.e., the sequence of length 0. In other words, the mathematical projection function allows obtaining a sequence including only events of a specific activity $\mathcal{A}_k$. For instance, let $w = e_1e_2e_1e_3e_2e_4e_1e_5e_2e_3$ be an observed sequence: the projection of w on $\Sigma_{\mathcal{A}_K} = \{e_1, e_3, e_5\}$ is the sequence $w_{p_k} = e_1e_1e_3e_1e_5e_3$.
6) $\mathcal{L}(w)$ is a language generated from sequence w and composed by all substrings of w such that $|w| \geq 2$.

IV. Activity Discovery

In this section, the AD method to model ADLs in a PFA framework is developed by following three steps: 1) generating the PFA structure; 2) analyzing and processing a set of streaming data; and 3) computing the probabilities of the PFA model.

A. Generation of the PFA Structure

Starting from the hierarchical description of each activity (given in Fig. 1), the PFA structure of each activity $\mathcal{A}_k$ is determined by building the associated transition digraph (direct graph) $D(\mathcal{A}_k) = (Q_{\mathcal{A}_k}, \delta_{\mathcal{A}_k})$. More precisely, the set of nodes of $D(\mathcal{A}_k)$ corresponds to the states set $Q_{\mathcal{A}_k}$ and the set of direct arcs is associated with the transitions in $\delta_{\mathcal{A}_k}$, i.e., there exists a direct arc starting from node q_g and ending to node q_h, if $(q_m, e_j, q_s) \in \delta_{\mathcal{A}_k}$. Fig. 3 shows an example of digraph describing the structure of an activity.

B. Analysis and Process of Streaming Data

This section briefly describes the approach for processing the streaming data in the context of the smart home data set. The sensors embedded in the considered smart apartments are binary sensors that are in two states — "ON" and "OFF." Then they can generate two events: one is linked to the rising edge of the binary information (sensor|1), the other one is linked to the falling edge (sensor|0) (see as an example Fig. 4).

In the related literature, different approaches are proposed for processing streaming data represented by sequences of the sensor firings [26], [27]. The explicit windowing is not adapted to our hypotheses because it requires an expert to

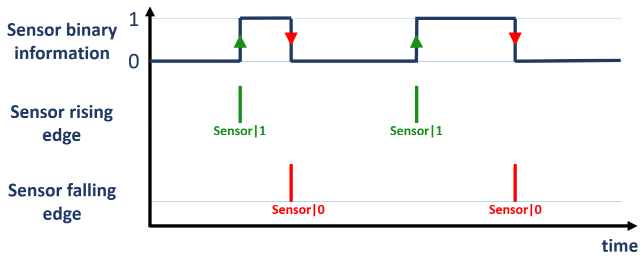

Fig. 4. Event emission from sensor binary information.

segment sequence. The time windowing could be interesting in case of timed approach, but our approach is event-based. Furthermore, with time windowing, some windows could have no event. By consequence, we choose the sensor event-based windowing approach that consists in dividing the sequence into windows containing a fixed number of sensor events. In this case, the windows vary in their duration and this is fine considering that during the performance of activities, multiple sensors could be triggered, while during silent periods, there will not be many sensor firings. Hence, we consider a long observation that we divide in windows w containing a fixed number of events and we denote by w_{p_k} the projection of w on activity alphabet $\Sigma_{\mathcal{A}_k}$.

C. Probabilities Computation

The goal of this section is to show how to compute the probabilities associated with each transition of the PFA model, that is the output of the AD module.

Let $\mathcal{A}_k$ be the PFA of the activity $\mathcal{A}_k$; let $e_i, e_j \in \Sigma_{\mathcal{A}_k}$ be two events of the activity $\mathcal{A}_k$; let w_{p_k} be a sequence of events of $\Sigma_{\mathcal{A}_k}$ obtained by projection function of the sequence w on alphabet $\Sigma_{\mathcal{A}_k}$; let $q_l, q_m \in Q_{\mathcal{A}_k}$ be two actions performed in the activity $\mathcal{A}_k$. The probability to move from action q_l into action q_m by the event e_i is defined by

$$P(q_l, e_i, q_m) = P(q_l \to q_m | q_l) \times P(e_i | q_l \to q_m, q_l). \quad (4)$$

In words, the probability to move from q_l to q_m by event e_i is defined by the probability to move from q_l to q_m multiplied by the probability to generate $e_i$during this move.

Moreover, the probabilities of the (4) are computed by a standard approach: the likelihood of one or more events happening divided by the number of possible outcomes. More precisely, it holds

$$P(q_l \to q_m | q_l) = \frac{\tilde{N}(q_l \to q_m | q_l)}{\sum_{q_n \in Q_{\mathcal{A}_k}} \tilde{N}(q_l \to q_n | q_l)} \quad (5)$$

$$P(e_i | q_l \to q_m, q_l) = \frac{\tilde{N}(e_i | q_l \to q_m)}{\sum_{e_j \in \Sigma_{\mathcal{A}_k}} \tilde{N}(e_j | q_l \to q_m)} \quad (6)$$

where $\tilde{N}(q_l \to q_m | q_l)$ denotes the number of occurrences of transitions from q_l to q_m and $\tilde{N}(e_i | q_l \to q_m)$ the occurrence of event e_i conditioned to the transition $q_l \to q_m$ during the run period.

Unfortunately, the number of the performed actions in each activity is unknown. By consequence, we can only consider the sequence w_{p_k} to evaluate probabilities (5) and (6). Then, in order to compute $\tilde{N}(q_l \to q_m | q_l)$ and $\tilde{N}(e_i | q_l \to q_m)$, we use observable occurrences or successions of events by the following indicators.

1) $N^k_{e_i}$: number of times event e_i is observed in the sequences w_{p_k}.
2) $N^k_{\text{init}\, e_i}$: number of times event e_i is observed as first event in the sequences w_{p_k}.
3) $N^k_{e_i \to e_j}$: number of times event e_j follows event e_i in the sequences w_{p_k}.

Let C_{e_i} (resp. C_{e_j}) be the number of actions (states) having the event e_i (resp. e_j) as input, and let Σ_{q_l} (resp. Σ_{q_m}) be the set of events that are in input of state q_l (resp. q_m). Now, $\tilde{N}(q_l \to q_m | q_l)$ and $\tilde{N}(e_i | q_l \to q_m)$ are determined as follows:

$$\tilde{N}(q_l \to q_m | q_l) = \begin{cases} \sum\limits_{e_i \in \Sigma_{q_m}} \frac{1}{C_{e_i}} \times N^k_{\text{init}\, e_i}, & \text{if } q_l = q_0 \\ \sum\limits_{e_i, e_j \in \Sigma_{q_l}} N^k_{e_i \to e_j}, & \text{if } q_l = q_m \neq q_0 \\ \sum\limits_{e_i \in \Sigma_{q_l}; e_j \in \Sigma_{q_m}; e_j \notin \Sigma_{q_l}} \frac{1}{C_{e_j}} \times N^k_{e_i \to e_j}, & \text{if } q_m \neq q_l \neq q_0 \end{cases} \quad (7)$$

and

$$\tilde{N}(e_i | q_l \to q_m) = N^k_{e_i} \text{ if } e_i \in \Sigma_{q_m}; \quad 0 \text{ otherwise}. \quad (8)$$

In (7) and (8), we consider the fact that when a human starts a new action, the performed move is not dependent on the past action. In (7), the number of occurrences of transitions from q_l to q_m is determined considering three cases: 1) if $q_l = q_0$ we consider the number of times event e_i that is in input of q_m is observed as first event, averaged by the number of states having the event e_i as input; 2) if $q_l = q_m \neq q_0$ we count the number of times event e_j follows event e_i where e_j and e_i are in input of q_l; and 3) if $q_m \neq q_l \neq q_0$ then the count is similar to case 1) but for the computation of $N^k_{e_i \to e_j}$ that represents number of times e_j follows e_i in the considered sequence.

Moreover, in (8), we consider the occurrence of e_i independent of the starting state q_l but it depends only on the reached state q_m. Hence, the occurrence of event e_i conditioned to the transition $q_l \to q_m$ during the run period is equal to number of times e_i, which is in input of q_m, is observed during the run period.

D. Complexity of the AD Algorithm

The complexity of the three steps of the AD algorithm is the following:

$$C_{\text{AD}} = O\Big(\big[(|\text{LearningDB}| - |w|) \max_{\mathcal{A}_k}((\text{card}(Q_{\mathcal{A}_k})))^2 + \max_{\mathcal{A}_k}(\text{card}(\Sigma_{\mathcal{A}_k}))\big] \times \text{card}(A)\Big)$$

where

1) symbol *card* stands for cardinality of the set;
2) $|\text{LearningDB}| - |w|$ is the number of windows obtained by sliding;
3) $\max_{\mathcal{A}_k}(\text{card}(Q_{\mathcal{A}_k}))$ is the maximum number of actions linked to the same activity;
4) $\max_{\mathcal{A}_k}(\text{card}(\Sigma_{\mathcal{A}_k}))$ is the maximum number of events linked to the same activity.

Note that the complexity of the AD algorithm is polynomial and mainly depends on the maximum number of actions linked to the same activity. Since the number of actions is not large, the algorithm is easily computable.

V. Activity Recognition

A. Recognition Protocol

The AR consists in recognizing online (i.e., in reasonable time), which activity is actually performed by the inhabitant.

Recognition is done by computing an indicator evaluating the probability that a PFA generates an observed sequence w. The activity having the maximum probability of generating sequence w can be considered as being the activity currently performed. In [17], perplexity is presented as a useful indicator to compute distances between a language $\mathcal{L}(w)$ generated from the observed sequence w and a PFA $\mathcal{A}_k$.

Definition 2 (Likelihood): Let $\Theta_{\mathcal{A}_k}(w)$ be the set of paths for w in $\mathcal{A}_k$. The probability of generating w with $\mathcal{A}_k$ is the likelihood of w considering $\mathcal{A}_k$ and can be computed by

$$P_{\mathcal{A}_k}(w) = \sum_{\theta \in \Theta_{\mathcal{A}_k}(w)} P_{\mathcal{A}_k}(\theta). \tag{9}$$

Note that $P_{\mathcal{A}_k}(\theta)$ is the probability of the sequence of transitions $\theta = (q_l, e_i, q_m), (q_m, e_j, q_s), \ldots, (q_p, e_n, q_r)$, consistent with $w = e_i, e_j, \ldots, e_n$ in $\mathcal{A}_k$ and calculated as $P_{\mathcal{A}_k}(\theta) = P(q_l, e_i, q_m) P(q_m, e_j, q_s), \ldots, P(q_p, e_n, q_r)$, assuming the independence among transitions.

Two definitions of the perplexity are reported [17]: perplexity "per string" and perplexity "per symbols." Here, we consider the perplexity "per string."

Definition 3 (Perplexity): The perplexity "per string" is defined as the inverse of the geometric mean of the likelihood

$$\mathrm{PP}(\mathcal{L}(w)|\mathcal{A}_k) = \left[\prod_{v \in \mathcal{L}(w)} P_{\mathcal{A}_k}(v)\right]^{-\frac{1}{card(\mathcal{L}(w))}}. \tag{10}$$

The perplexity is based on the computation of likelihood $P_{\mathcal{A}_k}(v)$ for $v \in \mathcal{L}(w)$ and can be used to distinguish between two automatons $\mathcal{A}_1$ and $\mathcal{A}_2$ only if the same sequence v is used for the two models, i.e., if $v \in \Sigma^*_{\mathcal{A}_1}$ and $v \in \Sigma^*_{\mathcal{A}_2}$. However, the values of the perplexity are not significant if the alphabet of the sequence is not included in the alphabet of automaton.

In the considered problem two issues prevent applying the standard likelihood-based approaches. First, all the PFAs $\mathcal{A}_k$ do not share the same alphabet $\Sigma_{\mathcal{A}_k}$. Second, it is necessary to project the observed sequence on the PFA's alphabet in order to filter the events not belonging to the PFA alphabet.

In order to overcome these two issues, we propose an AR protocol that is composed by four steps.

1) The windowing of observed events considers a sequence w composed with a fix number of events.
2) For each PFA $\mathcal{A}_k$, the projection $w_{p_{\mathcal{A}_k}}$ of the considered sequence w is obtained.
3) A language $\mathcal{L}(w_{p_{\mathcal{A}_k}})$ is generated for each projected sequence $w_{p_{\mathcal{A}_k}}$ as described in Section III-B.
4) The probability for each model $\mathcal{A}_k$ to generate the language $\mathcal{L}(w_{p_{\mathcal{A}_k}})$ is computed by likelihood algorithm.

B. Normalized Likelihood and Perplexity

Note that methods based on the likelihood computation are not pertinent to the projected sequences since it can lead to compare likelihood or perplexity of sequences having different lengths. The risk is the shorter sequences will always more probable than longer ones as a consequence of the projection of the observed sequences. In order to be able to compare sequences having different lengths, we propose a normalization of the classical likelihood computation.

Definition 4: (The Normalized Likelihood): Let us consider the PFA $\mathcal{A}_k$ and a given sequence $w \in \Sigma^*_{\mathcal{A}_k}$. The normalized likelihood of sequence w in $\mathcal{A}_k$ is defined as

$$\|P_{\mathcal{A}_k}(w)\| = \frac{P_{\mathcal{A}_k}(w)}{\max_{v \in \Sigma^{|w|}_{\mathcal{A}_k}} [P_{\mathcal{A}_k}(v)]}. \tag{11}$$

In other words, for a given length of the sequences, the normalized likelihood normalizes the probability of a sequence with regards to the probability of the sequence having the highest probability to be generated by the PFA. This is to address the problem of the probability which decreases with the sequence size.

Analogously, the normalized perplexity is defined as follows.

Definition 5 (The Normalized Perplexity): Let us consider the PFA $\mathcal{A}_k$, a language $\mathcal{L}(w)$ generated from the observed sequence w and the normalized likelihood $\|P_{\mathcal{A}_k}(v)\|$ of $v \in \mathcal{L}(w)$. We define the normalized perplexity "per string" as the inverse of the geometric mean of the normalized likelihood

$$\|\mathrm{PP}(\mathcal{L}(w)|\mathcal{A}_k)\| = \left[\prod_{v \in \mathcal{L}(w)} \|P_{\mathcal{A}_k}(v)\|\right]^{-\frac{1}{\mathrm{card}(\mathcal{L}(w))}}. \tag{12}$$

Since the perplexity (resp. normalized perplexity) represents the distance between a language and the probability that it is generated by a model, we impose minimizing its value. Moreover, the inverse of the perplexity (resp. normalized perplexity) represents the probability, for the considered model, to generate the considered sequence and has to be maximized. In order to simplify the computation, maximizing the inverse of the normalized perplexity $(1/\|\mathrm{PP}(\mathcal{L}(w)|\mathcal{A}_k)\|)$ is preferred.

C. Normalized Likelihood Computation

The computation of the normalized likelihood has a high complexity. In particular, the complexity of the computation of $P_{\mathcal{A}_k}(w)$ is polynomial with $|w|$ thanks to the optimized forward algorithm [28]: $C_L = O(\mathrm{card}(Q_{\mathcal{A}_k})^2 \times |w|)$.

On the other hand, the complexity of $\max_{v \in \Sigma^{|w|}_{\mathcal{A}_k}} [P_{\mathcal{A}_k}(v)]$ is exponential with $|w|$ and polynomial with the event set cardinality

$$C_M = O(\mathrm{card}(\Sigma_{\mathcal{A}_k})^{|w|} \times \mathrm{card}(Q_{\mathcal{A}_k})^2 \times |w|). \tag{13}$$

However, the normalized likelihood computation can be decomposed in to two parts corresponding to two different steps of the computation.

1) $P_{\mathcal{A}_k}(w)$ can be computed online when the estimation is done.
2) $\max_{v\in\Sigma_{\mathcal{A}_k}^{|w|}}[P_{\mathcal{A}_k}(v)]$ can be computed offline for each $\mathcal{A}_k$ and for all possible values of $|w|$.

Even if the maximum likelihood is computed offline, the computational effort can be too high, and the complexity of this step has to be reduced.

Two complementary methods of complexity reduction are presented: model reduction and dynamic computation.

1) Complexity Reduction by Model Reduction: In order to reduce the computational complexity, a reduction of the model can be envisaged. Indeed, by keeping, for each couple of states, only the transitions with higher probabilities, it is possible to obtain an abstraction of the considered PFA having the same maximum of likelihood.

In the following, we define the reduced PFA and the rules to obtain it.

Definition 6 (The Reduced PFA): Let $\mathcal{A}_k = \langle Q_{\mathcal{A}_k}, \Sigma_{\mathcal{A}_k}, \delta_{\mathcal{A}_k}, I_{\mathcal{A}_k}, F_{\mathcal{A}_k}, P_{\mathcal{A}_k}\rangle$ be a PFA, we denote by $\mathcal{A}_k^r = \langle Q_{\mathcal{A}_k}, \Sigma_{\mathcal{A}_k}, \delta_{\mathcal{A}_k}, I_{\mathcal{A}_k}, F_{\mathcal{A}_k}, P_{\mathcal{A}_k}\rangle$ the reduced PFA associated with $\mathcal{A}_k$ where $\Sigma^r_{\mathcal{A}_k}, \delta^r_{\mathcal{A}_k}$, and $P^r_{\mathcal{A}_k}$ are obtained by the reduction procedure $\mathcal{A}_k \to \mathcal{A}_k^r$.

In order to specify the reduction procedure $\mathcal{A}_k \to \mathcal{A}_k^r$, the following definitions are necessary.

First, we denote by $\mathrm{Geq}_{q_l,q_m}(e_i)$ (resp. $\mathrm{Equ}_{q_l,q_m}(e_i)$) the set of events $e_j \in \Sigma_{\mathcal{A}_k}$ having probability $P(q_l, e_j, q_m)$ to occur from state q_l to state q_m greater than or equal to (resp. equal to) $e_i \in \Sigma_{\mathcal{A}_k}$ having a probability $P(q_l, e_i, q_m)$ to occur. More formally, it holds

$$\begin{aligned}&\mathrm{Geq}_{q_l,q_m}(e_i)\\&\quad = \{e_j | e_j \in \Sigma_{\mathcal{A}_k} \text{ and } P(q_l, e_j, q_m) \geq P(q_l, e_i, q_m)\} \quad (14)\\&\mathrm{Equ}_{q_l,q_m}(e_i)\\&\quad = \{e_j | e_j \in \Sigma_{\mathcal{A}_k} \text{ and } P(q_l, e_j, q_m) = P(q_l, e_i, q_m)\}. \quad (15)\end{aligned}$$

In the following, the reduction procedure $\mathcal{A}_k \to \mathcal{A}_k^r$ is presented.

Reduction procedure $\mathcal{A}_k \to \mathcal{A}_k^r$

Step 1 (Selection of Candidate Events): We first keep events having the same intersection sets of Geq and Equ are selected

$$\Sigma^r_{\mathcal{A}_k} = \left\{ e_j | \begin{array}{l} e_j \in \Sigma_{\mathcal{A}_k} \text{ and } \bigcap_{q_l,q_m\in Q^2_{\mathcal{A}_k}} \mathrm{Geq}_{q_l,q_m}(e_j) \\ = \bigcap_{q_l,q_m\in Q^2_{\mathcal{A}_k}} \mathrm{Equ}_{q_l,q_m}(e_j) \end{array} \right\}. \quad (16)$$

Step 2 (Deletion of Equ*ivalent Events):* For all the event sets $\bigcap_{q_l,q_m\in Q^2_{\mathcal{A}_k}} \mathrm{Equ}_{q_l,q_m}(e_j)$, only one event is kept: a new $\Sigma^r_{\mathcal{A}_k}$ is thus obtained.

Step 3 (Conservation of Transitions Linked to the Kept Events): Only transitions implying kept events are conserved, all the others are deleted. Probability of those transitions are not changed

$$\begin{aligned}\delta^{r_k}_{\mathcal{A}} &= \left\{(q_l, e_j, q_m) | (q_l, e_j, q_m) \in \delta_{\mathcal{A}_k} \text{ and } e_j \in \Sigma^{r_k}_{\mathcal{A}}\right\} \quad (17)\\ P^{r_k}_{\mathcal{A}} &= \left\{P(q_l, e_j, q_m) | (q_l, e_j, q_m)\right.\\ &\qquad \left.\in \delta^{r_k}_{\mathcal{A}} \text{ and } P(q_l, e_j, q_m) \in P_{\mathcal{A}_k}\right\}. \quad (18)\end{aligned}$$

The problem reduction leads to a new model with a lower number of events than the original one. Thus, the number of combinations to compare in the determination of the normalized perplexity is reduced. The following proposition proves that the likelihood maximum value is conserved after the reduction.

Proposition 1: Let $\mathcal{A}_k = \langle Q_{\mathcal{A}_k}, \Sigma_{\mathcal{A}_k}, \delta_{\mathcal{A}_k}, I_{\mathcal{A}_k}, F_{\mathcal{A}_k}, P_{\mathcal{A}_k}\rangle$ be a PFA and

$\mathcal{A}_k^r = \langle Q_{\mathcal{A}_k}, \Sigma_{\mathcal{A}_k}, \delta_{\mathcal{A}_k}, I_{\mathcal{A}_k}, F_{\mathcal{A}_k}, P_{\mathcal{A}_k}\rangle$ be the reduced PFA obtained by the reduction procedure $\mathcal{A}_k \to \mathcal{A}_k^r$. Then $\forall w \in \Sigma^*_{\mathcal{A}}$ of length $|w|$ it holds

$$\max_{v\in\Sigma^{|w|}_{\mathcal{A}^r_k}}[P_{\mathcal{A}^r_k}(u)] = \max_{v\in\Sigma^{|w|}_{\mathcal{A}_k}}[P_{\mathcal{A}_k}(v)]. \quad (19)$$

The proof is in Appendix.

By applying the reduction procedure and thanks to Proposition 1, the computational complexity of the normalized likelihood is reduced by substituting $\Sigma_{\mathcal{A}_k}$ with $\Sigma^r_{\mathcal{A}_k}$ in (13).

Now, the following proposition shows that the complexity can be further reduced by the reduction procedure $\mathcal{A}_k \to \mathcal{A}_k^r$.

Proposition 2: Let $\mathcal{A}_k = \langle Q_{\mathcal{A}_k}, \Sigma_{\mathcal{A}_k}, \delta_{\mathcal{A}_k}, I_{\mathcal{A}_k}, F_{\mathcal{A}_k}, P_{\mathcal{A}_k}\rangle$ be a PFA and $\mathcal{A}_k^r = \langle Q_{\mathcal{A}_k}, \Sigma_{\mathcal{A}_k}, \delta_{\mathcal{A}_k}, I_{\mathcal{A}_k}, F_{\mathcal{A}_k}, P_{\mathcal{A}_k}\rangle$ be the reduced PFA obtained by the reduction procedure $\mathcal{A}_k \to \mathcal{A}_k^r$. Then, the computational complexity of the normalized likelihood is the following:

$$C_M = O(2^{[\mathrm{card}(Q_{\mathcal{A}_k})-1]^{|w|}} \times \mathrm{card}(Q_{\mathcal{A}_k})^2 \times |w|). \quad (20)$$

The proof is in Appendix.

D. Complexity Reduction by Dynamic Computation

In addition to the model reduction, a computational simplification can be employed. In fact, it is necessary to find the $\max_{v\in\Sigma^{|w|}_{\mathcal{A}_k}}[P_{\mathcal{A}_k}(v)]$.

To this aim, the probabilities $P[v_2, q_{\mathrm{final}} = q_i]$ to generate a subsequence v_2 of the considered v to reach state $q_{\mathrm{final}} = q_i$ can be computed only one time for all sequences $v \in \Sigma^{|w|}_{\mathcal{A}_k}$ sharing v_2. This dynamic reduction removes the linear component $|w|$ in (20) as follows:

$$C_M = O(2^{[\mathrm{card}(Q_{\mathcal{A}_k})-1]^{|w|}} \times \mathrm{card}(Q_{\mathcal{A}_k})^2). \quad (21)$$

Remark 1: It is important to note that (21) represents the complexity of the AR algorithm that is reduced by the proposed model reduction and dynamic computation. Moreover, also in this case, the complexity is a function of the number of actions connected with an activity. However, since a complex activity may be decomposed in no more than ten/twelve actions and the sliding window length is selected by the observer, the proposed AR algorithm is suitable for real-time computation.

VI. Case Study

In this section, a case study is discussed by considering a real flat and some experiments performed by experts in order to show the application of the presented AD and AR approaches. To this aim, we describe in detail the considered flat, the choice and the location of the sensors, the semantic classification of the activities, and the used database. The experiments are implemented by considering the real life of the flat not elderly inhabitant.

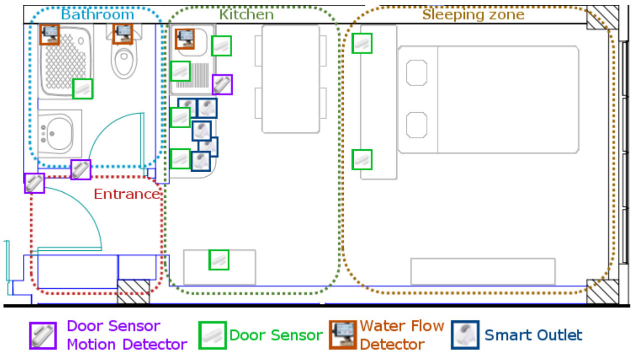

Fig. 5. Smart flat and the positioned binary sensors.

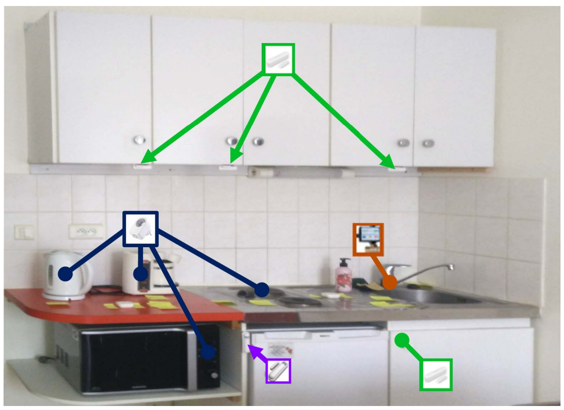

Fig. 6. Kitchen view of the smart flat.

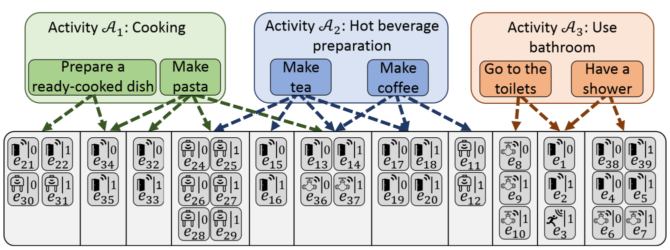

Fig. 7. Semantic decomposition of activities in actions and moves.

A. Smart Flat Description

The studied flat is provided by the ENS Paris-Saclay in France and is composed of two rooms that can be divided into four zones: entrance, bathroom, kitchen, and sleeping zone (see Figs. 5 and 6). The flat is equipped with 21 binary sensors positioned as shown in Fig. 6.

Each one is denoted by an explicit name and can generate two events: one linked to the rising edge and one linked to the falling edge (see Fig. 4). The list of the sensor events used in this case study is given in Table I.

B. Activities to be Monitored

In the case study, we consider three activities, six actions, and 42 observable moves (events). The activities analyzed for the ADL analysis are $\mathcal{A}_1$ = "Cooking," $\mathcal{A}_2$ = "Hot beverage preparation," and $\mathcal{A}_3$ = "Use bathroom" as Fig. 7 shows. Moreover, each activity is described by two states (actions) and each state is connected with a set of events (see Fig. 7).

Note that activities $\mathcal{A}_1$ and $\mathcal{A}_2$ share some sensors detecting the actions "Make pasta" and "Make tea": for instance,

TABLE I

LIST OF EVENTS LOGGED IN THE SMART FLAT

Id	Event name	Id	Event name
e_1	Bathroom|Door_PIR|Open|0	e_{22}	Kitchen|Fridge_PIR|Open|1
e_2	Bathroom|Door_PIR|Open|1	e_{23}	Kitchen|Fridge_PIR|Presence|1
e_3	Bathroom|Door_PIR|Presence|1	e_{24}	Kitchen|Hotplate_bottom|Power|0
e_4	Bathroom|Shower_Door|Open|0	e_{25}	Kitchen|Hotplate_bottom|Power|1
e_5	Bathroom|Shower_Door|Open|1	e_{26}	Kitchen|Hotplate_top|Power|0
e_6	Bathroom|Shower_Waterflow|Flow|0	e_{27}	Kitchen|Hotplate_top|Power|1
e_7	Bathroom|Shower_Waterflow|Flow|1	e_{28}	Kitchen|Kettle|Power|0
e_8	Bathroom|Sink_Waterflow|Flow|0	e_{29}	Kitchen|Kettle|Power|1
e_9	Bathroom|Sink_Waterflow|Flow|1	e_{30}	Kitchen|Microwave_oven|Power|0
e_{10}	Bathroom|Toilets_Waterflow|Flow|1	e_{31}	Kitchen|Microwave_oven|Power|1
e_{11}	Kitchen|Coffee_Machine|Power|0	e_{32}	Kitchen|Sideboard_Left|Open|0
e_{12}	Kitchen|Coffee_Machine|Power|1	e_{33}	Kitchen|Sideboard_Left|Open|1
e_{13}	Kitchen|Cupboard_Bottom|Open|0	e_{34}	Kitchen|Sideboard_Right|Open|0
e_{14}	Kitchen|Cupboard_Bottom|Open|1	e_{35}	Kitchen|Sideboard_Right|Open|1
e_{15}	Kitchen|Cupboard_CenterLeft|Open|0	e_{36}	Kitchen|Sink_Waterflow|Flow|0
e_{16}	Kitchen|Cupboard_CenterLeft|Open|1	e_{37}	Kitchen|Sink_Waterflow|Flow|1
e_{17}	Kitchen|Cupboard_CenterRight|Open|0	e_{38}	Kitchen|Wardrobe|Open|0
e_{18}	Kitchen|Cupboard_CenterRight|Open|1	e_{39}	Kitchen|Wardrobe|Open|1
e_{19}	Kitchen|Cupboard_Left|Open|0	e_{40}	Entrance|Door_PIR|Open|0
e_{20}	Kitchen|Cupboard_Left|Open|1	e_{41}	Entrance|Door_PIR|Open|1
e_{21}	Kitchen|Fridge_PIR|Open|0	e_{42}	Entrance|Door_PIR|Presence|1

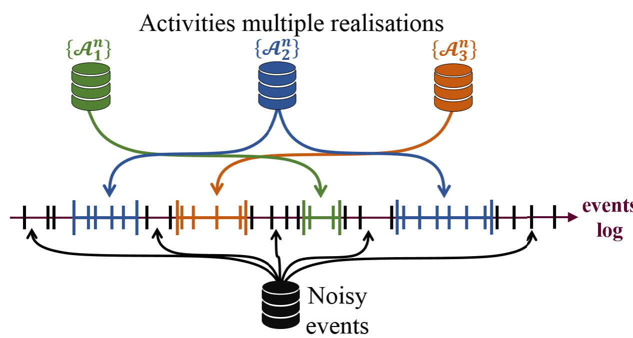

Fig. 8. Structure of the test sequence of observed events.

the events "boil water" and "using hotplates" are connected with both actions. However, since activity $\mathcal{A}_3$ is carried out in a different area of the flat (i.e., the bathroom), the events linked to $\mathcal{A}_3$ are fully independent of other activities. The list of events connected with moves and sensors is given in Table I.

In order to estimate the robustness of the approach, activities $\mathcal{A}_k$ with $k = 1, 2, 3$ are observed a large number of times by introducing the following variations.

1) The insertion of noisy events (i.e., events not included in $\Sigma_{\mathcal{A}_k}$) during their realization, for instance by wandering in the flat,
2) Some actions are interrupted.
3) The execution order of elementary moves composing actions is changed,
4) The action "make tea" is realized by two different ways: using the kettle or boiling water with hotplates.

The test database is generated using recorded activity instances placed in a random time order and separated by a random number of random noisy events not belonging to the performed activities. The resulting sequence is composed of 2087 events corresponding to 20 realizations of each activity (see Fig. 8).

C. Activity Discovery

In this section, the presented AD method is applied to the case study and in particular to $\mathcal{A}_1$ = "Cooking." The learning database is composed of 20 occurrences of each activity.

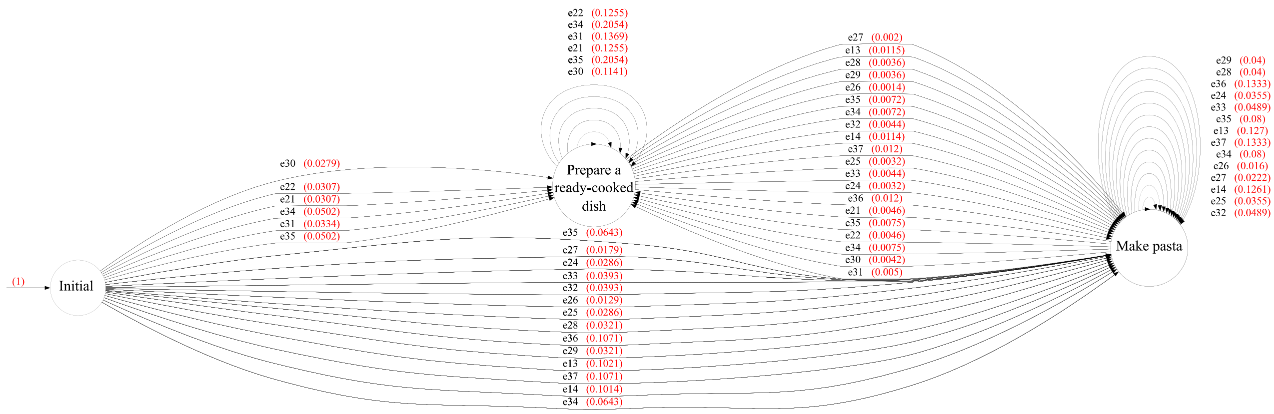

Fig. 9. Digraph D($\mathbf{A_1}$): Cooking modeled by a PFA. In black: the structure resulting from AD: step 1, in red: probabilities computed in AD: step 3.

Step 1: Generation of PFA Structure

From the semantic decomposition of Fig. 7, the model structure is generated and shown by the black digraph $D(\mathcal{A}_1)$ of Fig. 9.

Step 2: Analysis and Process of Streaming Data

In the considered case study, a length of five events is considered for the sliding window.

Indicators presented in Section IV-B are computed for each activity by considering the projected sequence of each sliding window. For instance, in the considered database, some indicators involving sensors e_{14} and e_{35} for activity $\mathcal{A}_1$ are the following:

$$\begin{aligned}
N^1_{e_{14}} &= 142, \quad N_{e_{35}} = 90\\
N^1_{\text{init}\, e_{14}} &= 112, \quad N_{\text{init}\, e_{35}} = 29\\
N_{e_{14}\to e_{13}} &= 96, \quad N_{e_{14}\to e_{26}} = 4, \quad N_{e_{14}\to e_{27}} = 4, \quad N_{e_{14}\to e_{24}} = 4\\
N_{e_{32}\to e_{14}} &= 3, \quad N_{e_{34}\to e_{14}} = 10, \quad N_{e_{28}\to e_{14}} = 5, \quad N_{e_{27}\to e_{14}} = 4\\
N_{e_{29}\to e_{14}} &= 4, \quad N_{e_{24}\to e_{14}} = 4, \quad N_{e_{35}\to e_{34}} = 66.
\end{aligned}$$

Step 3: Probabilities Computation

In this step, the PFA probabilities are computed according to (4) to (8). At the end of this step, ADLs are fully modeled and the PFA obtained for activity $\mathcal{A}_1$ = "Cooking" is depicted in Fig. 9.

Finally, the described three steps are applied for each activity $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$ but all the resulting models are not presented for the sake of brevity.

D. Activity Recognition

In this section, the proposed AR approach is applied to the case study. For the sake of brevity, only the results obtained during one realization of the activity "Cooking" are shown.

The observed sequence is the following: $\ldots e_{22}e_{21}e_{31}e_{23}e_{35}e_{34}e_{30}e_{23}\ldots$.

The described steps of Section V-A are applied to the case study.

1) If a window of length 5 is considered, then the sequences can be the following:

$$\begin{cases} w_1 = e_{22}e_{21}e_{31}e_{23}e_{35}\\ w_2 = e_{21}e_{31}e_{23}e_{35}e_{34}\\ \quad\ldots \end{cases}$$

2) For each PFA $\mathcal{A}_k$ with $k = 1, 2, 3$ modeling the activities, a projection of sequences w_i for $i = 1, 2$ is obtained and the projected sequence is denoted by $w_{p_{\mathcal{A}_k}}$. For example, the obtained projected sequences of w_2 are

$$\begin{cases} w_{p_{\mathcal{A}_1}} = e_{21}e_{31}e_{35}e_{34}\\ w_{p_{\mathcal{A}_2}} = \emptyset\\ w_{p_{\mathcal{A}_3}} = \emptyset. \end{cases} \tag{22}$$

3) A language $\mathcal{L}(w_{p_{\mathcal{A}_k}})$ is generated for each projected sequence $w_{p_{\mathcal{A}_k}}$ with $k = 1, 2, 3$. Sequences (22) give

$$\begin{cases} \mathcal{L}(w_{p_{\mathcal{A}_1}}) = \{e_{21}e_{31}, e_{31}e_{35}, e_{35}e_{34}, e_{21}e_{31}e_{35}\\ e_{31}e_{35}e_{34}, e_{21}e_{31}e_{35}e_{34}\}\\ \mathcal{L}(w_{p_{\mathcal{A}_2}}) = \emptyset\\ \mathcal{L}(w_{p_{\mathcal{A}_3}}) = \emptyset. \end{cases} \tag{23}$$

4) The inverse of the normalized perplexity for each model $\mathcal{A}_k$ to generate the language $\mathcal{L}(w_{p_{\mathcal{A}_k}})$ is computed. Fig. 10 shows the evolution of this value during the realization of the "Cooking" activity. Here, in order to validate the method, the log of actually performed activities is compared with the computed estimations. We enlighten that this log is for the validation procedure only and it is not required by the proposed method. The knowledge of performed activity is drawn with plain lines. The probability is equal to 1 when the activity is actually performed. The value of the presented estimator is drawn by the crossed lines.

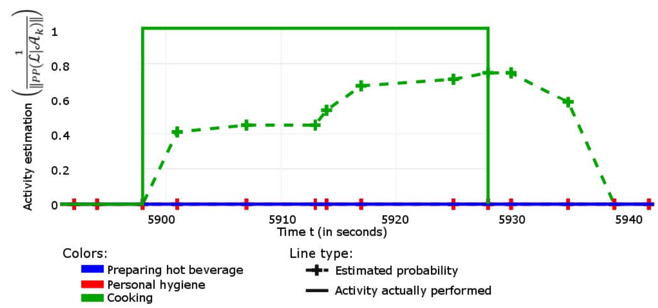

Fig. 10. Inverse of the normalized perplexity during the realization of the activity "Cooking" using no shared events.

The example shows that for each new observed event, the estimation of probability is actualized. The language is empty if the projected sequence length is lower than 2 and

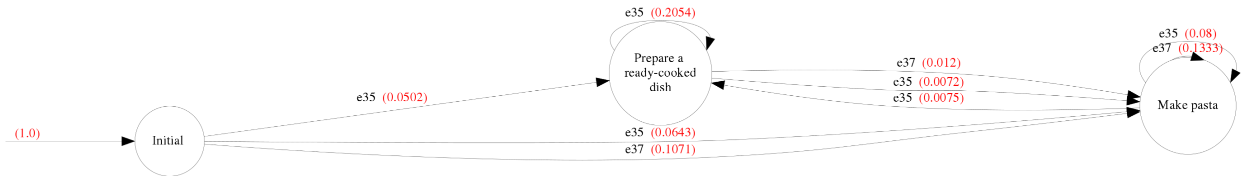

Fig. 11. Activity $\mathbf{A_1}$: **Cooking** reduced model for AR offline maximum likelihood computation.

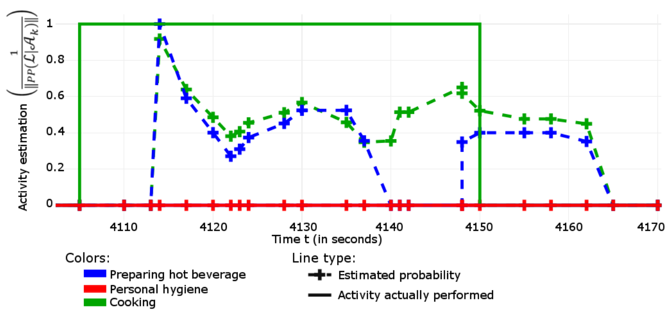

Fig. 12. Inverse of the normalized perplexity during the realization of the activity "Cooking" based on shared moves.

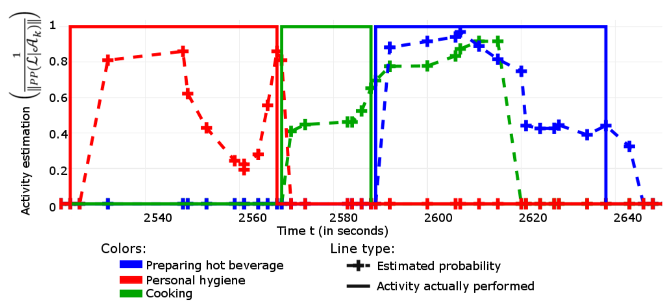

Fig. 13. Inverse of the normalized perplexity during the succession of three activities: "Personal Hygiene"; "Cooking," and "Preparing Hot Beverage."

an offset is systematically observed when the activity starts. Furthermore, another offset is present when an activity stops.

This second offset is due to the use of a sliding window storing the last five observed events.

Finally, Fig. 11 shows the reduced PFA for activity $\mathcal{A}_1$. By applying the two complexity reduction strategies of the offline computation, by a window of length $|w| = 5$, we use $2^5 \times 2^2 = 128$ operations instead of $20^5 \times 2^2 \times 5 = 64.000.000$.

E. Case Study Discussion

The presented methods to model and recognize ADLs are efficient also to activities sharing events. In fact, Fig. 12 shows a case of shared events between the two activities "Cooking" and "Preparing Hot Beverage." As expected, it is not possible to distinguish which activity is performed if a move that belongs to only one of the two activities does not occur. Hence, if two activities have many events in common, it is impossible to recognize the activity.

On the other hand, if a sensor is linked with too many activities, it is observed a lot of times during the learning period: it becomes predominant in all linked ADL models making it nondiscriminant and noisy. Therefore, the events detected by such sensors are not useful.

In order to evaluate the efficiency of the proposed method, activity sequences are considered, and the results are shown in Fig. 13. Once again, we can observe indecision during the transition between two activities sharing the same events. Since the presented estimator (projection and normalized perplexity) allows finding the performed activity, it is possible to conclude that the presented method allows discovering and recognizing activities performed by an inhabitant of a smart home without declaring the performed activity during the learning period.

VII. Conclusion

In this article, a global approach for ADL Discovery and Recognition is proposed. To this aim, a procedure to automatically obtain the activities model in a PFA framework is developed on the basis of the knowledge of training event logs database and a hierarchical decomposition of activities to monitor actions and moves. Then, an AR method is presented on the basis of a newly defined distance estimator called normalized likelihood and the extension to perplexity. In order to face the complexity of the AR algorithm, we propose a complexity reduction method. Moreover, we prove that the normalized likelihood can be efficiently computed without loss of accuracy.

Finally, the proposed approach for AD and AR is applied to a real living lab and the good quality of the obtained results is discussed.

Future works will be dedicated to the quantization of how valuable each sensor is in enabling the AR and on the detection of the drifts in the activity realization. Indeed, such drifts are often indicators of the evolution of numerous pathologies and the related detection will notably increase the performance of health at home technologies.

Appendix

In order to prove Propositions 1 and 2, the following notation is defined.

1) $e_i \in \Sigma_{\mathcal{A}_k}$.
2) $e_{\text{eq}} \in \Sigma_{\mathcal{A}_k}$ such as $e_{\text{eq}} \in \bigcap_{q_l, q_m \in Q^2_{\mathcal{A}_k}} \text{Equ}_{q_l,q_m}(e_i)$.
3) $\{e_{\text{eq}}\}_{e_i} = \bigcap_{q_l, q_m \in Q^2_{\mathcal{A}_k}} \text{Equ}_{q_l,q_m}(e_i)$.
4) $e_{\text{sup}} \in \Sigma_{\mathcal{A}_k}$ such as $e_{\text{sup}} \in \bigcap_{q_l, q_m \in Q^2_{\mathcal{A}_k}} \text{Geq}_{q_l,q_m}(e_i)$ and $e_{\text{sup}} \notin \bigcap_{q_l, q_m \in Q^2_{\mathcal{A}_k}} \text{Equ}_{q_l,q_m}(e_i)$.
5) $\{e_{\text{sup}}\}_{e_i}$ is the set of all possible e_{sup} associated with e_i.

Thus, by definition it holds

$$\{e_{\text{eq}}\}_{e_i} + \{e_{\text{sup}}\}_{e_i} = \bigcap_{q_l, q_m \in Q^2_{\mathcal{A}_k}} \text{Geq}_{q_l,q_m}(e_i).$$

Proof of Proposition 1: In (16), only event e_i with $\{e_{\sup}\}_{e_i} = \emptyset$ are kept in the reduced PFA $\mathcal{A}_k^r$. We prove that the rejection of events having $\{e_{\sup}\}_{e_i} \neq \emptyset$ does not change the value of the maximum likelihood.

Let $w = w'_1 \dots w'_k \dots e_i \dots w'_{|w|-1} w'_{|w|}$ be a sequence of events and e_i is one of the events in the sequence.

Let $v = w'_1 \dots w'_k \dots e_{\sup} \dots w'_{|w|-1} w'_{|w|}$ be a sequence of events that equals sequence w but for event e_i, which is changed by $e_{\sup} \in \{e_{\sup}\}_{e_i}$.

At this point, two cases exist $\forall q_l, q_m \in Q^2_{\mathcal{A}_k}$

$$\begin{cases} P_{\mathcal{A}_k}[(q_l, e_{\sup}, q_m)] = P_{\mathcal{A}_k}[(q_l, e_i, q_m)] \\ \qquad\qquad\qquad\qquad \text{if } e_{\sup} \in \text{Equ}_{q_l,q_m}(e_i) \\ P_{\mathcal{A}_k}[(q_l, e_{\sup}, q_m)] > P_{\mathcal{A}_k}[(q_l, e_i, q_m)] \\ \qquad\qquad\qquad\qquad \text{if } e_{\sup} \notin \text{Equ}_{q_l,q_m}(e_i). \end{cases}$$

Thus, for each path $\theta = (s_0, w'_1, s_1 \dots s_{j-1}, e_i, s_j \dots w'_{|w|}, s_{|w|})$ generating w, it exists a path $\theta' = (s_0, w'_1, s_1 \dots s_{j-1}, e_{\sup}, s_j \dots w'_{|w|}, s_{|w|})$ such as if $e_{\sup} \in \text{Equ}_{s_{j-1},s_j}(e_i)$ then

$$\begin{aligned} I(s_0) &\times \left(\prod_{h=1}^{|v|} P(s_{h-1}, v'_h, s_h) \right) \\ &= I(s_0) \times \left(\prod_{h=1}^{|w|} P(s_{h-1}, w'_h, s_h) \right) \\ &\to P_{\mathcal{A}_k}(\theta') = P_{\mathcal{A}_k}(\theta) \end{aligned}$$

else if $e_{\sup} \notin \text{Equ}_{s_{j-1},s_j}(e_i)$ then

$$\begin{aligned} I(s_0) &\times \left(\prod_{h=1}^{|v|} P(s_{h-1}, v'_h, s_h) \right) \\ &> I(s_0) \times \left(\prod_{h=1}^{|w|} P(s_{h-1}, w'_h, s_h) \right) \\ &\to P_{\mathcal{A}_k}(\theta') > P_{\mathcal{A}_k}(\theta). \end{aligned}$$

Since $e_{\sup} \notin \bigcap_{q_l,q_m \in Q^2_{\mathcal{A}_k}} \text{Equ}_{q_l,q_m}(e_i)$, the case $e_{\sup} \notin \text{Equ}_{s_{j-1},s_j}(e_i)$ occurs at least one time, thus:

$$P_{\mathcal{A}_k}(v) = \sum_{\theta' \in \Theta_{\mathcal{A}_k}(v)} P_{\mathcal{A}_k}(\theta') > P_{\mathcal{A}_k}(w) = \sum_{\theta \in \Theta_{\mathcal{A}_k}(w)} P_{\mathcal{A}_k}(\theta). \tag{24}$$

Hence, (24) proves that, for all sequences w including an event e_i with $\{e_{\sup}\}_{e_i} \neq \emptyset$, it exists another sequence v having the same length with a greater likelihood. Therefore, event e_i with $\{e_{\sup}\}_{e_i} \neq \emptyset$ can be excluded for the maximum likelihood computation.

In the same way, we can prove that the likelihood does not change by changing an event e_i by another event $e_{\text{eq}} \in \{e_{\text{eq}}\}_{e_i}$, then only one of them can be kept in A_k^r. Consequently, it holds $\max_{u \in \Sigma_{\mathcal{A}_k^r}} [P_{\mathcal{A}_k^r}(u)] = \max_{u \in \Sigma_{\mathcal{A}_k^r}} [P_{\mathcal{A}_k}(v)]$ and the thesis is proven.

Proof of Proposition 2: In order to prove Proposition 2, we recall that the following properties are direct consequences of (16).

Property 1: If event e_i is kept using (16), it exists a set of n_1 origin and destination states $C^{n_1}_{\mathcal{A}_k} = \{(q_{l_1}, q_{m_1}) \dots (q_{l_p}, q_{m_p}) \dots (q_{l_n}, q_{m_{n_1}})\}$ such that $\forall e_j \in \Sigma_{\mathcal{A}_k}, \forall p \in [1, n_1]$ it holds:

$$P[(q_{l_p}, e_i, q_{m_p})] \geq P[(q_{l_p}, e_j, q_{m_p})].$$

Furthermore, according to (8), $\tilde{N}(e_i | q_l \to q_m)$ and $P_{\mathcal{A}_k}[(q_l, e_i, q_m)]$ do not depend on q_l. Thus, $\text{Geq}_{q_l,q_m}(e_i)$ and $\text{Equ}_{q_l,q_m}(e_i)$ depend only on e_i and q_m. It is possible to rewrite Property 1 as follows.

Property 2: If event e_i is kept using (16), it exists a set of n_2 destination states $D^{n_2}_{\mathcal{A}_k} = \{q_{m_1} \dots q_{m_p} \dots q_{m_{n_2}}\}$ such that $\forall e_j \in \Sigma_{\mathcal{A}_k}, \forall q_l \in Q_{\mathcal{A}_k}, \forall p \in [1, n_2]$ it holds

$$P[(q_l, e_i, q_{m_p})] \geq P[(q_l, e_j, q_{m_p})].$$

Moreover, for each possible set $D^{n_2}_{\mathcal{A}_k}$, of destination states, only one event is kept by the equivalent events deletion performed by step 2 of the reduction procedure. Thus, the number of kept events $N_{\mathcal{A}_k} = \text{Card}(\Sigma^r_{\mathcal{A}_k})$ is bounded by the number of possible sets $D^{n_2}_{\mathcal{A}_k}$ that it is necessary to evaluate.

For a PFA with $m = \text{card}(Q_{\mathcal{A}_k})$ states, sets composed with $n_2 \in [1, m-1]$ destination states can be created. For each n_2, it exists $(m - 1/n)$ different possible sets $D^{n_2}_{\mathcal{A}_k}$. Thus, we have

$$N_{\mathcal{A}_k} \leq \sum_{i=1}^{m-1} \binom{m-1}{i} = 2^{m-1} - 1.$$

Thus, according to (17) the complexity of the normalized likelihood is the following:

$$C_M = O(2^{[\text{card}(Q_{\mathcal{A}_k})-1]^{|w|}} \times \text{card}(Q_{\mathcal{A}_k})^2 \times |w|). \tag{25}$$

This proves Proposition 2. ■

REFERENCES

[1] *Population Structure and Ageing*, EU Eurostat, Brussels, Belgium, 2010.

[2] D. Pal, S. Funilkul, N. Charoenkitkarn, and P. Kanthamanon, “Internet-of-Things and smart homes for elderly healthcare: An end user perspective,” *IEEE Access*, vol. 6, pp. 10483–10496, 2018.

[3] D. Bhatt and S. Kumar, “Internet of Things: Smart device for smart city,” *Int. J. Comput. Appl.*, vol. 181, no. 7, pp. 1–4, 2018.

[4] M. J. Deen, “Information and communications technologies for elderly ubiquitous healthcare in a smart home,” *Pers. Ubiquitous Comput.*, vol. 19, nos. 3–4, pp. 573–599, Jul. 2015.

[5] J. van Hoof, G. Demiris, and E. Wouters, Eds., *Handbook of Smart Homes, Health Care and Well-Being*. Basel, Switzerland: Springer, 2017.

[6] D. J. Cook, N. C. Krishnan, and P. Rashidi, “Activity discovery and activity recognition: A new partnership,” *IEEE Trans. Cybern.*, vol. 43, no. 3, pp. 820–828, Jun. 2013.

[7] J. Saives, C. Pianon, and G. Faraut, “Activity discovery and detection of behavioral deviations of an inhabitant from binary sensors,” *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 4, pp. 1211–1224, Oct. 2015.

[8] K. Viard, M. P. Fanti, G. Faraut, and J.-J. Lesage, “An event-based approach for discovering activities of daily living by hidden Markov models,” in *Proc. 15th Int. Conf. Ubiquitous Comput. Commun. Int. Symp. Cyberspace Secur. (IUCC-CSS)*, Granada, Spain, Dec. 2016.

[9] E. Kim, S. Helal, and D. Cook, “Human activity recognition and pattern discovery,” *IEEE Pervasive Comput.*, vol. 9, no. 1, pp. 48–53, Jan. 2010.

[10] K. Viard, M. P. Fanti, G. Faraut, and J.-J. Lesage, “Recognition of human activity based on probabilistic finite-state automata,” in *Proc. 22nd IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Limassol, Cyprus, Sep. 2017, pp. 1–7.

[11] L. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[12] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, “Accurate activity recognition in a home setting,” in *Proc. 10th Int. Conf. Ubiquitous Comput. (UbiComp)*, 2008, pp. 1–9.

[13] V. Kellokumpu, M. Pietikäinen, and J. Heikkilä, “Human activity recognition using sequences of postures,” in *Proc. MVA IAPR Conf. Mach. Vis. Appl.*, 2005, pp. 570–573.

[14] S. Hongeng, F. Bremond, and R. Nevatia, “Representation and optimal recognition of human activities,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Hilton Head Island, SC, USA, vol. 1, Jun. 2000, pp. 818–825.

[15] S. Hongeng, R. Nevatia, and F. Bremond, “Video-based event recognition: Activity representation and probabilistic recognition methods,” *Comput. Vis. Image Understand.*, vol. 96, no. 2, pp. 129–162, Nov. 2004.

[16] T. Duong, D. Phung, H. Bui, and S. Venkatesh, “Efficient duration and hierarchical modeling for human activity recognition,” *Artif. Intell.*, vol. 173, nos. 7–8, pp. 830–856, May 2009.

[17] E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. C. Carrasco, “Probabilistic finite-state machines—Part II,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 7, pp. 1026–1039, Jul. 2005.

[18] A. A. Chaaraoui, P. Climent-Pérez, and F. Flórez-Revuelta, “A review on vision techniques applied to human behaviour analysis for ambient-assisted living,” *Expert Syst. Appl.*, vol. 39, no. 12, pp. 10873–10888, Sep. 2012.

[19] S. Gaglio, G. L. Re, and M. Morana, “Human activity recognition process using 3-D posture data,” *IEEE Trans. Human-Mach. Syst.*, vol. 45, no. 5, pp. 586–597, Oct. 2015.

[20] C.-H. Lu, Y.-C. Ho, Y.-H. Chen, and L.-C. Fu, “Hybrid user-assisted incremental model adaptation for activity recognition in a dynamic smart-home environment,” *IEEE Trans. Human-Mach. Syst.*, vol. 43, no. 5, pp. 421–436, Sep. 2013.

[21] S. Himmel, M. Ziefle, and K. Arning, *From Living Space to Urban Quarter: Acceptance of ICT Monitoring Solutions in an Ageing Society*. Berlin, Germany: Springer, 2013, pp. 49–58.

[22] R. Agrawal and R. Srikant, “Mining sequential patterns,” in *Proc. 11th Int. Conf. Data Eng.*, Mar. 1995, pp. 3–14.

[23] E. M. Tapia, S. S. Intille, and K. Larson, *Activity Recognition in the Home Using Simple and Ubiquitous Sensors*, vol. 3001. Cham, Switzerland: Springer, 2004, ch. 10.

[24] D. J. Cook and N. C. Krishnan, *Activity Learning: Discovering, Recognizing, and Predicting Human Behavior From Sensor Data*. Hoboken, NJ, USA: Wiley, 2015.

[25] A. Fleury, M. Vacher, and N. Noury, “SVM-based multimodal classification of activities of daily living in health smart homes: Sensors, algorithms, and first experimental results,” *IEEE Trans. Inf. Technol. Biomed.*, vol. 14, no. 2, pp. 274–283, Mar. 2010.

[26] N. C. Krishnan and D. J. Cook, “Activity recognition on streaming sensor data,” *Pervas. Mobile Comput.*, vol. 10, pp. 138–154, Feb. 2014.

[27] A. Galata, N. Johnson, and D. Hogg, “Learning variable-length Markov models of behavior,” *Comput. Vis. Image Understand.*, vol. 81, no. 3, pp. 398–413, Mar. 2001.

[28] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, vol. 3. London, U.K.: Pearson, 2014.

[29] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. New York, NY, USA: Springer, 2008.

[30] K. Mukherjee and A. Ray, “State splitting and merging in probabilistic finite state automata for signal representation and analysis,” *Signal Process.*, vol. 104, pp. 105–119, Nov. 2014.

[31] P. Adenis, K. Mukherjee, and A. Ray, “State splitting and state merging in probabilistic finite state automata,” in *Proc. Amer. Control Conf.*, Jun. 2011, pp. 5145–5150.

Kévin Viard received the M.S. degree from the Ecole Normale supérieure Paris-Saclay, Cachan, France, in 2015, and the joint Ph.D. degree from the Ecole Normale supérieure Paris-Saclay and Politecnico di Bari, Bari, Italy, in 2018.

Since September 2018, he has been teaching engineering sciences to French higher school preparatory classes. His research interests include smart environments, ambient assisted living, activity discovery and activity recognition, and other topics related to automated systems.

Maria Pia Fanti (Fellow, IEEE) received the Laurea degree in electronic engineering from the University of Pisa, Pisa, Italy, in 1983.

She was a Visiting Researcher with the Rensselaer Polytechnic Institute, Troy, NY, USA, in 1999. Since 1983, she has been with the Department of Electrical and Information Engineering, Polytechnic of Bari, Bari, Italy, where she is currently a Full Professor of system and control engineering and the Chair of the Laboratory of Automation and Control. Her research interests include management and modeling of complex systems, such as transportation and logistics systems, discrete-event systems, Petri nets, consensus protocols, and fault detection. She has published more than 290 articles and 2 textbooks on these topics.

Prof. Fanti was a member-at-large of the Board of Governors of the IEEE Systems, Man, and Cybernetics Society. She is also a member of the AdCom of the IEEE Robotics and Automation Society. She was the General Chair of the 2011 IEEE Conference on Automation Science and Engineering, the 2017 IEEE International Conference on Service Operations and Logistics, and Informatics, and the 2019 Systems, Man, and Cybernetics Conference. She is also the Chair of the Technical Committee on Automation in Logistics of the IEEE Robotics and Automation Society. She was an Editor of the IEEE TRANSACTIONS ON AUTOMATION SCIENCES AND ENGINEERING. She is also an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS.

Gregory Faraut (Member, IEEE) received the B.S. degree in electrical engineering and the M.S. degree in computer science from the University of Nice Sophia Antipolis, Nice, France, in 2004 and 2006, respectively, and the Ph.D. degree in automatic control from the Ampere Laboratory, INSA Lyon, Villeurbanne, France, in 2010.

Since 2011, he has been an Associate Professor of automatic control with LURPA, ENS Paris-Saclay, Gif-Sur-Yvette, France. His research interests concern the field of discrete event systems with applications to manufacturing systems and production systems. More specifically, he is also interested in identification techniques and ambient assisted living (AAL) approaches based on machine learning. The most recent work concerns, for AAL, the discovery, the detection and the evaluation of behavior's deviations of frailty people, and, in the area of manufacturing and production systems, the improvement of techniques for the use of digital twin.

Dr. Faraut is a Founding Co-Chair of the Technical Committee on Automation in Health Care Management of the IEEE Robotics and Automation Society.

Jean-Jacques Lesage (Member, IEEE) received the Ph.D. degree from the Ecole Centrale de Paris, Paris, France, in 1989, and the Habilitation à Diriger des Recherches degree from the University of Nancy, Nancy, France, in 1994.

He is currently a Professor of automatic control with the Ecole Normale Supérieure Paris-Saclay, Cachan, France. His research interests include formal methods and models for identification, analysis, and diagnosis of discrete event systems, as well as applications to manufacturing systems, network automated systems, energy production, and more recently ambient assisted living.