# The ContextAct@A4H real-life dataset of daily-living activities Activity recognition using model checking

Paula Lago, Frederic Lang, Claudia Roncancio, Claudia Jiménez-Guarín, Radu Mateescu, Nicolas Bonnefond

# The ContextAct@A4H real-life dataset of daily-living activities
## Activity recognition using model checking

Paula Lago[1], Fréderic Lang[2,3], Claudia Roncancio[2], Claudia Jiménez-Guarín[1], Radu Mateescu[2,3], and Nicolas Bonnefond[3]

[1] Systems and Computing Engineering Dept, Universidad de los Andes, Colombia
[2] Univ. Grenoble Alpes, CNRS, Grenoble, France
[3] INRIA, Grenoble, France
{pa.lago52,cjimenez}@uniandes.edu.co, {Claudia.Roncancio}@imag.fr, {Frederic.Lang,Radu.Mateescu,Nicolas.Bonnefond}@inria.fr

**Abstract.** Research on context management and activity recognition in smart environments is essential in the development of innovative well adapted services. This paper presents two main contributions. First, we present *ContextAct@A4H*, a new real-life dataset of daily living activities with rich context data[4]. It is a high quality dataset collected in a smart apartment with a dense but non intrusive sensor infrastructure. Second, we present the experience of using temporal logic and model checking for activity recognition. Temporal logic allows specifying activities as complex events of object usage which can be described at different granularity. It also expresses temporal ordering between events thus palliating a limitation of ontology based activity recognition. The results on using the CADP toolbox for activity recognition in the real life collected data are very good.

**Keywords:** Smart home, context, activity recognition, temporal logic

## 1 Introduction

Activity recognition in smart environments is a necessary step for the development of innovative services. In the health-care domain, recognizing activities of daily living (ADL) enables health status monitoring, functional assessments and smart assistance. Despite the broad efforts made in recent years to improve activity recognition, ADL recognition (e.g. cooking, washing dishes, showering) is still not commercially available as is physical activity recognition (e.g. running, cycling). ADL recognition is complex and some barriers have hindered its passage from laboratory settings to real-life. One of these barriers is the difficulty of evaluating methods in real-life scenarios, which differ widely from the scripted scenarios that are usually used for testing activity recognition algorithms. Real-life scenarios are inherently imbalanced, not only because some activities are

[4] This research is supported by the Amiqual4Home Innovation Factory, http://amiqual4home.inria.fr funded by the ANR (ANR-11-EQPX-0002)

more frequent than others but also because of the different durations of the activities. Learning from imbalanced data poses new challenges and can impact the performance of traditional algorithms [9]. Moreover, in real-life there are often several ongoing activities, making it challenging to recognize the "interesting" ones [12]. More real-life datasets and shared activity labels are needed to allow researchers evaluating their proposals and comparing the results of their recognition methods [2].

Activity recognition methods are broadly classified into data-based and knowledge-based [3]. Data based methods use machine learning or data mining algorithms to learn a model of activities. However, supervised methods require labeled data which is often hard to acquire. On the other hand, knowledge based methods specify the semantics of activities to be inferred from sensor measurements. Ontologies that specify both objects used and other characteristics of an activity (e.g. actor, location, duration) is one of the most used knowledge-based method for their ability to specify concepts at different granularity and their reasoning capabilities on uncertain knowledge. Nevertheless, ontologies lack support for the temporal characterization of activities [19] which is an important aspect of activity recognition as they take place during a span of time. For ADL recognition it is important to consider context information such as environmental conditions, visitor presence, current location and part of day [14]. Nonetheless, the usefulness of different context variables has not been proven in real-life scenarios due to the lack of available data in the same dataset.

In this paper we present two main contributions. First, we present a real-life dataset of daily living activities. This new dataset, named *ContextAct@A4H*, will be publicly available with reference to this paper. *ContextAct@A4H* includes sensor raw data reflecting the context as well as standard daily living annotated activities. Context information can help study how contextual changes affect user behavior and preferences [14]. To the best of our knowledge it will be the first public available dataset including such rich real-life data. *ContextAct@A4H* was gathered at the Amiqual4Home flat using a large variety of sensors to improve the potential reuse of the data. Researchers can choose the desired "configuration" according to their evaluation requirements. We highlight the experience of collecting the dataset and some lessons learned in the process. The second contribution of this paper, is the experience of using temporal logic for activity recognition. Using temporal logic allows specifying activities as complex events of object usage which can be described at different granularity. It also expresses temporal order between events thus palliating a limitation of ontology based activity recognition. The results on using the CADP toolbox [7] for ADL recognition in the real life collected data are very good.

The rest of this paper is organized as follows: Section 2 presents the experience and the setting up of our experience to collect *ContextAct@A4H*, a real life dataset of ADL and context data. Section 3 presents the formalism of temporal logic used for ADL recognition in real life settings. Section 4 presents the results of using temporal logic in the recognition of ADL in the *ContextAct@A4H* dataset. In Section 5 we present a summary of related work in activity recognition methods and real-life datasets. Finally, in Section 6 we present the conclusions of this work and research perspectives.

# 2 Amiqual4Home System Architecture and Experience Setup

Amiqual4Home is an experimental platform consisting of a smart apartment, a rapid prototyping platform and tools for observing human activity. For collecting the *ContextAct@A4H* dataset of activities of daily living, a 28 years old woman lived in the apartment during one week in July 2016 and three weeks in November 2016. The woman is a part of the research group designing the experiment.

In the following we first describe the sensing infrastructure at the apartment and then present the activity annotation process and the dataset itself.

## 2.1 Sensing infrastructure at the smart home

The Amiqual4Home apartment is equipped with 219 sensors and actuators. To make activity monitoring non-intrusive, in our experience all sensors are ambient sensors. No wearable sensors nor cameras were used for this experiment. Sensors allow observing both object usage and context conditions of each room and the exterior. Measuring context conditions is one of the new contributions of the *ContextAct@A4H* dataset with respect to other publicly available datasets.

We measured the following context variables: temperature, CO2, noise, humidity, presence and music information for each room and weather information for the exterior. Appliance and object usage are measured through electric/water consumption sensors, contact sensors and state change sensors (for lamps, curtains and switches). These measurements are precise indicators of object usage. Other sensors are indirect measures of object usage such as pressure sensor in the bed. The main advantage of the sensing methods we used is that sensors do not interfere with the normal use of objects (no RFID tags that need to be taken care of for example). Table 1 presents a summary of the sensing infrastructure. Some sensors, as those related with energy consumption, are integrated in a single device.

Amiqual4Home uses the OpenHab [6] integration platform for all the sensors and actuators installed. Sensors use different communication protocols such as KNX, MQQT or UPnP to send measurements to the central server. To preserve privacy, only local network access is permitted to the sensing infrastructure, no cloud platforms are used. The general architecture of the platform is shown in Fig. 1.

## 2.2 Activity annotation

The choice of activities to be annotated was motivated by two main concerns: being in concordance with available datasets and with the classification of basic

[6] http://www.openhab.org/

| Sensing Devices (# variables) | Measured Property | Locations |
|---|---|---|
| *Interaction Sensors* | | |
| Water consumption meters (15) | Hot or Cold water faucet usage | Handwasher, Shower, Toilet, Sink |
| Power consumption meters (11) | Appliance usage (direct measure) | Stove, fridge, exhaust fan, washing machine, electric dishwasher and heating systems (5) |
| Voltage consumption meters (11) | Appliance usage (direct measure) | Stove, fridge, exhaust fan, washing machine, electric dishwasher and heating systems (5) |
| Current consumption meters (11) | Appliance usage (direct measure) | Stove, fridge, exhaust fan, washing machine, electric dishwasher and heating systems (5) |
| Other electric-related meters (Energy, frequency, power factor, counter status) (44) | Appliance usage (direct measure) | Stove, fridge, exhaust fan, washing machine, electric dishwasher and heating systems (5) |
| Pressure Sensors (1) | Presence on furniture | Bed |
| Contact sensor (21) | Door/Window State (open — closed) | Kitchen gabinets (5), fridge, Drawers (2), Bedroom Door, Studio Door, Bathroom Door, Main Entrance Door, Closet Door, Terrace Door, All Windows (8) |
| Curtain Aperture Sensor (8) | Curtain State (open—close and aperture percentage) | All curtains |
| Lamp State Sensor (17) | Lamp state (on—off and luminosity percentage) | Living room (4), Bedroom (4), Studio (1), Kitchen (2), Floor Hall (2), Main Entrance Hall (2), Bathroom (2) |
| Smart Electric Outlet (3) | Appliance On — Off status | TV, Coffee Machine, Blender |
| Hue Lamp State (5) | Hue Lamp State (on — off), Color, Intensity | Living Room (3), Bedroom, Studio |
| Switches (54) | Change of lamp/curtain/music state | Throughout the apartment |
| Thermostat ( 5 ) | Thermostat Temperature | Living room, Dining room, Bedroom, Studio, Bathroom |
| *Context feature sensors* | | |
| Microphone (7) | Noise Levels | Kitchen, Living room, Dining room, Floor Hall, Main Entrance Hall, Bedroom, Studio |
| Infrared Sensor (6) | Movement in Room | Kitchen, Living room, Dining room, Bedroom, Studio, Bathroom |
| Luminosity Sensor (6) | Luminosity level in Room | Kitchen, Living room, Dining room, Bedroom, Studio, Bathroom |
| CHT device (12) | $CO_2$, Temperature and Relative Humidity of room | Kitchen, Living room, Bedroom, Bathroom |
| OpenWeather Information (14) | Exterior weather variables | Exterior |
| Music Information (25) | Speaker Status, Music Volume, Song title, Song Artist, Song Genre | Kitchen, Bedroom, Studio, Living Room, Bathroom |
| Indoor Positioning System (1)[5] | Inhabitant Position | Wearable device |

Table 1: Sensing devices deployed at the Amiqual4Home apartment

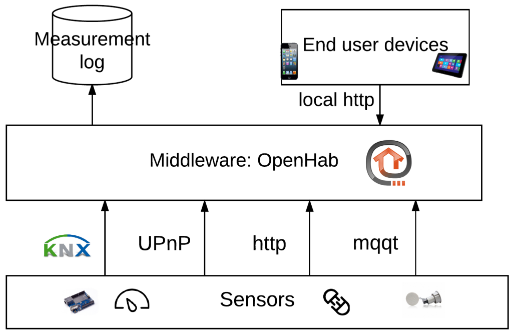


Fig. 1: General architecture of the data collection system

and instrumental ADL that can be measured at home. Some activities can be measured at different granularity levels. For example eating is annotated as eating breakfast in the ARAS dataset [1]. We chose high level labels to reduce the number of different activities preserving the desired semantics while considering time of day as part of context. Some activities were not annotated (like doing laundry) but the event using washing machine fully corresponds with this activity (same for watching tv). The following is the list of annotated and measured activities.

**Activities of Daily Living**
- Bathing/Showering (Annotated)
- Toileting (Annotated)
- Feeding/Eating (Annotated)
- Mobility (Unannotated but measured by bspoon sensor)
- Dressing (Unannotated but recognized by opening of dressing drawers)

**Instrumental Activities of Daily Living**
- Preparing meals (Annotated as Cook)
- Housework. Two finer activities are found: wash dishes (Annotated) and do laundry (Unannotated but measured through washing machine use).
- Other instrumental activities not included

**Other Annotated Activities (for correspondence with other datasets)**
- Sleeping
- Going out

**Other Unannotated but Measurable Activities**
- Watching TV
- Use of handwasher (for handwashing, brushing teeth and general grooming)

Activities were annotated by self-reporting in place at the moment of starting and ending each activity. For this, three different interfaces were available: a web and mobile app, a remote control and a button interface placed strategically


Fig. 2: Interfaces used to annotate the *ContextAct@A4H* dataset.

(see Fig. 2). The combination of annotation methods allowed the inhabitant to remember easily to do the annotations and minimized interruptions of the activities. The inhabitant annotated the beginning and end of each activity. The inhabitant reported missing some annotations, specially at the end of the second period when there was more familiarity with the environment. This confirms the fact that sensing technology "dissapears" when it truly blends with our environments.

### 2.3 *ContextAct@A4H* dataset

We collected a dataset for one week during summer and three weeks during fall in the same apartment configuration. Having data of different months of the year enables both the analysis of the generalization of a model of activities and the analysis of changing behavior throughout the year. As mentioned before, 219 properties were measured during the whole experiment. For the duration of the experiment, all sensors logged their measurements successfully. The accuracy of the measurements was not verified and filtering methods were not used. The properties represent either object interactions or context features. Most object interactions are measured redundantly. For example, opening the fridge is measured by the electric consumption and the door state of the fridge. In this way, the dataset enables many different forms of detecting activities. Context features are measured per room. They include: temperature, carbon dioxide level, noise level, relative humidity, luminosity level and presence in the room. Each context feature is measured at a constant rate, but a change point representation is available. A change is considered when the value varies more than 2% with respect to the last measurement. Additionally, weather information from OpenWeather[7] is logged every hour and visitor presence is self-reported by the user.

The dataset with changepoint representation of features contains a total of 1 473 011 tuples (364394 in july and 1108617 in november) of sensor measurements. A tuple consists of a timestamp, a sensor id and a measurement value. Of those 397 correspond to activity records (157 in july and 240 in november). Figure 3 shows the distribution of time in each activity.

[7] https://openweathermap.org/

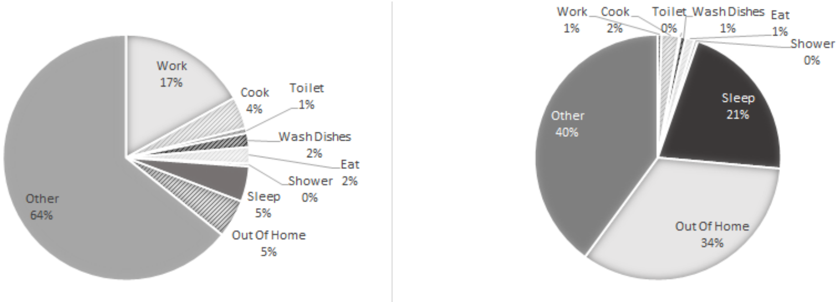


Fig. 3: Distribution of activities during summer (left) and fall (right).

Other data logged that can be exploited for different applications is the music information (title, artist, genre and timestamp) corresponding to the music the inhabitant listened while in the apartment.

## 3 Using the MCL Temporal Logic to Monitor Activities in Logs of Events

An alternative to self-reporting of activity annotations is automatic detection of activities by analysing the sensor raw data present in the dataset. In this section, we report on our use of model checking techniques to do so.

*Model checking* [4] is a formal verification technique that consists in exploring a user-written *formal model* of a system in order to check the satisfaction of properties describing its proper functioning. The formal model generally has the form of an automaton obtained from a description in a language with mathematically defined semantics. The most general way of defining properties is to use *temporal logics* [17], which combine the standard connectors of logic with modalities and fixpoint operators that enable to reason about the necessity or possibility to reach certain states of the model.

In this work, we consider so-called *action-based* temporal logics, which are appropriate to reason about the actions which involve a state change in the model (e.g., the communication actions in a distributed/concurrent system), rather than the contents of states (e.g., the state variables).

In cases where there is no formal model of a system, it turns out that action-based temporal logics can also be used to analyze sequences of actions generated by a concrete, running system (e.g., logs of events). Indeed, such sequences can be (trivially) seen as particular cases of automata. The model can be simply replaced by the log, which becomes the object on which checks are performed. In this work, we thus naturally consider *ContextAct@A4H* as log of events.

In practice, we use the CADP toolbox [7], which is equipped for such a task. Among the (more than) 50 tools and code libraries present in CADP, we use in particular the following ones:

– SEQ.OPEN [8] is a tool that can parse a sequence of actions in the SEQ format of CADP and provide an application programming interface with the CADP verification tools, following the principles of OPEN/CAESAR [6].
– EVALUATOR 4.0 [16] is a model checker that evaluates temporal logic formulas written in the language MCL, an alternation-free modal mu-calculus [11] extended with data handling, regular expressions, and fairness operators. EVALUATOR 4.0 can be used to verify formulas on any system connected to the OPEN/CAESAR interface, thus including action sequences in the SEQ format.

In the framework our experiment, each event constitutes a line in the log, in CSV format. Each line contains first the event occurrence date (including day time), the sensor id, and the sensor value, separated by semicolons. The first step is to translate this log into the SEQ textual format of CADP, which is done using a simple Unix sed script. The transformation principally consists in turning lines of the form "`date; sensor_id; sensor_value`" into the form "`A !date !sensor_id !sensor_value`" accepted as input by SEQ.OPEN, where `A` is a dummy symbol added here for technical (and unimportant) reasons, and where "`!`" is the symbol used to delimit action fields in CADP.

MCL macro definitions can then be defined to interpret log lines as more abstract events. Each macro definition can be understood as an action pattern that recognizes certain fields in log lines. For instance, the following macro definition recognizes that the light of the office is on, sensor `L14` corresponding to a light dimmer located in the office of the Smart apartment:

```
macro Office_Light_On () =
  { A ... !"L14" ?value:Nat where value > 0 }
end_macro
```

Note how the "`...`" symbol (wildcard) allows the date field to be ignored and how the "`?`" symbol allows some typed field to be captured in a variable, whose value can be constrained by a Boolean condition (guard) specified by the "`where`" clause.

Macro definitions can be parameterized as the following one, which matches log lines indicating that a light button in the bathroom (whose sensor_id starts with I7_ or I11_) has the value SWITCH passed as parameter:

```
macro Bathroom_Button (SWITCH) =
  { A ... ?snsr:String SWITCH where
    (prefix (snsr, 3) = "I7_") or (prefix (snsr, 4) = "I11_") }
end_macro
```

Other, more complex macro definitions can be defined by combining simpler ones using Boolean connectors. For instance, the macro definition below detects the fact that either the oven or the cooktop is switched on, where `Cooktop_On` and `Oven_On` are already defined macro definitions:

```
macro Cooking_Appliance_On () = (Cooktop_On or Oven_On) end_macro
```

In our experiment, we wrote 125 macro definitions.

Once macros have been defined to give simple meanings to complex log events, thus facilitating the log analysis, they can be used in temporal logic formulas to detect activities in the log. We give two examples.

The first example below is a formula that detects the beginning of a shower activity, which is defined as follows: the door of the shower is open then closed, and then the water flows. Angles denote the existence of a sequence of events, defined by a regular expression. The symbol "`.`" denotes sequential composition, "`*`" denotes repetition (0 or more times), `true` matches any log event, and `not` and `or` have their usual meaning. This formula evaluates to true if their exists a sequence in the log that matches this regular expression.

```
< true* . Shower_Door (!"OPEN") .
  (not Shower_Door (!"CLOSED"))* . Shower_Door (!"CLOSED") .
  (not Shower_Door (!"OPEN"))* . Shower_Water_On > true
```

Note how the "`!`" symbol allows a field to be matched against the value of a data expression (here, a character string constant).

The second example, defined in Figure 4, is slightly more complex. It detects the start of a cooking activity, defined as follows: a food container (the fridge or a drawer) was opened and at least one cooking appliance is on.

This formula is defined as a least fixpoint (`mu`) which, evaluated on a sequence (as opposed to a general automaton), can be seen as a recursive function on the events of the log. It has two parameters, namely a Boolean `Fd_Cntnr_Opn`, which is initially false and becomes true once an event corresponding to the opening of a food container is detected, and a natural number `N_Ckng_Appl_On`, which counts the number of cooking appliances that are currently on (initially 0). When `Fd_Cntnr_Opn` is true and `N_Ckng_Appl_On` is greater than 0, then the start of a cooking activity was detected. Otherwise, the property is applied recursively depending on the next event that is found in the sequence, parameters being updated accordingly.

CADP detects the time the activity starts as the time when all conditions defined by the model are met. It creates a file with all sensor measurements up to this time as an output. To detect all occurrences of an activity we find the difference between the original dataset and the output file using the *grep* utility on Linux, and then use this new file as the input log file. This process is repeated until the output file is empty.

## 4 Experimental Evaluation

We evaluated the model checking approach for activity start and/or end recognition using *ContextAct@A4H* taking annotation as ground truth. We defined models to recognize the start of the sleeping, cooking and washing dishes activities and the end of the showering and using toilet activities. When the model infers the start or end of the same activity within 3 minutes, we take the smaller time (greater time) as the start time of the activity (end time of the activity). We

```
mu Ckng_Act (
   Fd_Cntnr_Opn: Bool := false, (* true if fridge or drawer was opened *)
   N_Ckng_Appl_On: Nat := 0 (* nb of cooking appliances that are on *)
) .
   if Fd_Cntnr_Opn and (N_Ckng_Appl_On > 0) then
      true (* cooking activity detected *)
   else
      < Food_Container_Door (!"OPEN") > Ckng_Act (true, N_Ckng_Appl_On)
      or
      < Cooking_Appliance_On > Ckng_Act (Fd_Cntnr_Opn, N_Ckng_Appl_On+1)
      or
      < Cooking_Appliance_Off >
         if N_Ckng_Appl_On > 1 then
            Ckng_Act (Fd_Cntnr_Opn, N_Ckng_Appl_On-1)
         else (* was not a cooking activity *)
            Ckng_Act (Fd_Cntnr_Opn, 0)
         end if
      or
      < not Food_Container_Door (!"OPEN") and
        not Cooking_Appliance_On and not Cooking_Appliance_Off >
         Ckng_Act (Fd_Cntnr_Opn, N_Ckng_Appl_On)
   end if
```

Fig. 4: MCL formula for detecting the start of a cooking activity

calculate the difference between the inferred time and the actual time annotated by the inhabitant. Table 2 shows the results obtained.

| Activity | Precision | Recall | Avg. time diff (minutes) |
|---|---|---|---|
| Sleep (start) | 78 % | 95% | 4,42 |
| Toilet use (end) | 98 % | 78 % | 0,71 |
| Cooking (start) | 81 % | 88% | 1,5 |
| Taking a shower (end) | 70 % | 89 % | 3 |
| Washing dishes (start) | 14% | 97% | 12,45 |

Table 2: Activity recognition results with model checking approach

To evaluate the performance of the activity recognition approach we calculate its *precision* and *recall* considering that the annotated activities are the relevant ones that should be recognized. A recognized activity is considered "relevant" if it corresponds effectively to an annotated activity. *Precision* is the fraction of the recognized activities that are relevant wrt the number of recognized activities. *Recall* is the fraction of annotated activities that are successfully recognized wrt the number of annotated activities.

Most of the activities have high precision and recall measures, similar to results reported with hidden markov models or support vector machines in other datasets. To provide a comparison, the highest average f-measure (a weighted average of the precision and recall ) reported by Krishnan et al. [12] was 0.61 while our average f-measure is 0.72. This depends on many factors such as sensors used, features selected and the number of activities making the comparison non-depending on just the recognition method. In our case, the high precision is due to the precise object usage measurements that we gathered thanks to the sensing devices installed. We can see that the average time difference between the inferred start or end of the activity and the time the inhabitant annotated is rather small. The activity washing dishes has a low precision due to a high number of false positives. The false positives are due to the fact that the only detectable event corresponding to this activity is the sink faucet being opened. Yet, the sink faucet can also be opened for many other activities including washing fruits and vegetables, washing hands, and washing few pieces of tableware while cooking, which were not annotated as washing dishes. Possible ways to increase precision could be to take into account both a minimal proportion of hot water flowing through the sink faucet (taking into account that vegetables and fruits are usually washed with cold water whereas dish washing requires hotter water) and the time during which the sink faucet has been opened (thus requiring to define a minimum time that dish washing usually takes and that hand washing or tableware washing usually do not exceed).

## 5 Related Work

In this section we briefly discuss related work on activity recognition and current available datasets of daily living in smart homes.

### 5.1 Daily Living Activity Recognition in Real Life Scenarios

Activity recognition refers to inferring an activity label from observations made through sensing devices. These sensing devices can be either vision-based, wearable sensing or ambient sensing. Vision based sensing is often seen as a privacy invasion and its use is not accepted by many elders. Wearable sensing is better for physical activity such as running or walking and is now commercially available for this. For daily living activities however, wearables can be forgotten by the user and have been used for fined-grained activities such as grabbing a glass or turning the lights on [15]. Ambient sensing allows inferring object use and coarsed-grained activities. Since activities are often performed with the same objects this is better for daily living monitoring. Also, it is less intrusive as sensing capabilities merge with the environment and do not interfere with normal activity performance.

Sensor-based activity recognition in smart homes methods can be classified in knowledge based and data based [3]. Knowledge based methods use semantic descriptions of activities based on object usage and context features such as location or time of day. The most common methods include ontologies [20], situational models and event calculus. Data-based methods, on the other hand, use sensor features to build machine-learned models. The models can be learned with supervised or unsupervised methods. Supervised methods, such as Hidden Markov Models [22], Bayes Networks and Support Vector Machines [12] have shown promising results but need ground truth data to be learned. Unsupervised methods such as clustering, topic models and frequent sequences do not need ground truth data and have been used as methods for activity discovery [18].

Most proposed activity recognition methods have been tested in scripted datasets showing promising results. Nevertheless, passing from laboratory settings to real life settings is not as transparent since there are crucial differences in both settings. Activity recognition in real-life datasets differs in the following aspects from activity recognition in scripted datasets:

- Class imbalance: real-life scenarios are inherently imbalanced due to the different frequencies and durations of each activity [15].
- Intra-class variability: the same activity can be performed in different ways, using different objects, at different locations, etc. [14, 15].
- The *Other* activity: scripted datasets only represent activities of interest, while in real-life scenarios most of the time it is other activity is being performed [12]. This 'other' activity is a highly variable class since it represents many activities.

### 5.2 Datasets of daily living in smart homes

Testing activity recognition and behavior modeling algorithms in real life is difficult due to the lack of real data. Real-life scenarios are difficult to gather both for the costs of equipping a real home and the availability of subjects willing to live in such homes. Therefore, making these datasets public helps advance research and is of great interest for the community. Some equipped apartments over the world, notably: The Aware Home [10] at Georgia Tech, Domus at France, Van Kasteren [22], CASAS at Washington University [5], PlaceLab [21] at MIT, ARAS [1] at Istanbul, Turkey. Most publicly available datasets feature scripted and acted daily life activities, lack ground truth annotations or feature office or other scenario activities that do not correspond to daily living activities.

Nonetheless, the Placelab, CASAS and ARAS projects have publicly available datasets of real life daily activities with ground truth annotations. Common activities in these datasets include sleeping, toilet use, taking a shower or bath, eating, leaving home, cooking or preparing meals, washing dishes, working or using computer and leisure activities (relax, watching tv, reading). These are daily living activities that can be measured through available sensors. Eating and fine-grained activities are usually the most complex to recognize [12, 21]. Context features are often not measured, being the ones measured temperature at some locations and presence in rooms. Nonetheless, rich context information can not only improve activity recognition but also it can help to better understand behavior patterns and user triggers (for example, loud noises may trigger anxiety) [13].

## 6 Conclusions and research perspectives

In this paper we have detailed our approach of using temporal logic for ADL recognition in real-life settings showing promising results. We use the CADP tool to describe each activity as a temporal logic property that is checked repeatedly in the log of sensor measurements until all occurrences of the activity have been found. The property is described based on object usage by an expert (in our case the inhabitant). One of the main advantages of this approach over other techniques is the ability to recognize start and end of the activity, thus not requiring to segment sensor data. The model checking approach using CADP also enables to specify durations and gaps between events, by capturing the date fields present in events and storing them as parameters of fixpoint operators. We plan to exploit this feature in the future for carrying out timed analyses of activities. We believe model checking is an interesting alternative for activity recognition in real life settings.

We have also presented the *ContextAct@A4H* dataset, an annotated real-life dataset of activities and context of daily living covering four weeks of data. One drawback in public datasets for ADL recognition is that there is no correspondence between the activities nor the sensors used, which makes it difficult to compare results from one dataset to the other. This is since available sensors change through time. Data in terms of semantic measured properties can be more stable and we propose to focus on these to compare results and for future datasets. Still, the need of creating shared knowledge and agree on a taxonomy of activities [2] continues, although some common activities emerge. Most datasets include other activities so more data is available but these are not common with other datasets. We have presented a dataset with these common activities and some others as well. The dataset is described with both the sensor measurements and with semantic measured properties.

*ContextAct@A4H* is the first daily living dataset featuring rich context information. It features six context variables per room in the apartment plus weather, music and number of visitor information. This information is of high interest for applications such as content recommendation, domotics and behavior analysis among others.

## References


1. Alemdar, H., Ertan, H., Incelt, O.D., Ersoy, C.: ARAS Human Activity Datasets In Multiple Homes with Multiple Residents. pp. 232–235 (2013)
2. Brush, A., Krumm, J., Scott, J.: Activity Recognition Research: The Good, the Bad, and the Future. In: Pervasive 2010 Workshop: How to do Good Research in Activity Recognition. pp. 1–3 (2010)
3. Chen, L., Hoey, J., Nugent, C.D., Cook, D.J., Yu, Z.: Sensor-based activity recognition. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42(6), 790–808 (Nov 2012)
4. Christel Baier, J.P.K.: Principles of Model Checking. MIT Press (2008)
5. Crandall, A.S., Krishnan, N.C., Thomas, B.L., Cook, D.J.: Casas: A smart home in a box. Computer 46, 62–69 (2013)

6. Garavel, H.: OPEN/CÆSAR: An Open Software Architecture for Verification, Simulation, and Testing. In: Steffen, B. (ed.) Proceedings of the First International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'98), Lisbon, Portugal. Lecture Notes in Computer Science, vol. 1384, pp. 68–84. Springer-Verlag, Berlin (Mar 1998)
7. Garavel, H., Lang, F., Mateescu, R., Serwe, W.: CADP 2011: A Toolbox for the Construction and Analysis of Distributed Processes. Springer International Journal on Software Tools for Technology Transfer (STTT) 15(2), 89–107 (Apr 2013)
8. Garavel, H., Mateescu, R.: Seq.open: A tool for efficient trace-based verification. In: Graf, S., Mounier, L. (eds.) Proceedings of the 11th International SPIN Workshop on Model Checking of Software (SPIN'04), Barcelona, Spain. Lecture Notes in Computer Science, vol. 2989, pp. 150–155. Springer-Verlag (Apr 2004)
9. Garcia, E.: Learning from Imbalanced Data. IEEE Transactions on Knowledge and Data Engineering 21(9), 1263–1284 (sep 2009)
10. Helal, S., Mann, W., El-Zabadani, H., King, J., Kaddoura, Y., Jansen, E.: The Gator tech smart house: A programmable pervasive space. Computer 38(3), 50–60 (2005)
11. Kozen, D.: Results on the Propositional $\mu$-Calculus. Theoretical Computer Science 27, 333–354 (1983)
12. Krishnan, N.C., Cook, D.J.: Activity recognition on streaming sensor data. Pervasive Mob. Comput. 10, 138–154 (Feb 2014)
13. Lago, P., Jiménez-Guarín, C., Roncancio, C.: Contextualized behavior patterns for change reasoning in Ambient Assisted Living: A formal model. Expert Systems *To Appear*
14. Lago, P., Jiménez-Guarín, C., Roncancio, C.: A Case Study on the Analysis of Behavior Patterns and Pattern Changes in Smart Environments. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 8868, pp. 296–303 (2014)
15. Logan, B., Healey, J., Philipose, M., Tapia, E.M., Intille, S.: A long-term evaluation of sensing modalities for activity recognition. In: Proceedings of the 9th International Conference on Ubiquitous Computing. pp. 483–500. UbiComp '07, Springer-Verlag, Berlin, Heidelberg (2007)
16. Mateescu, R., Thivolle, D.: A model checking language for concurrent value-passing systems. In: Cuellar, J., Maibaum, T., Sere, K. (eds.) Proceedings of the 15th International Symposium on Formal Methods FM'08 (Turku, Finland). Lecture Notes in Computer Science, vol. 5014, pp. 148–164. Springer-Verlag (May 2008)
17. Pnueli, A.: The temporal logic of programs. In: Proceedings of Foundations of Computer Science. pp. 46–57. IEEE (1977)
18. Rashidi, P.: Stream Sequence Mining for Human Activity Discovery. In: Plan, Activity, and Intent Recognition. pp. 123–148. Elsevier (2014)
19. Riboni, D., Pareschi, L., Radaelli, L., Bettini, C.: Is ontology-based activity recognition really effective? In: 2011 PERCOM Workshops. pp. 427–431 (March 2011)
20. Rodríguez, N.D., Cuéllar, M.P., Lilius, J., Calvo-Flores, M.D.: A survey on ontologies for human behavior recognition. ACM Comput. Surv. 46(4), 43:1–43:33 (Mar 2014)
21. Tapia, E.M., Intille, S.S., Larson, K.: Activity recognition in the home using simple and ubiquitous sensors. In: Pervasive Computing. Lecture Notes in Computer Science, vol. 3001, pp. 158–175 (2004)
22. Van Kasteren, T.L.M., Englebienne, G., Kröse, B.J.A.: Transferring knowledge of activity recognition across sensor networks. Lecture Notes in Computer Science 6030 LNCS, 283–300 (2010)