Towards Active Learning Interfaces for Multi-Inhabitant Activity Recognition

Claudio Bettini, Gabriele Civitarese
EveryWare Lab, Dept. of Computer Science
University of Milan
Milan, Italy
{claudio.bettini, gabriele.civitarese}@unimi.it

***Abstract*—Semi-supervised approaches for activity recognition are a promising way to address the labeled data scarcity problem. Those methods only require a small training set in order to be initialized, and the model is continuously updated and improved over time. Among the several solutions existing in the literature, active learning is emerging as an effective technique to significantly boost the recognition rate: when the model is uncertain about the current activity performed by the user, the system asks her to provide the ground truth. This feedback is then used to update the recognition model. While active learning has been mostly proposed in single-inhabitant settings, several questions arise when such a system has to be implemented in a realistic environment with multiple users. Who to ask a feedback when the system is uncertain about a collaborative activity? In this paper, we investigate this and more questions on this topic, proposing a preliminary study of the requirements of an active learning interface for multi-inhabitant settings. In particular, we formalize the problem and we describe the solutions adopted in our system prototype.**

***Index Terms*—active learning, interface, multi-inhabitant, activity recognition**

I. Introduction

Smart-home activity recognition is a research field which has been deeply studied in the last decades [1]. This topic is still hot in the pervasive computing community, since continuously monitoring the activities performed in a home environment by its inhabitants enables several health-care applications, like the early diagnosis of cognitive disorders for the elderly population [2]. Typically, the activity recognition problem is tackled with supervised machine learning methods [3]. While these approaches lead to high recognition rates, it is unfeasible to acquire the high amount of labeled data that they require. For this reason, semi-supervised learning techniques for activity recognition are arising [4]. Those methods require a small amount of labeled data to initialize the recognition model, which is continuously improved over time. Among the many semi-supervised learning approaches, *active learning* is one of the most effective [5], [6]. An active learning approach consists of asking the user a feedback about the activity she is performing when the model is uncertain about the current prediction. The feedback is then used to update and improve the recognition model [7].

The majority of works in the literature proposed methods to detect activities in single-inhabitant settings. On the one hand, this scenario is realistic considering the amount of elderly subjects which live alone in their home [8]. However, it often happens that multiple subjects live in the same home. Those subjects may perform the same activity in cooperation or different individual activities in parallel [9]. Hence, multi-inhabitant activity recognition methods have been proposed to tackle this issue [10].

Since we are currently investigating the application of semi-supervised methods based on active learning to multi-inhabitant settings, we faced an interesting research question: *who to ask a feedback when the system is uncertain about collaborative activities?* The answer to this question is not trivial. In order to minimize the intrusiveness, we do not want to send a question to all the users involved in the uncertain collaborative activity. The choice of the user to query should be based on context information (e.g., his/her availability).

Another important aspect to consider is the type of interface which is used to prompt the query. A naive approach would consist in prompting the query directly on the personal device of the selected user (e.g., smartphone, smartwatch). However, notifications on personal devices can be considered very distracting. A promising direction is deploying one or more vocal assistants in the home, thus allowing the inhabitants to provide a feedback using their voice without physically interacting with prompting devices.

In this paper, we investigate the requirements and the emerging issues that arise in the design of an active learning interface for multi-inhabitant activity recognition systems. Hence, we hypothesize realistic solutions to address this problem. Our method considers the users context to decide how to prompt active learning queries. The considered context includes the availability and the interruptibility of each inhabitant.

The paper offers the following main contributions:

- A formalization of the problem of prompting active learning queries in multi-inhabitant environments
- The identification and discussion of the main challenges in multi-inhabitant active learning systems
- A description of the solutions adopted for our system prototype.

This paper is organized as follows. In Section II we discuss about the related work on smart-home interfaces which inspired this work. Section III formalizes the problem of prompting active learning queries in multi-inhabitant settings.

Our method is presented in details in Section IV. In Section V we discuss alternative interfaces that can be used in multi-inhabitant active learning systems. Finally, Section VI concludes the paper.

II. Related work

Active learning for activity recognition has been widely studied in the literature [4]–[7]. However, the majority of proposed methodologies simply evaluated their methods on existing datasets, simulating the user feedback using the ground truth. We believe that a more realistic evaluation of active learning effectiveness should involve real smart-home interfaces that prompt queries to the inhabitants while they are performing daily activities.

The main requirements of a smart-home interface are usability and acceptability [11]. In this direction, the work in [12] compared different smart-home interfaces modalities. From that study, it emerges that users appreciate voice interfaces since they can be used from anywhere in the home without using hands. However, touch interfaces are sometimes considered a better option since they make users feel more comfortable, perceiving a stronger sense of control. That study also highlights that users would rather use a wall mounted interface (e.g., a tablet) rather than a personal mobile device (e.g., a smartphone), since the former is perceived as less distracting.

The majority of smart-home interfaces proposed in the literature are mainly related to domotics applications [11], [13]. Nowadays, many commercial solutions are very common (e.g., Alexa, Google Home) [14]. Those interfaces are currently not suitable for active learning, since they do not actively interact with the user. Indeed, those interfaces require an input from the user to start the conversation. Moreover, while existing solutions implement voice recognition methods to understand which user is communicating with the interface, this is mainly related to access control.

Finally, an important aspect to consider in order to enable real-time active learning is user's interruptibility [15]. Indeed, it is necessary to analyze user's context (e.g., the current activity, the prompting modality, the importance of the prompt) to understand if she can be interrupted at a specific time instant to provide a feedback [16].

To the best of our knowledge, no prior work investigated active learning interfaces for multi-inhabitant activity recognition. Our solution takes advantage of inhabitants' contextual information to decide how to prompt active learning queries with the objective of achieving a good trade-off between acceptability, usability and utility.

III. Problem formulation

Let $\mathbf{U} = \{u_1, u_2, \dots, u_n\}$ be the set of users (the inhabitants of the smart-home) and $\mathbf{A} = \{A_1, A_2, \dots, A_n\}$ the set of considered human activities. The smart-home system (named just *system* in the following) continuously records a stream of time-stamped sensor events and, given an instant t and a user u, it provides a stream $s(u)^t$ of sensor events associated with user u and collected in a time window $[t, t-k]$ where k is the window size parameter. For example, suppose that Alice turns on the cooker at time t'. The corresponding sensor event (and its timestamp) generated by the plug sensor connected to the cooker is recorded by our system, and it is part of $s(Alice)^t$ when $0 \leq t - t' \leq k$.

The overall goal of the system is to periodically predict, for each user, the activity that she has been performing. We also would like the system to identify situations in which activities are jointly performed by multiple users. Formally, given an instant t, the system should return a set of tuples $PA_t = \{\langle (u_r, \dots, u_s), A_i \rangle | u_r, \dots, u_s$ *are all the users predicted to jointly perform activity* A_i *at time* $t\}$.

Since we assume that each user is performing a single activity at a time, the same user cannot appear in more than one tuple. Moreover, each user that is present in a monitored room should appear in a tuple.

Since the system we are considering is based on active learning, the final prediction is obtained by evaluating the probability of users performing a given activity, and actively querying users whenever the system confidence on the prediction is insufficient.

For the sake of usability, we assume that queries are directed to a specific user $u^\star$, and ask which of two activities a set of users U (with $u^\star \in U$) is performing. A third alternative is given with the answer "*None of those*" with the intended semantics that U is neither performing A_i nor A_j. Hence, a technical problem we are facing is specifying at each time t in which the system requires a feedback, the parameters of the query

$$Q^t(u^\star, U_t, A_i, A_j)$$

where $u^\star$ is the query target user, and U_t is the set of candidates users collaboratively performing activity A_i or activity A_j.

IV. System architecture and solution

In this section we show our multi-inhabitant activity recognition system based on active learning. The architecture of our method is depicted in Figure 1. In the following, we describe in detail each component of our architecture.

A. Sensing infrastructure

In this work, we assume that users perform activities in a smart-home equipped with several environmental sensors. Those sensors are in charge of monitoring the interaction of each user with the environment. Examples of such sensors are magnetic sensors to detect opening/closing of drawers and doors, pressure sensors on the chairs to detect when inhabitants are sitting, power plugs to detect home appliances usage, etc.. The stream of raw sensor data is continuously transmitted in real-time to the MULTI-INHABITANT ACTIVITY RECOGNITION module.

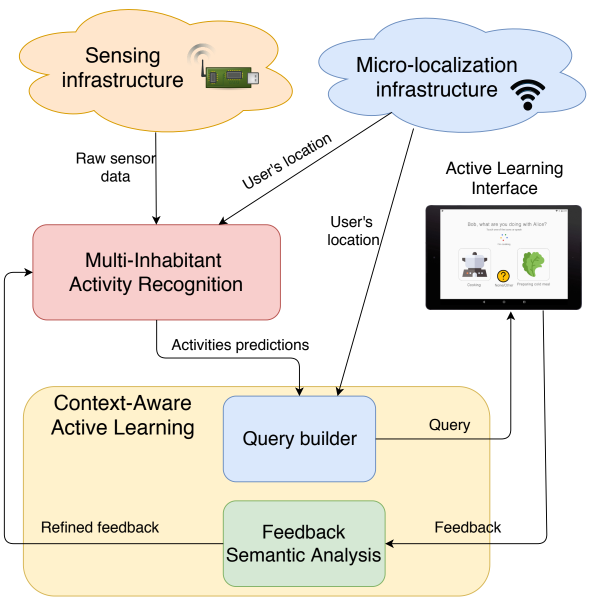
Sensing infrastructure
Micro-localization infrastructure
Raw sensor data
User's location
User's location
Active Learning Interface
Multi-Inhabitant Activity Recognition
Activities predictions
Context-Aware Active Learning
Query builder
Query
Refined feedback
Feedback Semantic Analysis
Feedback

Fig. 1. The architecture of our system.

B. Micro-localization infrastructure

The MICRO-LOCALIZATION INFRASTRUCTURE is in charge of inferring at each time t the location $l(u)^t$ of a user u in the home. In our system prototype, each user u wears a smart-watch which is in charge of continuously collecting WiFi signal strengths to infer $l(u)^t$. While this approach identifies and locates each user in the home, it requires users to continuously carry their personal devices. Ideally, for the sake of usability, micro-localization should be based on wearable sensors or other technologies which do not impose the use personal devices.

For the sake of this work, we assume to have a micro-localization service which is always accurate in computing $l(u)^t$ at the room granularity. As we will explain in the following, user's location is a context information which is used both in MULTI-INHABITANT ACTIVITY RECOGNITION and CONTEXT-AWARE ACTIVE LEARNING modules.

C. Multi-inhabitant activity recognition

Periodically, the system computes, for each user u, the probability distribution over the possible activities being performed by u at the current time t:

$$h(s(u)^t) = \langle p_{A_1}, p_{A_2}, \ldots, p_{A_n} \rangle$$

where p_{A_i} is the probability $P(A_i | s(u)^t)$ that the user u at time t is performing activity $A_i \in \mathbf{A}$, based on the sensor data stream $s(u)^t$ as defined in Section III. We also have $p_{A_i} \in [0,1]$ $\forall i$ and $\sum_{i=1}^{n} p_{A_i} = 1$.

At each time instant t, the multi-inhabitant activity recognition layer forwards to the CONTEXT-AWARE ACTIVE LEARNING module the activity prediction $h(s(u))^t$ for each user u.

Note that associating environmental sensor events to the correct user (i.e., computing $s(u)^t$) is an open research problem (also known as *data association*) [17]. In our system prototype, data association is based on $l(u)^t$: we associate to each user the sensor events triggered in the room where she is currently located. This allows us to obtain, for each user $u \in \mathbf{U}$, a personalized $s(u)^t$ at each time t. Then, we compute $h(s(u)^t)$ taking advantage of a deep learning classifier.

For the sake of this paper, the specific method for data association is not relevant and we assume to have a system which accurately associates sensor events to the subject that triggered them. Regarding the classifier, our specific solution is also irrelevant, and the only requirement for this module is an on-line algorithm that is initialised using a small training set, and that can refine its model based on feedback.

D. Context-aware active learning

In the following, we show our context-aware approach to issue active learning queries in multi-inhabitant settings and our semantic-based method to refine the feedback provided by the users.

1) Query builder: The QUERY BUILDER module is in charge of using available context data to decide if at time t (when the periodical system prediction needs to be provided) a query is needed, and, if this is the case, which are the query parameters. Moreover, once the query is specified, it should also decide if the current situation is appropriate to interrupt the target user asking for a feedback.

This task involves first understanding if the system is uncertain about the activity performed by a user or a group of users U_t. If this is the case we need to determine the parameters of the query $Q^t(u^\star, U_t, A_i, A_j)$ which include deciding which user $u^\star \in U_t$ to query.

a) Defining uncertainty: In order to evaluate the system's uncertainty we use the entropy measure, which has been widely used for active learning [18]. For each $h(s(u)^t)$ received by the MULTI-INHABITANT ACTIVITY RECOGNITION module, the QUERY BUILDER computes the entropy H as follows:

$$H(h(s(u)^t)) = \sum_i p_{A_i} \log \frac{1}{p_{A_i}}$$

When $H(h(s(u)^t))$ is higher than a threshold δ, we assume that the system is uncertain about the activity currently performed by u at time t.

b) Deciding the set of users and the candidate activities: Intuitively, a group of users jointly performing an activity should include users that are in the same place, and that, according to the system prediction are performing the same activity[1]. If the system has high confidence on the predicted activity for all of these users, then there is no need to issue a query. However, when the system is uncertain about the activity performed by any of those users, we consider as candidate set U_t for a query the subset of the users for which the system shows uncertainty about the performed activity and

[1]Note that here we make the assumption that users that are performing the same activity in the same room at the same time are actually jointly performing the activity. This is indeed the case in our considered setting.

Algorithm 1 Choosing the user to query

```
Input: a group of users $U_t$, the threshold $\Omega$, and the overall
    amount of prompted queries $q$
Output: the chosen user $u^\star \in U_t$
1: $u^\star \leftarrow RANDOM(U_t)$
2: if $r \geq \Omega$ then
3:     $\epsilon \leftarrow \frac{1}{q}$
4:     $u^\star \leftarrow \begin{cases} \operatorname{argmax}_{u \in U_t} av(u), & \text{with probability } (1-\epsilon) \\ u^\star, & \text{with probability } \epsilon \end{cases}$
5: end if
6: return $u*$
```

having the same two activities as the most likely according to the classifier.

Formally, the conditions to identify U_t, A_i, and A_j are expressed as follows:

- $\forall u_r, u_s \in U_t$, $l(u_r)^t = l(u_s)^t$
- $\forall u \in U_t$, $H(h(s(u)^t)) > \delta$
- $\forall u \in U_t$, A_i and A_j are the two activities with the highest probability values in $h(s(u)^t)$, independently from their order.

Note that the first condition is about the location, the second is about the uncertainty of the system, and the third is identifying the two most likely common activities.

c) Deciding the user to be queried: In order to decide which user $u \in U_t$ should be the one to answer (the parameter $u^\star$ of the query), we keep track of each user availability with the function $av(u) \in [0, 1]$. Intuitively, the availability indicates the average "willingness" of the user to actively answer to prompted queries. Indeed, some users may be more reluctant than others in interacting with the active learning interface. The value of $av(u)$ is computed as the ratio of the number of queries answered by u over the overall number of queries prompted to u, and it is updated every time a query is prompted to u.

The specific method to find $u^\star$ is shown in Algorithm 1. Our intuition is that $u^\star$ should be one of the users in the group with the highest $av(u)$. Technically, our system initially randomly picks $u^\star$. When the overall number q of queries prompted by the system is greater than a threshold Ω, we start considering $av(u)$ to choose $u^\star$ instead of choosing randomly. However, in order to continuously improve the estimation of $av(u)$, we adopt an ϵ-greedy approach. Indeed, we assign $u^\star$ as one of the users with the maximum $av(u)$ with probability $1 - \epsilon$, while we still choose at random with probability $\epsilon = \frac{1}{q}$. Hence, the value of ϵ continuously decreases as the amount q of overall queries prompted by our system increases. This allows our method to converge to a stable estimate of $av(u)$. Note that, in order to maintain a simple notation, Algorithm 1 chooses the only user with the highest availability with probability $(1-\epsilon)$ at line 4. However, more than one user could potentially share the same highest value of availability. In that case, our algorithm chooses at random among the users with the highest availability value.

d) Deciding if submitting the query: We have shown how the QUERY BUILDER module can identify all the parameters and hence determine the complete query $Q^t(u^\star, U_t, A_i, A_j)$. However, to finally decide whether to issue the query or not, we evaluate the *interruptibility* of the users when performing the activities involved in the query. Intuitively, the interruptibility of a user u which is currently performing an activity A is the estimated probability that u will temporally suspend the execution of A (or concurrently continue A and answer the query) to provide a feedback to the active learning interface. For the sake of this work, we adopt the same interruptibility model for each user $u \in \mathbf{U}$ and we represent interruptibility as a property of activities by the function $int(A_i) \in [0, 1]$ $\forall A_i \in \mathbf{A}$. In our current implementation, we use common-sense knowledge to model the degree of interruptibility assigning a value to each activity using the following qualitative criteria:

$$int(A) = \begin{cases} 0, & \text{if } A \text{ is } \textit{never} \text{ interruptible} \\ 0.25, & \text{if } A \text{ is } \textit{rarely} \text{ interruptible} \\ 0.5, & \text{if } A \text{ is } \textit{on average} \text{ interruptible} \\ 0.75, & \text{if } A \text{ is } \textit{often} \text{ interruptible} \\ 1, & \text{if } A \text{ is } \textit{always} \text{ interruptible} \end{cases}$$

Clearly, since the system is considering to issue $Q^t(u^\star, U_t, A_i, A_j)$, it is unclear if the users are actually performing A_i or A_j. Hence, we must consider the minimum of the two interruptibility values in order to consider the worst-case scenario.

Finally, we want to balance the trade-off between interruptibility and the potential impact of the feedback to improve the recognition model. Hence, we prompt the query only when the following inequality is satisfied:

$$\min(int(A_i), int(A_j)) \cdot H(h(s(u^\star)^t)) \geq \sigma$$

where σ is an empirically defined threshold. A low σ would result in many queries that may remain unanswered, while a high σ would lead to a reduced number of queries and a consequent delayed improvement of the recognition model.

Once the query parameters have been selected and the decision on submission is taken, the query is sent to the ACTIVE LEARNING INTERFACE module.

2) Feedback semantic analysis: When a feedback is received from the interface, it must be mapped to an activity label in order to update the recognition model with a new labeled example. Depending on how the interface is designed, a feedback provided through the interface may not exactly match one of the labels used by the recognition model. For instance, considering a vocal interface, suppose that the following query is prompted to Bob: "What are you doing with Alice? Preparing a meal or Washing dishes?". Then, Bob provides the vocal feedback: "We are Cooking". Since the feedback does not exactly match any of the activity labels "Preparing meal" and "Washing dishes" (actually used by the recognition model), the FEEDBACK SEMANTIC ANALYSIS module is in charge of identifying the semantically closest activity label.

In our system prototype, this module is implemented using the method proposed in [19]. The approach consists of applying natural language processing algorithms to each vocal feedback f provided by the users. In particular, the WordNet ontology is used to compute the similarity $sim(f, A)$ between a feedback f and every activity $A \in \mathbf{A}$. The activity $A^\star = \text{argmax}_{A \in \mathbf{A}}\, sim(f, A)$ associated to the highest similarity is then used as label to update the recognition model only when none of the activities is sufficiently similar to f, or when two or more activities share the same highest similarity value. This is formalised by the following necessary conditions:

- $sim(f, A^\star) \geq \tau$
- $\exists! A \in \mathbf{A}$ s.t. $sim(f, A) = sim(f, A^\star)$

When any of the above conditions is not satisfied, different strategies can be considered including repeating one or more times the query.

E. Active Learning Interface

A major challenge in the envisioned system is to design an effective interface to prompt the queries to the users and to receive an answer. This involves identifying the type and location of interfaces, selecting appropriate media, selecting strategies for concurrent queries and answers timeouts. In this paper we focus on a system with a single wall mounted tablet interface since this was the setting for our prototype. Figure 2 shows a screenshot of our tablet interface.

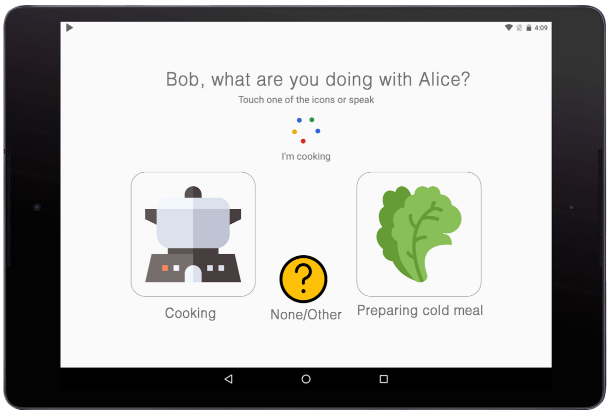
Bob, what are you doing with Alice?
Touch one of the icons or speak
I'm cooking
Cooking
None/Other
Preparing cold meal

Fig. 2. Our smart-home interface. The tablet runs a dedicated Android application which is asking Bob which activity is doing with Alice. Bob can answer using his voice or touching one of the graphics.

In Section V we discuss cases in which multiple and different interfaces may be available.

a) Prompting the query: When a query $Q^t(u^\star, U_t, A_i, A_j)$ from the QUERY BUILDER module is received, the VOICE AND TOUCH INTERFACE prompts a question in the form of: "$u^\star$, *what are you doing with* $U_t \setminus \{u^\star\}$? A_i *or* A_j?". For instance, given a query $Q^t(Bob, \{Bob, Alice\}, Eating, Relaxing)$, our interface will prompt the following query *"Bob, what are you doing with Alice? Eating or Relaxing?"*. While the question visually appears in the interface, it is also reproduced using *text-to-speech* tools. The users can interact with the interface using their voice (i.e., *speech-to-text*) or in a more traditional way (i.e., touch). While voice interaction allows users to answer without using hands (i.e., avoiding to abruptly interrupt their activities), they sometimes prefer to use touch interfaces to perceive more control on the system [12]. Moreover, touch interface is also suitable in those situations where the user can not talk for some reasons (e.g, she is at the phone).

b) Strategies for timely queries, answers timeout, and possibly concurrent queries: In order to robustly improve the recognition model, the query must be prompted during the activity, and the feedback is useful only if provided soon after the query. Since the query target user may not be available or simply ignoring the query, if no feedback is received within a timeout T, the interface should drop the query. If the feedback is received within T, it is forwarded to the FEEDBACK SEMANTIC ANALYSIS module.

Another important aspect to consider during the design of a multi-inhabitant active learning interface is how to handle concurrent queries. For example, at time t the QUERY BUILDER module may be uncertain both on the activity performed by Alice and Bob and on the activity performed by Carl and David, thus concurrently forwarding two queries $Q(Alice, \{Alice, Bob\}, Cooking, WashingDishes)$ and $Q(David, \{David, Carl\}, WatchingTV, Relaxing)$ to the same interface. For the sake of usability, we aim at prompting only one of the many concurrent queries. In these cases, our interface selects the query which maximizes the recognition model improvement, which is the one with the highest entropy for the target user. Continuing our example, if $H(h(s(Alice)^t) = 1.2$ and $H(h(s(David)^t)) = 0.8$, our system will prompt the first query.

V. ALTERNATIVE ACTIVE LEARNING INTERFACES

As we explained in Section IV, our prototype system uses a single wall-mounted tablet as prompting device. In the following, we briefly discuss different settings and alternative devices which can be used to prompt active learning queries, indicating their advantages and their limits.

A. Personal devices

Since our framework aims at selecting a specific user to prompt an active learning query, personal devices like smartphones and smartwatches are an alternative natural solution for the interface component. The main advantage of this choice is that concurrent queries to different target users can be prompted in parallel. However, studies confirm that notifications on personal devices are potentially very distracting with respect to other prompting modalities [12]. For instance, a user which receives a query on her smartphone located in her pocket should temporarily interrupt her current activity to extract the device from the pocket and provide a feedback to the system.

B. Single and multiple vocal assistants

Vocal assistants implemented in appliances like Google Home or Amazon Echo or hosted on tablets, should also be considered to prompt active learning queries. The vocal interface has the potential of requiring a significantly reduced amount of time in order to provide a feedback to the system. Indeed, users may even answer without interrupting their current activity. Tablets have the advantage of supporting both voice and touch interfaces.

The main drawback of having a single interface is that they should issue a query only if the target user is in the same room where the interface is located. Ideally, there should be an interface in each room of the smarthome, an approach that is already being considered for commercial vocal assistants and that can be economically more sustainable than the installation of multiple wall mounted tablets.

A drawback of this setting is that a single query can be prompted at each time in each room. Indeed, if multiple users are in the same room and the system issues concurrent queries, the interface has to decide which query to show (as we discussed in Section IV-E). An additional drawback of Internet based vocal assistants is a possible privacy threat, since the information about the activities being performed would be acquired by the external service.

C. Social robots

Social robots have the potential to revolutionize the smart-home interfaces of the future, especially for ambient assisted living applications for elderly subjects [20]. Considering our active learning setting, a social robot may directly interact with the query target user, navigating to the appropriate location. Clearly, this approach would combine the benefits of vocal interfaces and the ones of personal devices that typically move with the user. Among the drawbacks, social robots are currently very primitive, expensive, and their acceptability in home environments is still questionable [12]. Moreover, social robots would have the same problem of vocal assistants in handling concurrent queries, and in preserving privacy if they use external services.

VI. Conclusion and future work

In this paper, we studied the requirements and the main issues in the design of an active learning interface for multi-inhabitant activity recognition. We formalized the problem and we provided a description of the solutions adopted for our prototype.

In future work, we will quantitatively evaluate the effectiveness of our approach in a realistic scenario, considering real users performing activities of daily living in a home environment and exploring different interface modalities.

From the methodology point of view, we will also explore how to model our problem as a reinforcement learning task. In this case, the active learning system would be in charge of continuously learning a policy to decide whether to ask and which user to select.

References

[1] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 790–808, 2012.

[2] D. Riboni, C. Bettini, G. Civitarese, Z. H. Janjua, and R. Helaoui, "SmartFABER: Recognizing fine-grained abnormal behaviors for early detection of mild cognitive impairment," *Artificial Intelligence in Medicine*, vol. 67, pp. 57–74, 2016.

[3] D. J. Cook, A. S. Crandall, B. L. Thomas, and N. C. Krishnan, "Casas: A smart home in a box," *Computer*, vol. 46, no. 7, pp. 62–69, 2012.

[4] M. Stikic, K. Van Laerhoven, and B. Schiele, "Exploring semi-supervised and active learning for activity recognition," in *2008 12th IEEE International Symposium on Wearable Computers*. IEEE, 2008, pp. 81–88.

[5] E. Hoque and J. Stankovic, "Aalo: Activity recognition in smart homes using active learning in the presence of overlapped activities," in *2012 6th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*. IEEE, 2012, pp. 139–146.

[6] H. S. Hossain, M. A. A. H. Khan, and N. Roy, "Active learning enabled activity recognition," *Pervasive and Mobile Computing*, vol. 38, pp. 312–330, 2017.

[7] G. Civitarese, C. Bettini, T. Sztyler, D. Riboni, and H. Stuckenschmidt, "newnectar: Collaborative active learning for knowledge-based probabilistic activity recognition," *Pervasive and Mobile Computing*, vol. 56, pp. 88–105, 2019.

[8] P. Rashidi and A. Mihailidis, "A survey on ambient-assisted living tools for older adults," *IEEE journal of biomedical and health informatics*, vol. 17, no. 3, pp. 579–590, 2012.

[9] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, "Recognizing independent and joint activities among multiple residents in smart environments," *Journal of ambient intelligence and humanized computing*, vol. 1, no. 1, pp. 57–63, 2010.

[10] A. Benmansour, A. Bouchachia, and M. Feham, "Modeling interaction in multi-resident activities," *Neurocomputing*, vol. 230, pp. 133–142, 2017.

[11] F. Portet, M. Vacher, C. Golanski, C. Roux, and B. Meillon, "Design and evaluation of a smart home voice interface for the elderly: acceptability and objection aspects," *Personal and Ubiquitous Computing*, vol. 17, no. 1, pp. 127–144, 2013.

[12] M. Luria, G. Hoffman, and O. Zuckerman, "Comparing social robot, screen and voice interfaces for smart-home control," in *Proceedings of the 2017 CHI conference on human factors in computing systems*. ACM, 2017, pp. 580–628.

[13] S. Soda, M. Nakamura, S. Matsumoto, S. Izumi, H. Kawaguchi, and M. Yoshimoto, "Implementing virtual agent as an interface for smart home voice control," in *2012 19th Asia-Pacific Software Engineering Conference*, vol. 1. IEEE, 2012, pp. 342–345.

[14] G. López, L. Quesada, and L. A. Guerrero, "Alexa vs. siri vs. cortana vs. google assistant: a comparison of speech-based natural user interfaces," in *International Conference on Applied Human Factors and Ergonomics*. Springer, 2017, pp. 241–250.

[15] J. Fogarty, S. E. Hudson, C. G. Atkeson, D. Avrahami, J. Forlizzi, S. Kiesler, J. C. Lee, and J. Yang, "Predicting human interruptibility with sensors," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 12, no. 1, pp. 119–146, 2005.

[16] J. Cumin, F. Ramparany, J. L. Crowley *et al.*, "Inferring availability for communication in smart homes using context," in *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 2018, pp. 1–6.

[17] A. Benmansour, A. Bouchachia, and M. Feham, "Multioccupant activity recognition in pervasive smart home environments," *ACM Computing Surveys (CSUR)*, vol. 48, no. 3, p. 34, 2016.

[18] B. Settles, "Active learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009.

[19] S. Fernando and M. Stevenson, "A semantic similarity approach to paraphrase detection," in *Proceedings of the 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics*, 2008, pp. 45–52.

[20] J. Broekens, M. Heerink, H. Rosendal *et al.*, "Assistive social robots in elderly care: a review," *Gerontechnology*, vol. 8, no. 2, pp. 94–103, 2009.