NECTAR: Knowledge-based Collaborative Active Learning for Activity Recognition

Gabriele Civitarese*, Claudio Bettini*, Timo Sztyler†, Daniele Riboni‡, Heiner Stuckenschmidt†

*University of Milano
Milano, Italy
{gabriele.civitarese, claudio.bettini}@unimi.it

†University of Mannheim
Mannheim, Germany
{timo, heiner}@informatik.uni-mannheim.de

‡University of Cagliari
Cagliari, Italy
riboni@unica.it

Abstract—**Due to the emerging popularity of pervasive healthcare applications, tools for monitoring activities in smart homes are gaining momentum. Existing methods mainly rely on supervised learning algorithms for recognizing activities based on sensor data. A key issue with those approaches is the acquisition of comprehensive training sets of activities. Indeed, that task incurs significant costs in terms of manual labeling effort; moreover, labeling by external observers violates the individual's privacy. For these reasons, there is an increasing interest in unsupervised activity recognition methods. A popular approach relies on knowledge-based models expressed by ontologies of activities, environment and sensors. Unfortunately, those models require significant knowledge engineering efforts, and are often limited to a specific application. In this paper, we address the issues of existing methods by proposing a novel hybrid approach. Our intuition is that a generic knowledge-based model of activities can be refined to target specific individuals and environments by collaboratively acquiring feedback from inhabitants. Specifically, we propose a collaborative active learning method to refine correlations among sensor events and activity types that are initially extracted from a high-level ontology. Generic correlations are personalized to each target smart-home considering the similarity between the feedback target and the feedback provider in terms of environment and inhabitant's profiles. Moreover, thanks to this method, new sensors installed in the home are seamlessly integrated in the recognition framework. In order to reduce the burden of providing feedback, we also propose a technique to carefully select the conditions that trigger a feedback request. We conducted experiments with a real-world dataset and a generic ontology of activities. Results show that our hybrid method outperforms state-of-the-art supervised and unsupervised activity recognition techniques while triggering an acceptable number of feedback queries.**

I. Introduction

Human activity recognition has been deeply investigated in the last decade taking advantage of the effective sensing infrastructure that is becoming available with off-the-shelf products as part of domotics, smart objects, personal and wearable devices. Among the many applications in mobile and pervasive computing, the continuous recognition of activities of daily living (ADL) has been identified as a key enabler of assisted living and e-health systems [1], [2].

Most ADL recognition systems rely on supervised learning which requires a large training set of sensor events annotated with activity labels (usually mapping sequences of events to the specific activity that generated them). Besides privacy issues in observing individuals in their private spaces, the heterogeneity of the environments and of the way a complex activity can be performed by different individuals limits the value of the acquired training set. This is a serious limitation to the large scale deployment of these systems. Active learning has been proposed to mitigate this problem, but the majority of these techniques need anyway a starting labeled training set. Alternative approaches propose the use of a structured knowledge representation of activities, infrastructure, and events to guide the recognition process in an unsupervised way [3]. In order to be effective, they require a significant effort of knowledge engineers to build a comprehensive ontology, and it remains questionable if such an ontology could actually cover an heterogeneous large set of environments and individuals.

We address the problem explained above by a novel framework, called *NECTAR*, exploiting *kNowledge-basEd Collaborative acTive learning for Activity Recognition*. NECTAR does not require an initial training set, since it relies on a (possibly incomplete) ontology to derive a first set of semantic correlation values between sensor events and activities. These correlations, together with the stream of sensor events, are processed by a Markov Logic Network to identify ADLs through probabilistic reasoning. In order to cope with the incompleteness of the ontology and the heterogeneity of environments and individuals, we introduce a collaborative active learning process to refine the correlations derived by the ontology. The stream of sensor events is segmented in real-time, and based on the discrimination value of correlations on the segment, a feedback may be asked to the subject about which activity is being performed. Feedback responses coming from different homes are collected in a cloud infrastructure and each home receives personalized information to refine its recognition model. The collaborative active learning feature of NECTAR also deals with the common situation in which a new device is installed in the infrastructure, by producing a new set of correlation values regarding the new device events.

NECTAR has been experimentally validated on a real world and publicly available dataset involving multiple subjects performing ADLs in both sequential and interleaved fashion.

The main contributions can be summarized as follows:

- We propose a new active learning approach to ADL recognition that addresses the main problems of current statistical and knowledge-based methods;
- The NECTAR technique supports collaborative and personalized refinement of the recognition model;
- Our experiments show the gain obtained by collaborative active learning, the moderate effort required to the involved subjects, and the overall effectiveness of NECTAR even compared with supervised approaches.

The rest of the paper is structured as follows. Section II summarizes related work. Section III presents the NECTAR architecture, while the techniques are explained in Section IV. Section V shows experimental results and Section VI discusses open issues. Finally, Section VII concludes the paper.

II. Related Work

As explained in the introduction, acquiring comprehensive training sets of ADLs is expensive in terms of annotation costs and may violate the individuals' privacy [4]. Hence, several efforts have been devoted to devise unsupervised or semi-supervised activity recognition techniques [5]–[7], as well as transfer learning methods for activity models [8], [9].

Unsupervised activity recognition methods do not require the acquisition of labeled training sets. As a consequence, the model of activities must be either manually specified (e.g., through an ontology) or mined from other sources (e.g., Web resources, or unlabeled datasets of activities). A popular approach to the manual specification of activity models consists in the use of description logics or logical rules to define the formal semantics of activities [5]–[7]. In those approaches, complex activities are defined in terms of their simpler components. Sequences of simple actions, recognized based on firing of specific sensor events, are matched to activity definitions to identify the occurred activity. However, those approaches rely on rigid assumptions about the execution patterns of ADLs [10]. On the contrary, complex activities are characterized by large variability of execution. In order to cope with that issue, other works investigated the use of less rigid formalisms to define ADLs. In [11], probabilistic description logics are used to define a multi-level ontology of domestic activities. Hybrid ontological-probabilistic reasoning is used in [3] to recognize ADLs based on semantic correlations among sensor events and activities. However, all those approaches require significant knowledge engineering efforts, and are hardly scalable to the definition of a comprehensive set of ADLs in different contexts.

In order to avoid the burden of manual specification, different techniques have been proposed to mine activity models from Web resources. A first attempt in this sense was due to Perkowitz et al. in [12] and refined in later works [13], [14]. Those methods analyze textual descriptions of activities mined from the Web in order to obtain correlations among activities and objects used for their execution. Those correlations are used to recognize the executed activity based on the observed sequence of used objects. That approach has been recently extended to exploit visual cues extracted from the Web, such as images and videos [15]. However, it is questionable whether object-activity correlations are sufficient to recognize complex ADLs. Indeed, in this work we rely not only on correlations, but also on knowledge-based constraints about activity components.

A further approach is to infer activity models from unlabeled datasets. An approach for activity discovery on data acquired from body-worn sensors is presented in [16]. In that work, multi signal motifs mining is used to recognize approximately repeated subsequences of sensor data, that represent the execution of a specific (unknown) activity. A similar approach, but applied to domestic sensors, is proposed in [17]. In [18], data mining methods are used to cluster sequences of sensor events, such that each cluster represents an activity class. The inhabitant is asked to provide the actual class of each cluster. In USMART [19], a knowledge-based method is used to compute similarity among pairs of sensor events based on their temporal, spatial, and usage dimensions. Objects similarity is used to segment sensor event traces that should represent the execution pattern of a single activity. Sequential pattern mining is used to identify frequent sequences of sensor events that typically appear during an activity. Exploiting an ontology of activities and objects, each frequent sequence is associated to one or more activities, according to the objects that triggered the sensor events in the sequence. Sequences are refined thanks to a clustering algorithm, and refined sequences are used for activity recognition. Compared to those unsupervised methods, in our work we exploit collaborative users' feedback to assign a certain semantics to sets of sensor events.

Semi-supervised learning methods use unlabeled data to improve the model computed through a training set. Different semi-supervised methods, including the ones presented in [20]–[22], address the recognition of physical activities based on accelerometer data. Self-training and co-training is used in [23] for recognition of ADLs. The same work also investigates the use of active learning, with the objective of identifying the most informative sequences of sensor events for which to query the individual. A sequence is considered informative either when the confidence of the classifier about its predicted activity class is low, or when two classifiers disagree about its class. In [24], the authors propose to use active learning to dynamically adapt the recognition model to the changes of the home environment. In that work, an entropy based measurement is used to query the most informative sequences of sensor events to update a Dynamic Bayesian Network. An active learning method to iteratively refine the annotations of video provided by crowdsourcing services (like Mechanical Turk) is presented in [25]. That method relies on confidence scores about the annotation. Annotations with low confidence are submitted again to the crowdsourcing service for revision. A similar approach is proposed in [26]. In that work, privacy of individuals depicted in videos is protected by automatically identifying people and veil them by coloring their silhouette. Work presented in [25] proposes three techniques to choose the most informative data points for which to query the user. Methods are based on (i) low confidence for the most probable activity class, (ii) small difference between the confidence of the most and second most probable class, or (iii) high entropy among the probability of classes. Experimental results in smart home settings show that the three methods achieve similar accuracy. The work presented in [27] proposes strategies to select the most appropriate annotators in a crowdsourcing framework for active learning of ADLs. Differently from those works, in this paper we propose *collaborative* active learning to share the burden of providing ADLs labels among a community of inhabitants.

III. NECTAR's architecture

We assume a set of different smart-homes equipped with unobtrusive sensing infrastructures. Environmental sensors are deployed in each home in order to monitor the interaction of the inhabitant with home artifacts but also context conditions (e.g., temperature) and presence in certain locations. Each home may have a different set of deployed sensors and monitored items. A gateway in the home is in charge of collecting and pre-processing raw data from the sensor network in order to reconstruct the most probable activities that generated them. In this context, we distinguish between an *activity class* and an *activity instance* where the former is an abstract activity, e.g., taking medicine, and the latter the actual occurrence of an activity of a given class during a certain time period. We denote with $\mathbf{A}$ the set of the considered activity classes (e.g., $\mathbf{A} = \{$*Eating, Cooking, Taking Medicines*$\}$).

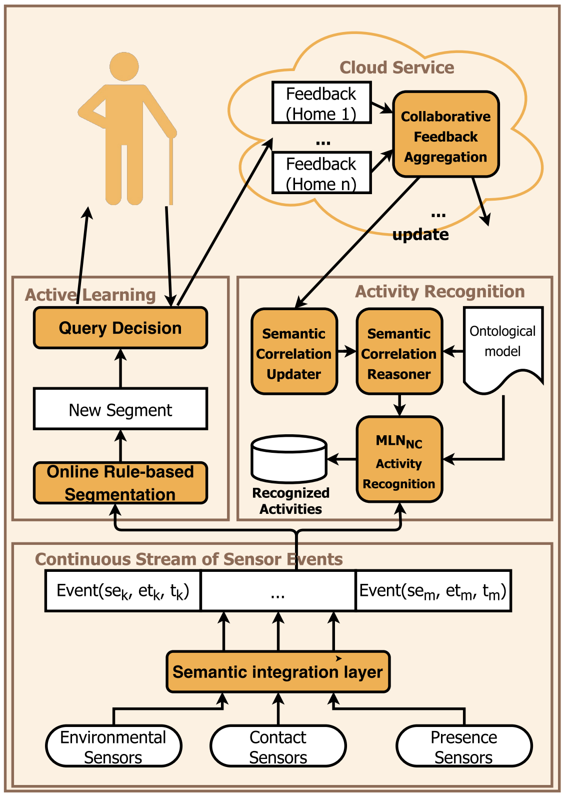


Fig. 1. The general architecture of NECTAR

The general architecture of our system is shown in Figure 1. The SEMANTIC INTEGRATION LAYER is in charge of applying pre-processing rules in order to detect high-level events from raw sensor signals. For example, if at time t the medicine drawer sensor produces the raw event *open*, then the high-level event at t is *opening the medicine drawer*. We denote with $\mathbf{E}$ the set of high-level event types that correspond to the set of monitored operations (e.g., $\mathbf{E} = \{$ *opening_the_medicine_drawer*, *closing_the_medicine_drawer* $\}$). In addition, $\mathbf{T}$ describes the set of all possible event timestamps. A temporal sequence of events is represented as:

$$\langle\, se_1 = \langle et_1, t_1 \rangle, \ldots, se_k = \langle et_k, t_k \rangle \,\rangle,$$

where $se_i = \langle et_i, t_i \rangle$ indicates that se_i is an instance of the event type $et_i \in \mathbf{E}$ occurred at timestamp $t_i \in \mathbf{T}$.

Reasoning on high-level events allows our system to be robust to minor changes in activity execution. For instance, if the inhabitant decides to keep his/her medicines in a different drawer than before, it is sufficient to adjust the above mentioned pre-processing rules, thus adapting the mapping between raw sensors data and events.

Like in our previous work [3], we define the semantics of activities and high-level events in an OWL 2 ontology. In particular, the ontology includes axioms stating that an instance of a given activity class (e.g., "prepare soup") must necessarily generate a set of high-level events (e.g., "take water", "pour water"). Moreover, it also includes other common-sense axioms regarding time and location; e.g., "every instance of cooking is executed in the kitchen". Ontological reasoning is used to derive probabilistic dependencies among sensor event types and classes of executed activities; we denote them as *semantic correlations*. Periodically (e.g., daily), the MLN_{NC} ACTIVITY RECOGNITION layer is in charge of recognizing performed activities by modeling and reasoning with detected events and semantic correlations through an extension of Markov Logic Networks with numerical constraints [29].

However, being manually designed by knowledge engineers with a specific application in mind, the ontological model is necessarily limited to specific environments and activities. Thus, semantic correlations may not be sufficiently comprehensive to cover different application domains. Moreover, some sensor event types (e.g., motion sensors) do not convey any explicit semantic information; hence, no semantic correlation can be inferred for these event types from the ontology. For this reason, our system collects *feedback items* from the smart-homes in order to discover semantic correlations not inferred from the ontology. For acquiring a feedback, the system interactively queries the user to provide the class of his/her current activity. Acquired feedback is collaboratively shared among the smart-homes to update semantic correlation values in a personalized fashion. For the sake of clarity, in the following we name *origin* the environment (home and inhabitant) providing feedback, and *target* the environment where feedback is used to update semantic correlations.

The feedback acquisition mechanism relies on the concept of *segments*. Formally, a segment $\vec{s} = \langle se_j, \ldots, se_k \rangle$ represents a vector of consecutive sensor events. Each event is assigned to exactly one segment. In order to cope with interleaved activities, a single activity instance can span multiple segments. Each segment belongs to exactly one activity instance. The class of each segment is the one of its activity instance. As explained below, when the system determines that a segment's events do not provide enough hints to reliably determine its class according to an information-theoretic metric, it queries the user to obtain a feedback.

To this purpose, the ONLINE RULE-BASED SEGMENTATION layer is in charge of segmenting the continuous stream of sensor events. The segmentation method is based on semantic rules that consider different aspects like time constraints, objects interaction, and change of location. The role of these rules is to group together those consecutive events which most likely originate from the same activity instance. As soon as a segment is finalized, it is processed by the QUERY DECISION layer in order to decide whether triggering a feedback query or not. That module processes the segment to apply an information-theoretic metric considering the segment's events and the semantic correlations. If the activity class is uncertain according to that metric, the module triggers a feedback query. A user-friendly and unobtrusive interface is in charge of issuing the feedback query and collecting the inhabitant answer.

The acquired *feedback* is transmitted to a CLOUD SERVICE, where the COLLABORATIVE FEEDBACK AGGREGATION layer is in charge of computing personalized feedback items for the different environments. Personalization is based on the similarity between the origin and target environment. The CLOUD SERVICE periodically sends personalized feedback items to each target.

Received feedback is used by the SEMANTIC CORRELATIONS UPDATER layer to discover novel semantic correlations and to update the values of existing ones.

For the sake of this work, we assume that the CLOUD SERVICE is trusted. However, in a real deployment it would likely be a *honest-but-curious* third party. Proper privacy techniques are thus needed to protect sensitive data and at the same time to preserve the CLOUD SERVICE functionalities.

IV. NECTAR UNDER THE HOOD

In this section, we describe in detail each component of our system.

A. *Activity recognition*

Our activity recognition system relies on an ontological model of home environment, sensors, activities and actors, and on probabilistic reasoning through an extension of Markov Logic Networks (MLN) [30].

A.I Ontological model:

Our ontology has been defined using the OWL 2 language. For the sake of space, we omit technical details about the ontological model and ontological reasoning methods, which can be found in [3]. In the following, we outline the main characteristics of our ontology. Ontological properties describe relations among instances. For example, an ontological axiom states that "firing of an accelerometer a attached to a kitchen chair c indicates the occurrence of an action of class *MoveKitchenChair*". In our ontology, we express necessary conditions for a set of sensor events to be generated by a given activity. For example, the sensor events generated by an instance of activity class *PrepareHotMeal* must include an event of class *UsingCookingInstrument*. Other ontological axioms describe temporal and location-based constraints. As explained below, by reasoning with those ontological axioms, we can infer generic semantic correlations among sensor event types and activity classes, which are used by the MLN module to associate sensor events to their activity instance.

A.II Semantic correlation reasoner:

Intuitively, given a certain sensor event se of type et, the semantic correlation function $OSC(et, ac)$ represents the probability of ac being the class of the activity instance that generated se. Hence, given et, we have that $OSC(et, ac)$ is a probability distribution over all ac values:

$$\forall\, et \in \mathbf{E} \sum_{ac \in \mathbf{A}} OSC(et, ac) = 1. \tag{1}$$

Inference of semantic correlations relies on the property composition operator of OWL 2 [10]. In particular, in the ontology, we defined an axiom stating that: "if an event of type et is produced by a sensor that indicates the usage of an artifact possibly used for an activity of class ac, then et is a *predictive sensor event type* for ac". For each event type et, we compute the set of activities for which et is a predictive event type:

$$predAct(et) = \{ac \,|\, et \text{ is a predictive event for } ac\}$$

To enforce property (1), we set the values of the semantic correlation function OSC for each combination of event type et and activity class ac in the following way. We consider two cases. If et is predictive of at least one activity class, we compute et's correlations using the following formula:

$$OSC(et, ac) = \begin{cases} \frac{1}{|predAct(et)|} & \text{if } ac \in predAct(et) \\ 0 & \text{otherwise} \end{cases}$$

Otherwise, having no information about the associations between et and activity classes, we uniformly distribute its correlation values to all possible activity classes:

$$OSC(et, ac) = \frac{1}{|\mathbf{A}|},$$

where $\mathbf{A}$ is the set of all activity classes. It is easy to verify that in both cases property (1) is enforced.

A.III MLN activity recognition:

MLN is a probabilistic first-order logic that naturally supports reasoning with uncertain axioms and facts [30]. In our framework, we use an extension of MLN, named MLN$_{NC}$, which supports numerical constraints useful to reason with temporal information. In the following, we sketch the main inference tasks for activity recognition; more details can be found in [3]. MLN supports the definition of both hard and soft axioms. The former are certainly true, and are mainly automatically extracted from our ontology. The latter are associated to a weight that represents their probability of being true, considering the inferred semantic correlations. We instantiate the MLN knowledge base by translating the axioms of our ontology in hard MLN axioms. In this way, we ensure that the MLN_{NC} knowledge base is consistent with our OWL 2 ontology. Moreover, we add soft MLN axioms to represent common-sense knowledge about typical activity execution. In particular, the extension of MLN with numerical constraints allows us to express probabilistic common-sense knowledge about the typical locations in which activities are executed. Similarly, we define soft axioms about the maximum and minimum duration of activities. At activity recognition time, we add facts that represent the observation of occurred sensor events. Each fact $SensEv(et, ts)$ includes the corresponding event type et and timestamp ts. We also add facts regarding initial hypothesis of activity instances, which are computed by a heuristic algorithm considering semantic correlations. Each fact $ActClass(ai, ac)$ includes the corresponding candidate activity instance identifier ai and its most likely activity class ac. Finally, we add probabilistic axioms that relate sensor events to candidate activity instances according to the semantic correlation values of the corresponding event type and activity class. For instance, the probabilistic axiom:

$$-0.619 \; BelongTo(\text{`}\textit{PourWater}\text{'}, 1029, ai) \\ \wedge ActClass(ai, \text{`}\textit{PrepareSoup}\text{'})$$

states that, with weight -0.619, the sensor event of type 'PourWater' observed at timestamp 1029 belongs to an activity instance ai of class 'PrepareSoup'. According to the MLN semantics of weights, the axiom weight is computed applying the *logit* function to the corresponding probability value obtained by the semantic correlation function. For instance, the value -0.619 in the above formula is obtained by applying *logit* to 0.35, which is the value of $OSC(\text{`}\textit{PourWater}\text{'}, \text{`}\textit{PrepareSoup}\text{'})$. Finally, maximum a-posteriori inference is used to compute the most probable assignment of (i) sensor events to activity instances, and (ii) activity class to activity instances.

B. Online Segmentation and Query Decision

The segmentation of the continuous stream of sensor data is performed by the ONLINE RULE-BASED SEGMENTATION layer based on knowledge-based conditions. For each produced segment, the QUERY DECISION layer thus computes its entropy considering the semantic correlations. If the entropy of a segment exceeds a fixed threshold, the inhabitant is queried in order to provide an activity class to the segment. This allows us to limit the number of queries issued to the inhabitant.

B.I Online rule-based segmentation:

The ONLINE RULE-BASED SEGMENTATION layer is in charge of deciding whether it is appropriate to finalize the current segment and initiate a new one. For that purpose, it considers semantic conditions in order to interpret the continuous stream of sensor events. This includes the observation of interactions with objects (*C1*), changes between rooms (*C2*), and unusual gaps in time between consecutive sensor events (*C3*). Whenever a new sensor event e_{new} is observed, all these three conditions are reviewed. If at least one of the conditions is fulfilled the current segment is finalized. Hence, the sensor event e_{new} is the first element of the new segment. Of course, the objective is to reconstruct the ground truth segments based on the observed stream of sensor events. In the following, we describe the mentioned conditions in detail:

C1) This condition keeps track of the events that result from interaction with the objects in the home. If at a certain point in time the subject stops to interact with all items, we consider this as an indicator that an activity instance was completed and thus the current segment is finalized.

C2) Several activities are bound to a certain location or room. For that reason, if sensor events show that the inhabitant moves from a room to another, C2 considers this situation as an indicator to finalize the current segment.

C3) An increasing temporal distance between consecutive sensor events can be interpreted as a reduced probability that they describe the same activity instance. This is especially the case if the subject leaves the observed home. Therefore, we keep track of the median distance between sensor events collected in the previous days, and we assume that two consecutive sensor events belong to different segments if their temporal distance is twice as large as the median. When no sufficient statistical information about previous sensor events is available, we use a manually fixed threshold.

These conditions aim to generate segments which cover at most one activity instance: we prefer to split an activity instance in more segments instead of trying to build a segment that perfectly fits an activity instance, as this would also increase the risk of including unrelated sensor events. Indeed, we want to reduce the risk of associating the user's answer with wrong sensor events. Moreover, this segmentation strategy allows us to cope with interleaved activities.

As soon as a new segment is generated, it is forwarded to and processed by the QUERY DECISION layer.

B.II Query decision:

Given a segment S, the QUERY DECISION layer decides if it is necessary to query the inhabitant. In particular, if the *semantic correlations* of the types of the events in S are inconclusive when considered together (i.e., they do not converge on a specific activity class), we ask the inhabitant which activity he/she was actually performing. For that purpose, we introduce the concept of a *segment's bag*:

$$Bag(S) = \{et \mid se = \langle et, t\rangle \in S\}$$

where S is a finalized segment and $Bag(S)$ is a bag (i.e., a multiset) which contains the types of the events contained in S. It is important to note that the temporal order of events of a segment is not reflected by its bag. Hence, for each bag $Bag(S_i)$, we compute for all the activity classes $ac \in \mathbf{A}$ the likelihood that the segment S_i represents an activity instance of ac. This is computed as follows:

$$L(ac \mid S) = \frac{\sum_{et \in Bag(S)} OSC(et, ac)}{|Bag(S)|}$$

where $OSC(et, ac)$ is the *semantic correlation* between et and ac.

After we compute $L(ac|S)$ for all activity classes, we normalize these values in order to have a probability distribution. Subsequently, the *entropy* is calculated on the distribution to determine the system's confidence for the segment S:

$$H(S) = \sum_{ac \in A} P(X = ac \mid S) \cdot log(\frac{1}{P(X = ac \mid S)})$$

where $P(X = ac \mid S)$ results from the normalized $L(ac \mid S)$ values.

Finally, if $H(S)$ is higher than a predefined threshold λ, the system ranks S as *uncertain*. In this case, the system queries the inhabitant in order to provide an activity label ac for S, and each event type $et \in Bag(S)$ is associated with ac. These associations are transmitted in real-time to the CLOUD SERVICE together with the identification of the origin.

Note that segments containing noisy events which occurred outside activities execution (e.g., trigger of presence sensors) would likely lead to high entropy values. To overcome this issue, we rely on the SEMANTIC INTEGRATION LAYER presented in Section III to reduce as much as possible the generation of those noisy events. Moreover, we also discard segments with few events in order to further reduce noisy data.

C. Collaborative adaptation

In the following, we describe our collaborative adaptation framework, which relies on two main components. The COLLABORATIVE FEEDBACK AGGREGATION layer (which runs on the CLOUD SERVICE) collects and aggregates the *feedback* received from the several homes and it periodically transmits personalized updates to each target home. On the other hand, the SEMANTIC CORRELATION UPDATER algorithm (which runs in the home's gateway) is in charge of analyzing the personalized update in order to improve the semantic correlations produced by the ontology.

C.I Collaborative Feedback Aggregation:

The CLOUD SERVICE continuously receives and stores feedback transmitted by the participating homes. Each feedback item f is represented by a vector $f = \langle et, ac, o\rangle$, where et is an event type, ac is an activity class, and o is the origin of the feedback.

Based on the received feedback, the CLOUD SERVICE periodically transmits *personalized feedback items* to each target home. A personalized feedback item is represented by a vector $\langle et, ac, p, s\rangle$, where $p \in (0, 1]$ is the *predictiveness* of event type et for activity class ac computed based on feedback items, and $s \in (0, 1]$ is the *estimated similarity* between the feedback origins and target.

The COLLABORATIVE FEEDBACK AGGREGATION layer is in charge of computing personalized feedback items based on the received feedback. In order to measure the similarity between the origin and target of a feedback, that module relies on a similarity function $sim : H \times O \rightarrow [0, 1]$, where H is the set of targets, and O is the set of origin environments. The output of $sim(h, o)$ is a value between 0 and 1. Of course, the most appropriate definition of the target environment features, as well as the method to compute sim values, depend on the addressed application.

Based on a multiset F of feedback items, the module computes personalized feedback items for each target environment. In particular, consider a target h. At first, for each event type et and activity class ac, the following formula computes the personalized feedback *support*:

$$supp(et, ac, h, F) = \sum_{f = \langle et, ac, o\rangle \in F} sim(h, o).$$

In order to exclude unreliable feedback, the CLOUD SERVICE transmits only personalized feedback whose support is larger than a threshold σ. For each reliable personalized feedback, the module computes its *predictiveness* value:

$$pred(et, ac, h, F) = \frac{supp(et, ac, h, F)}{\sum_{ac_i \in \mathbf{A}} supp(et, ac_i, h, F)},$$

which is the normalization of et's support values, distributed over all the activity classes.

Finally, the module computes the *estimated similarity* as the median value of the similarity between the feedback items' origin and the target:

$$s(et, ac, h, F) = \underset{f = \langle et, ac, o\rangle \in F}{median} sim(h, o).$$

C.II Semantic Correlation Updater:

Periodically, each home receives an update from the CLOUD SERVICE consisting of a set $\mathbf{P}$ of *personalized feedback items*. The SEMANTIC CORRELATION UPDATER algorithm analyzes $\mathbf{P}$ along with the semantic correlations inferred by the ontology in order to refine the semantic correlations. We denote $OSC(et, ac)$ as the semantic correlation between et and ac computed by the ontology, while $SC(et, ac)$ the one computed by our algorithm.

The pseudo-code of the SEMANTIC CORRELATION UPDATER algorithm is shown in Algorithm 1. At first, the algorithm initializes the current semantic correlations with the ones computed by the ontology. Then it initializes the set U of *unpredictive event types*:

$$U = \{et \mid predAct(et) = \emptyset\}$$

U contains all the event types which the current ontology does not consider predictive for any activity. Then, the algorithm iterates on each *personalized feedback item* $\langle et, ac, p, s\rangle$

Algorithm 1: Semantic correlation updater

```
Input: A set of personalized feedback items
  $P = \{\langle et_1, ac_1, p_1, s_1\rangle, \langle et_2, ac_2, p_2, s_2\rangle, \dots\}$, semantic correlation
  function $OSC$ computed by the ontology, and set $U$ of unpredictive events
Output: Refined semantic correlation function $SC$
1: $SC \leftarrow OSC$
2: $newevents \leftarrow \emptyset$
3: for each $\langle et, ac, c, s\rangle \in \mathbf{P}$ do
4:     if $et \in U$ then
5:         $SC(et, ac) \leftarrow c$
6:         if $et \notin newevents$ then
7:             $newevents \leftarrow newevents \cup \{et\}$
8:             for each $ac_i \in \mathbf{A}$ s.t. $ac_i \neq ac$ do
9:                 $SC(et, ac_i) \leftarrow 0$
10:            end for
11:        end if
12:    else if $OSC(et, ac) = 0$ then
13:        $ac_{ont} \leftarrow$ an activity $ac_j \in \mathbf{A}$ s.t. $OSC(et, ac_j) > 0$
14:        $SC(et, ac_{ont}) \leftarrow \frac{SC(et, ac_{ont})}{1 + s \cdot SC(et, ac_{ont})}$
15:        $SC(et, ac) \leftarrow s \cdot SC(et, ac_{ont})$
16:        for each $ac_i \in \mathbf{A}$ do
17:            if $ac_i \neq a_{ont}$ and $ac_i \neq ac$ then
18:                $SC(et, ac_i) \leftarrow SC(et, ac_i) \cdot (1 - SC(et, ac))$
19:            end if
20:        end for
21:    end if
22: end for
23: return $SC$
```

contained in $\mathbf{P}$ in order to update the semantic correlations produced by the ontology. If et belongs to U, $SC(et, ac)$ is set to its predictiveness value p. Moreover, if et is observed for the first time during the current iteration (i.e., if it is not yet part of the set *newevents*), the semantic correlation value $SC(et, ac_i)$ for any other activity class $ac_i \neq ac$ is initialized to 0, and et is added to the set of new events. Intuitively, since *unpredictive event types* have uniform semantic correlations for all the activities, they are usually queried more than other event types since they contribute most in increasing the entropy value. This makes the *predictiveness* values provided by the CLOUD SERVICE reliable to be used as semantic correlations for et, thus overriding the uniform semantic correlations inferred by the ontology.

In the case of $et \notin U$, we update the semantic correlations only if $SC(et, ac)$ is 0. Indeed, our algorithm does not modify the non-zero semantic correlations inferred by the ontology, since they are considered reliable. Instead, whenever a new semantic correlation between et and ac is discovered from a *personalized feedback item*, it is necessary to correspondingly scale all the other semantic correlations regarding et so that $SC(et, ac)$ remains a distribution probability (i.e., $\sum_{ac \in \mathbf{A}} SC(et, ac) = 1$).

Hence, we select a random activity ac_{ont} correlated to et according to the ontology (i.e., such that $OSC(et, ac_{ont}) > 0$). Then we scale $SC(et, ac_{ont})$ considering the *estimated similarity* value s:

$$SC(et, ac_{ont}) := \frac{SC(et, ac_{ont})}{1 + s \cdot SC(et, ac_{ont})}$$

Since the event types for which the ontology already provided a semantic correlation are generally less queried, it is not reliable to use the predictiveness value to update the semantic correlations. This is why we use the *estimated similarity* s instead. The next step consists in updating $SC(et, ac)$:

$$SC(ac, et) := s \cdot SC(et, ac_{ont})$$

Finally, we update the semantic correlations of all the remaining activities ac_j (such that $ac_j \neq ac_{ont}$ and $ac_j \neq ac$) in the following way:

$$SC(et, ac_j) := SC(et, ac_j) \cdot (1 - SC(et, ac)).$$

It can be easily verified that, by construction, Algorithm 1 enforces property (1) introduced in Section III; i.e., given an event type et, the revised $SC(et, ac)$ function is a probability distribution over all ac values.

After each update, the function $SC(et, ac)$ computed by our algorithm thus replaces $OSC(et, ac)$ for both the QUERY DECISION and MLN_{NC} ACTIVITY RECOGNITION layers.

V. Experimental Evaluation

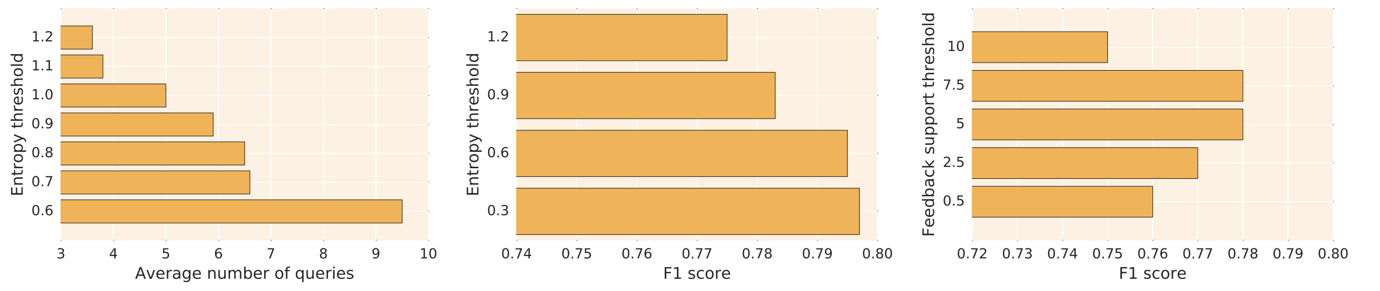
In order to evaluate our system, we use the well-known CASAS dataset [31], [32]. This dataset includes eight high-level ADLs performed by 21 subjects in a smart-home. Several sensors were deployed to monitor movements, use of water and interaction with objects, doors, and drawers. During the data collection, one subject at a time was present in the smart-home environment. Each subject was instructed to perform the following ADLs: *fill medication dispenser* (ac_1), *watch DVD* (ac_2), *water plants* (ac_3), *answer the phone* (ac_4), *prepare birthday card* (ac_5), *prepare soup* (ac_6), *clean* (ac_7), and *choose outfit* (ac_8). The activities were performed both in sequential and interleaved fashion, and their execution time and order were up to the subject. Due to limited space, we refer the reader to the original publication concerning the floor plan of the flat, the sensor positions, and a more detailed description of the activities [32].

We used the CASAS dataset to simulate 21 apartments with identical sensing infrastructures but inhabited by different subjects. This setup resembles the one of a residence for elderly people consisting of several similar apartments. We fixed the similarity $sim(h_1, h_2)$ between each pair of apartments to 0.5, since the sensing infrastructures are identical (i.e., their similarity is 1), while the profiling of the subjects is unknown.

During a pre-processing phase, we removed from the dataset the occurrences of those motion sensors which we found out to be *noisy*; i.e., producing measurements essentially independent from the performed activities. Most noisy motion sensors were those placed in locations irrelevant for the activity recognition task. Other ones triggered too many events, possibly due to excessively high sensitivity or too wide coverage area. Hence, we kept motion sensor events from 7 devices only[1].

We performed *leave-one-subject-out* cross validation. In each fold, NECTAR collects feedback items from 20 subjects and uses them to update semantic correlations for the remaining one. Table I summarizes our overall experimental results.

[1]Those sensors are identified as M02, M03, M04, M05, M13, M23, and M24 in the dataset.



(a) How entropy affects the number of queries (b) How entropy affects F_1 score (c) How feedback support threshold affects F_1 score

Fig. 2. The impact of entropy and feedback support thresholds on the number of issued feedback queries and average recognition rates of NECTAR.

TABLE I
RESULTS ($\mathbf{F_1}$ SCORE) OF THE PROPOSED ACTIVITY RECOGNITION METHOD COMPARED TO RELATED WORK FOR INTERLEAVED ACTIVITIES.

Activity	**Supervised machine learning** [33]	**Unsupervised probabilistic reasoning** [15]	**NECTAR without active learning**	**NECTAR with active learning**
ac_1	0.80	0.74	0.78	**0.82**
ac_2	**0.87**	0.84	0.85	**0.87**
ac_3	0.59	0.36	0.70	**0.71**
ac_4	0.52	0.49	0.67	**0.72**
ac_5	**0.88**	0.83	0.77	0.78
ac_6	0.85	0.67	**0.89**	**0.89**
ac_7	0.57	0.36	0.46	**0.63**
ac_8	**0.84**	0.69	0.71	0.82
$avg.$	0.74	0.70	0.73	**0.78**

TABLE II
RESULTS (F_1 SCORE) OF NECTAR WITH VARYING ENTROPY THRESHOLD.

Class	**Entropy threshold λ**			
	0.3	**0.6**	**0.9**	**1.2**
ac_1	0.819	0.813	0.824	0.811
ac_2	0.875	0.874	0.869	0.876
ac_3	0.743	0.739	0.709	0.730
ac_4	0.719	0.724	0.724	0.724
ac_5	0.813	0.807	0.784	0.780
ac_6	0.896	0.894	0.887	0.886
ac_7	0.659	0.645	0.633	0.629
ac_8	0.859	0.863	0.824	0.774
$avg.$	0.798	0.795	0.782	0.776

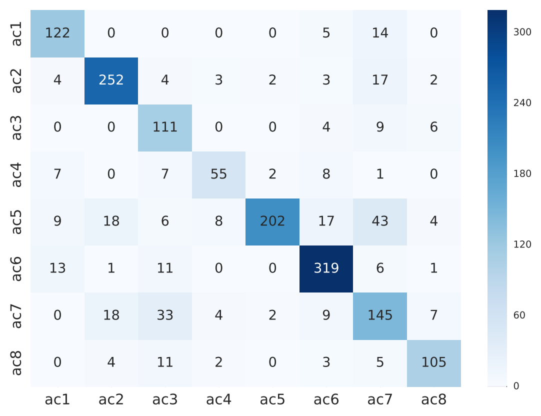
The results show that the application of our collaborative active learning method increases recognition performance of about 5%. In order to compare NECTAR with state-of-the-art techniques, we also implemented the supervised method proposed in [33], which relies on machine learning and time-based feature extraction. As machine learning algorithm we used Random Forests, since it is commonly used in activity recognition systems. We executed the experiments using that method with the same dataset using leave-one-subject-out cross validation. Results show that NECTAR outperforms the supervised method in terms of average F_1, and achieves equal or better results in recognizing 6 activities out of 8. The supervised technique performs significantly better in recognizing *prepare birthday card* (ac_5). The main reason is that the classifier was trained on temporal-based features that represent relations between sensor events. Thus, the order of certain sensor events but also their temporal distance leads to a reliable pattern for ac_5 in this dataset. We also compared NECTAR with a recent unsupervised method proposed in [15], where correlations are extracted from the Web and used by a probabilistic reasoner. Results show that NECTAR outperforms that method in recognizing 7 activities out of 8 on the CASAS dataset.

Inspecting the results of NECTAR, we observe that with the introduction of active learning the recognition rate remains stable or increases. Investigating the results in detail, we notice that the recognition rate of *clean* has a strong increase (ac_7, $+17\%$), while *prepare soup* (ac_6) remains unchanged. A deeper investigation pointed out that activity ac_6 was almost never queried, since its initial semantic correlations derived from our ontology were already sufficient to accurately recognize it. Regarding the other activities, we report an improvement which varies from 1% to 11%.

Considering the individual activities, Figure 3 highlights that there are almost no conflicting activity classes and that in general each activity is well recognized. However, we observe that *clean* (ac_7) is often confused with the remaining activities. Indeed, this is due to the fact that *clean* is not clearly bound to a certain location or sensorized object; hence, during that activity the inhabitant triggers several sensor events that indicate the execution of other activities.

The above mentioned results were obtained setting the entropy threshold to 0.9. As this value directly influences the number of queries issued by the system, it is an important parameter to consider. Figure 2a clarifies that on average a user had to answer 6 questions to achieve the reported improvement of 5%. In the considered dataset, only one day of ADLs for each subject was available. We expect that the average number of queries in a day for a specific user will significantly decrease over time, thus converging to 0 queries after few days. It is important to note that lowering the entropy threshold would still improve our results (see Figure 2b) but would determine a significantly higher number of feedback queries (see Figure 2a). As expected, we observe a tradeoff between the overall improvement of the recognition rate and the user's effort spent to provide feedback.

Table II outlines the individual F_1 scores for each activity achieved using different values of the entropy threshold. The results confirm that the mentioned tradeoff holds for almost



	ac1	ac2	ac3	ac4	ac5	ac6	ac7	ac8
ac1	122	0	0	0	0	5	14	0
ac2	4	252	4	3	2	3	17	2
ac3	0	0	111	0	0	4	9	6
ac4	7	0	7	55	2	8	1	0
ac5	9	18	6	8	202	17	43	4
ac6	13	1	11	0	0	319	6	1
ac7	0	18	33	4	2	9	145	7
ac8	0	4	11	2	0	3	5	105

Fig. 3. Confusion matrix using NECTAR (with active learning). Entropy threshold $\lambda = 0.9$, feedback support threshold $\sigma = 7.5$

every activity. An exception is *answer the phone* (ac_4) as the recognition rate remains almost unchanged. This can be due to the fact that the entropy computed on the segments related to this activity is always very high; hence, increasing the entropy threshold does not reduce the number of queries.

Besides entropy, we also assessed the impact of the feedback support value σ, which ensures that a personalized feedback item is transmitted only if it was derived from a sufficient number of feedback items from similar homes. Figure 2c outlines that when σ drops under a certain value, the system uses unreliable feedback, obtaining a detriment of recognition rates. On the contrary, using an excessively large value of σ, the system filters out relevant feedback that could improve recognition rates.

In general, our results clearly show that collaborative active learning is a reliable tool to discover new semantic correlations and in turn to improve the recognition rate. This is especially the case for sensors that do not carry explicit semantic information with respect to activities. For instance, our ontology did not cover the events related to motion sensors. Our system was able to automatically learn the semantic correlation for those sensors' types improving the recognition rate. Moreover, our method required on average only 6 feedback queries for inhabitant, ranging from a minimum of 3 to a maximum of 10. We believe that this number of questions is acceptable in many application domains, especially if user-friendly and context-aware interfaces for feedback acquisition are used.

VI. Discussion

The preliminary results reported in Section V are promising, but several aspects still need to be deeply investigated.

A. Interaction with the inhabitant

In order to make our system practical in real scenarios, we aim to investigate important contextual aspects that should be considered when evaluating whether to ask a feedback or not. These aspects include the number of queries that have already been asked recently, the current mood of the subject and whether he/she can be currently interrupted.

Moreover, the interface used to query the user should be intuitive and user-friendly. We are developing a prototype of such interface, which also includes a speech recognition module in order to let the inhabitant answer queries in natural language. Voice interface is particularly suitable for elderly subjects, thus facilitating their interaction with our system. We will carry out extensive experiments to understand the impact of this interface in real scenarios.

B. Privacy aspects

For the sake of this work, we assumed that the CLOUD SERVICE is trusted, while in a real scenario it can be considered an untrusted *honest-but-curious* third party. Hence, there is the need of protecting the confidentiality and integrity of user and infrastructure profiles, as well as the information about events and activities provided by the feedback. We intend to investigate solutions based on homomorphic encryption [34] and secure multi-party computation [35] in order to let the CLOUD SERVICE run its algorithms on encrypted data.

C. Ontology engineering

Even if our system relies on a generic and possibly incomplete ontology which considers general relationships between activities and home infrastructure, the engineering effort is still noticeable. We believe that this effort could be reduced by reusing and extending existing ontologies. However, one could argue that it would be easier to manually estimate correlations among activities and sensor events based on common sense. However, manual modeling is unfeasible in realistic scenarios. For instance, the dataset we used in our experiments involves 70 sensors and 8 activities, resulting in 560 different values of semantic correlations. Other real-world deployments are much more complex.

VII. Conclusion and Future Work

In this paper we presented a novel framework which exploits collaborative active learning to improve a generic ontological model of activities manually crafted by knowledge engineers. Experimental results show that our framework significantly improves the overall recognition rate, while issuing a limited number of queries to the inhabitants. In future work, we intend to extend our framework to exploit feedback for learning correlations between activities and *temporal patterns* of events. Moreover, the current system re-evaluates correlations values from scratch every time an update is received. We plan to improve the system by devising an algorithm to continuously adjust those correlations as the updates are received.

VIII. Acknowledgement

The authors want to thank the anonymous reviewers' for their constructive feedback and comments to help improve the quality of this work. The work of D. Riboni was partially supported by the DomuSafe project, funded by Sardinia regional government (CRP 69, L.R. 7 agosto 2007, n.7).

References

[1] P. N. Dawadi, D. J. Cook, and M. Schmitter-Edgecombe, "Automated cognitive health assessment using smart home monitoring of complex tasks," *IEEE transactions on systems, man, and cybernetics: systems*, vol. 43, no. 6, pp. 1302–1313, 2013.

[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. Riboni, T. Sztyler, G. Civitarese, and H. Stuckenschmidt, "Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning," in *Proceedings of ACM UbiComp*. ACM, 2016, pp. 1–12.

[4] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, J. Doppler, C. Holzmann, M. Kurz, G. Holl, R. Chavarriaga, H. Sagha, H. Bayati, M. Creatura, and J. del R. Millán, "Collecting complex activity datasets in highly rich networked sensor environments," in *Proceedings of the Seventh International Conference on Networked Sensing Systems*. IEEE, 2010, pp. 233–240.

[5] S. W. Loke, "Representing and reasoning with situations for context-aware pervasive computing: A logic programming perspective," *The Knowledge Engineering Review*, vol. 19, no. 3, pp. 213–233, 2004.

[6] X. H. Wang, T. Gu, D. Q. Zhang, and H. K. Pung, "Ontology Based Context Modeling and Reasoning using OWL," in *Proceedings of Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*. Washington, D.C.: IEEE Computer Society, 2004, pp. 18–22.

[7] L. Chen and C. D. Nugent, "Ontology-based activity recognition in intelligent pervasive environments," *International Journal of Web Information Systems*, vol. 5, no. 4, pp. 410–430, 2009.

[8] D. H. Hu and Q. Yang, "Transfer learning for activity recognition via sensor mapping," in *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*. IJCAI/AAAI, 2011, pp. 1962–1967.

[9] D. J. Cook, K. D. Feuz, and N. C. Krishnan, "Transfer learning for activity recognition: A survey," *Knowledge and Information Systems*, vol. 36, no. 3, pp. 537–556, 2013.

[10] D. Riboni and C. Bettini, "OWL 2 modeling and reasoning with complex human activities," *Pervasive and Mobile Computing*, vol. 7, no. 3, pp. 379–395, 2011.

[11] R. Helaoui, D. Riboni, and H. Stuckenschmidt, "A probabilistic ontological framework for the recognition of multilevel human activities," in *Proceedings of ACM UbiComp*. ACM, 2013, pp. 345–354.

[12] M. Perkowitz, M. Philipose, K. P. Fishkin, and D. J. Patterson, "Mining models of human activities from the web," in *Proceedings of WWW Conference*. ACM, 2004, pp. 573–582.

[13] D. Wyatt, M. Philipose, and T. Choudhury, "Unsupervised activity recognition using automatically mined common sense," in *Proceedings AAAI*. AAAI Press / The MIT Press, 2005, pp. 21–27.

[14] E. M. Tapia, T. Choudhury, and M. Philipose, "Building reliable activity models using hierarchical shrinkage and mined ontology," in *Proceedings of Pervasive*, ser. LNCS, vol. 3968. Springer, 2006, pp. 17–32.

[15] D. Riboni and M. Murtas, "Web mining & computer vision: New partners for object-based activity recognition," in *Proceedings of the 26th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*. IEEE Computer Society, 2017, pp. 158–163.

[16] A. Vahdatpour, N. Amini, and M. Sarrafzadeh, "Toward unsupervised activity discovery using multi-dimensional motif detection in time series," in *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, C. Boutilier, Ed., 2009, pp. 1261–1266.

[17] P. Rashidi, D. J. Cook, L. B. Holder, and M. Schmitter-Edgecombe, "Discovering activities to recognize and track in a smart environment," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 4, pp. 527–539, 2011. [Online]. Available: https://doi.org/10.1109/TKDE.2010.148

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

[19] J. Ye, G. Stevenson, and S. Dobson, "Usmart: An unsupervised semantic mining activity recognition technique," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 4, no. 4, pp. 16:1–16:27, 2014.

[20] R. Liu, T. Chen, and L. Huang, "Research on human activity recognition based on active learning," in *Machine Learning and Cybernetics (ICMLC), 2010 International Conference on*, vol. 1. IEEE, 2010, pp. 285–290.

[21] L. Yao, F. Nie, Q. Z. Sheng, T. Gu, X. Li, and S. Wang, "Learning from less for better: semi-supervised activity recognition via shared structure discovery," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*. ACM, 2016, pp. 13–24.

[22] T. Sztyler and H. Stuckenschmidt, "Online personalization of cross-subjects based activity recognition models on wearable devices," in *Proceedings of IEEE PerCom*. IEEE, 2017, pp. 180–189.

[23] M. Stikic, K. V. Laerhoven, and B. Schiele, "Exploring semi-supervised and active learning for activity recognition," in *Proceedings of the 12th IEEE International Symposium on Wearable Computers (ISWC)*. IEEE Computer Society, 2008, pp. 81–88.

[24] Y. Ho, C. Lu, I. Chen, S. Huang, C. Wang, L. Fu *et al.*, "Active-learning assisted self-reconfigurable activity recognition in a dynamic environment," in *Proceedings of the 2009 IEEE international conference on Robotics and Automation*. IEEE Press, 2009, pp. 1567–1572.

[25] L. Zhao, G. Sukthankar, and R. Sukthankar, "Robust active learning using crowdsourced annotations for activity recognition," in *Human Computation, Papers from the 2011 AAAI Workshop, San Francisco, California, USA, August 8, 2011*. AAAI, 2011.

[26] W. S. Lasecki, Y. C. Song, H. A. Kautz, and J. P. Bigham, "Real-time crowd labeling for deployable activity recognition," in *Proceedings of Computer Supported Cooperative Work (CSCW)*. ACM, 2013, pp. 1203–1212.

[27] H. M. 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.

[28] J. Wen, J. Indulska, and M. Zhong, "Adaptive activity learning with dynamically available context," in *Proceedings of IEEE PerCom*. IEEE Computer Society, 2016, pp. 1–11.

[29] M. Chekol, J. Huber, C. Meilicke, and H. Stuckenschmidt, "Markov logic networks with numerical constraints," in *22st European Conference on Artificial Intelligence (ECAI2016)*. Amsterdam, The Netherlands: IOS Press, 2016, pp. 1–9.

[30] M. Richardson and P. Domingos, "Markov logic networks," *Mach Learn*, vol. 62, pp. 107–136, 2006.

[31] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, "Tracking activities in complex settings using smart environment technologies," *Int. J. Biosci. Psychiatr. Technol.*, vol. 1, no. 1, pp. 25–35, 2009.

[32] 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, 2013.

[33] N. C. Krishnan and D. J. Cook, "Activity recognition on streaming sensor data," *Pervasive Mob Comput*, vol. 10, pp. 138–154, 2014.

[34] A. A. Atayero and O. Feyisetan, "Security issues in cloud computing: The potentials of homomorphic encryption," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 2, no. 10, pp. 546–552, 2011.

[35] Y. Lindell and B. Pinkas, "Secure multiparty computation for privacy-preserving data mining," *Journal of Privacy and Confidentiality*, vol. 1, no. 1, p. 5, 2009.