Using Ontologies in Case-Based Activity Recognition

Stephen Knox
Systems Research Group
School of Computer Science and Informatics
UCD Dublin, Ireland
stephen.knox@ucd.ie

Lorcan Coyle
Lero—the Irish Software
Engineering Research Centre
University of Limerick
lorcan.coyle@lero.ie

Simon Dobson
School of Computer Science
Universty of St Andrews, UK
sd@cs.st-andrews.ac.uk

Abstract

Pervasive computing requires the ability to detect user activity in order to provide situation-specific services. Case-based reasoning can be used for activity recognition by using sensor data obtained from the environment. Pervasive computing systems can grow to be very large, containing many users, sensors, objects and situations, thus raising the issue of scalability. This paper presents a case-based reasoning approach to activity recognition in a smart home setting. An analysis is performed on scalability with respect to case storage, and an ontology-based approach is proposed for case base maintenance. We succeeded in reducing the casebase size by a factor of one thousand, while increasing the accuracy in recognising some activities.

Introduction

Pervasive computing is a branch of computing which deals with the integration of computing devices and services into our everyday life. The pervasive, or ubiquitous, system was first envisioned by Mark Weiser (1999), who said computers would "recede into the background of our lives until they become indistinguishable from it".

Pervasive computing has numerous aspects which make Weiser's vision difficult to achieve. Not least of these is the requirement of the environment to decide upon a user's current situation or activity and to decide which course of action, if any, it should take. To detect the activity of a user, sensors are needed. These sensors can detect many types of information about people: their heart rate, location or upcoming appointments. Sensors also detect environmental aspects such as the temperature of a room, water flow to a faucet or amount of light in a room. By analysing current and past sensor readings, it is possible to determine user activities.

A number of techniques have been presented for determining user activity (Ye et al. 2009; Logan et al. 2007; van Kasteren et al. 2008), but these methods have demonstrated a number of drawbacks. Situation lattices (Ye et al. 2009) show promising results but have scalability issues. Naïve Bayes and Decision trees, like lattices, require significant training data for good accuracy. As a static training set is used by these techniques, they need to be re-trained whenever a new activity occurs. This is a serious limitation in pervasive computing as new activities can occur very frequently. We believe that case-based reasoning provides good alternative to these techniques, as it can be run with no training data, and can learn new situations and activities incrementally as they occur. We propose SituRes, a case-based approach to recognising user activity in a smart-home[1] environment. We use pre-recorded datasets which are replayed in real-time to emulate a live system. Using this simulated environment, we demonstrate how CBR performs by examining the cases which are learned and the accuracy of recognising activities.

Case-based reasoning fits very well into the dynamic and unpredictable nature of pervasive computing as it can learn new activities, thus eliminating the requirement for a training phase. However, as new users, devices and locations are introduced, a greater number of complex activities can occur. If the casebase is not maintained, the casebase can increase to an unmanageable size. If CBR is to be a viable technique for activity recognition in a large pervasive system, the issue of scalability must be addressed. To this end, we present and evaluate a technique for reducing the size of a casebase while maintaining accuracy. We show the limitations of this approach and introduce a technique, based on a semantic description of the domain to further reduce the casebase size, while improving accuracy for some activities.

Weiser's vision is also restricted by the sheer complexity and dynamism of the "real world". There are so many variables, be they users, locations or sensors, each with different characteristics, it is impossible to hard code the possible activities which can occur. There is therefore a need for a model of the environment which can be altered and updated as required. Using this model, relationships can be defined between entities, locations, objects and activities. These relationships can then be exploited in applications to provide the correct services to the correct users.

Ontologies provide the most complete way of defining the contextually relevant aspects of an environment (Strang and Linnhoff-Popien 2004). An ontology is an explicit modelling of the fundamental concepts of a domain, which can be shared and reused. We use an ontology, written in the Web Ontology Language (OWL) to define our pervasive en-

[1] A smart home is effectively a small-scale pervasive system

vironment. This ontology is then used to represent the relationship between activities and the sensors which are used for their identification. Using these relationships, we can infer how a situation relates to the environment. For example, we can automatically identify that the situation "meal preparation" tends to be solely associated with the kitchen.

By exploiting the relationships between activity and location, we define a new casebase, where the cases contain only features semantically linked to the solution. In doing this we succeed in reducing the casebase further, while demonstrating that accuracy in recognising activities can be improved by using formally defined domain knowledge.

Related Work

Leake et al (Leake, Maguitman, and Reichherzer 2006) present a set of practical motivations for the use of CBR in context-aware systems. These include the *personalisation* of the dataset through learning, *explanation* of decisions made by the reasoner, and enabling user interaction. They also note that there should be a means of addressing the volume of cases added while learning.

Casebase maintenance attempts to ensure that a casebase does not become overloaded with unnecessary information, while maintaining an acceptable level of accuracy (Smyth and Keane 1995). As case bases grow to contain thousands, if not millions of cases, massive overhead can be put on CBR systems, thus requiring that some deletion or summarisation strategy be employed (Sun et al. 2009). We apply two case base maintenance techniques to a CBR engine in a smart-home environment. The first is to summarise the casebase based on feature utility (page 3). The other uses ontologies to reason about the state of the environment, choosing the most appropriate features for each case (page 5).

An ontology is a formal specification of a something which is commonly understood. Ontologies can be used to describe system architectures or the situation of a person or thing. Some formal language must be used to define an ontology. These typically include programming languages or markup languages. The Resource Description Framework (RDF) is a commonly used language for describing ontologies. An extension of this, the Web Ontology Language (OWL) is used most often. Ontologies are powerful as they provide the ability to infer relationships between entities.

A number of CBR systems have been proposed to deal with activity recognition in pervasive systems. Kofod-Petersen used the CREEK (Aamodt 2004) system to determine activities in a hospital ward scenario (Kofod-Petersen and Aamodt 2009). While this work does demonstrate the applicability of CBR in determining situations, there are a number of limiting factors. The cases built were limited to nine features including *location, user, role* and *time*. These are high-level pieces of information, which must be inferred from lower granularity sensed data. This information was gathered in their user study by a participant who followed a doctor for a number of days, noting the criteria and the corresponding situation. Kofod-Petersen also noted, in (Kofod-Petersen 2006), that scalability is an issue for CBR in activity recognition due to the possible volume of cases being added to the casebase.

PlaceLab is a sensor-rich home environment, built by MIT as a shared research facility (Logan et al. 2007). This apartment is custom-built to contain many different sensors to help track participants as they perform tasks. The sensors used in PlaceLab are used to record many aspects of the environment including levels of electrical current, light and water flow, as well as tracking user movement and interactions using infra-red motion sensors and RFID tags.

A number of datasets were recorded using the PlaceLab and made publicly available. This work is based on one of these datasets[2], where a couple were recorded living in the apartment for a period of ten weeks. Using video recordings, the activities of the couple were annotated after the trial was completed. Due to the expense and time constraints of annotating video data by hand, only 104 hours of the 10 week period were annotated. Out of these 104 hours, 25 hours of data were chosen in this work, where the data was between 6 and 11 PM on five separate days. We used the same time period over five days as this would give a good chance of recurring events.

While numerous datasets are available for smart-homes, the PlaceLab dataset is the most comprehensive in terms of the number and variety of sensors. There are a number of issues with the PlaceLab dataset, however.

- Although there were two participants, only the male's actions were annotated, so while both participants were interacting with their environment, only one activity was annotated. The result of this is that the actions and resulting sensor readings of the female act as noise. This noise can make it very difficult to distinguish the actions of the male.
- The annotations provided, while comprehensive, are sometimes inaccurate. For example, the events "using a computer", "watching TV" and "reading paper" overlap at one point. This is inaccurate given that the TV is in a different room to the computer.

For these reasons, the accuracy for activity recognition in this dataset tends to be quite low. However some promising techniques have been proposed for increasing accuracy based on sparse annotation (Stikic and Schiele 2009).

SituRes

SituRes is situation reasoner based on the Fionn CBR toolkit(Doyle et al. 2004). Fionn uses the Case-Based Modelling Language (CBML) to represent cases, which allows their definition to be made independently of application code (Coyle, Doyle, and Cunningham 2004). Using CBML, the type, weight and similarity metric of each feature can be manipulated externally. Each feature defined in CBML is directly related to the status of a sensor in the environment.

The PlaceLab data set is used to evaluate our approach to activity recognition. By replaying the sensor data back as though it were occurring live, we also show how our approach performs in a real-time system. The simulator used in this work retrieves sensor data from a database, stores the

[2]We used the *PLCouple1* dataset

events in order of timestamp and replays them at the appropriate time intervals. As an event occurs, it is classified based on its sensor type and reading. For example, if the event is of type electrical current and the value is above 5, then that sensor is classified as "ON". To create these classifications, we use the ontological description of the sensors, which define the units of measurement they use, their frequency of output and expected levels of accuracy (how likely a reading is to be correct).

All classifications are stored for a period of time, typically ten seconds, until a new case is created. When this happens, each classification is added as a feature to a new test case.

Once the test case has been created, a K-NN retrieval algorithm is used to find the closest match for this case. This algorithm ranks cases based on a weighted average of the feature similarities. The K highest ranked cases are then returned in order of activation. By altering the importance of different case features the retrieval algorithm will return a different order to the retrieved cases. The ability to rank features is a very important aspect of this work, as all sensors are not equal. We might judge the values returned by the location sensors as being more important than the electrical current going to the TV for determining a person's activity.

When the nearest neighbours are returned, their activation is compared to that of the test case. If the activation is below a threshold value we view this test case as being novel and add it to the casebase automatically – we judge that the casebase does not have adequate coverage in this area. The threshold value is defined externally, and is chosen based on what degree of learning is required. A threshold of 0 encourages new additions by requiring an exact match of activations (an average of 488 cases were added per 5-hour period). A higher threshold value discourages the addition of new cases, but can have an impact on accuracy as the case base competence is less likely to increase. We found through experimentation that a threshold value of 2 provides a good trade-off between encouraging addition of cases, while allowing similar cases to be accepted (An average of 100 cases were added).

Before being added to the casebase, we select the solution by retrieving the corresponding activity from the diary annotations contained in the dataset. The dataset contains a set of annotations, each with two timestamps – start-time and end-time. Should a case fall between the start time of one annotation and the end time of another, the situations are seen to be co-occurring, thus the solution contains two activities with an "AND" between them. For example, if "Using a Computer" is annotated between time T1 to T10 and "Eating" between time T5 and T15, then should a case occur between T5 and T10, the case solution is "Using a Computer AND Eating".

The co-occurrance of activities causes difficulties when retrieving cases from the casebase. The K-NN algorithm retrieves the K best solutions, but these solutions may contain multiple activities which overlap and contradict one another. It is very difficult to merge multi-activity solutions in a satisfactory manner. This problem restricts the number of cases we can retrieve from the casebase – K – to 1.

The Placelab Ontology

The Ontonym ontologies (Stevenson et al. 2009) allow the specification of pervasive computing environments and applications. Using Ontonym as a template, we created an ontology which represents the objects, sensors, and locations within the PlaceLab apartment. Our ontology provides a very flexible way of defining relationships between these entities. For example, we define a sensor based on its output measurements, frequency of readings and location. We then associate this sensor with an object by asserting the *attachedTo* property. This simple relationship is very powerful, as it allows an object to be located based on the location of its associated sensor. In other forms of data representation, this link would have to be explicitly stated, but by availing of *property chaining*, this relationship is inferred automatically. The PlaceLab ontology is available at the Ontonym website[3].

Reducing Casebase Size

To demonstrate how the the casebase size may be reduced, we created a training set by recording the current case every ten seconds for each of the five days in the data set. Consecutive identical cases were not recorded. 6003 cases were recorded using this learning-only process. A casebase of 6000 cases is comparatively modest, but this represents the cases for a single user in a single location, engaging in a small number of situations. Should a system contain several users, the casebase would conceivably see 1000 additions per day.

To create a cut-down case base, the unique solutions are identified. For each of these solutions, a feature selection process is employed, whereby the features of each corresponding case are ordered based on how predictive they are of that case. This ordering is performed using the *information gain* algorithm. We used Ontonym to link each case feature with its related sensor and from this we could determine the location and type of this sensor.

A single case is then created based on these features, and added to the casebase. This process is repeated for each solution, leaving a casebase which contains as many cases as there are solutions; in this example the casebase is reduced from 6003 to 36 cases.

This approach has a number of drawbacks, however. Although the casebase size is reduced dramatically, incorrect or inappropriate features might have far more influence should they be ranked highly. Given the limited size of the data set we used, noisy data has a high impact on accuracy, and this is magnified by the pruning process. For example, in figure 1, we can see that there are a large number of current sensors active during the "watching TV" activity. This can be explained by user habits in the PlaceLab environment – the participants often left appliances and lights turned on regardless of their situation. Therefore, current sensors cannot be fully trusted for accuracy. This leaves us with RFID, WaterFlow, Object Motion and Person Motion sensors. From this analysis, we reduce the weights of the current and light sensors to 0.5, and increase the weights of the other sensors.

[3] `http://ontonym.org/PlaceLab`

Object	Loc	Type	Count	I-Gain
light	office	CURRENT	1.0	0.48
light	kitchen	CURRENT	2.0	0.45
room	office	MOTION	87.0	0.25
room	kitchen	MOTION	93.0	0.21
television	livingroom	OM	248.0	0.19
light	livingroom	CURRENT	416.0	0.19
room	hallway	MOTION	98.0	0.07
light	livingroom	CURRENT	30.0	0.05
room	officehallway	MOTION	75.0	0.04
room	dinningroom	MOTION	116.0	0.04
light	diningroom	CURRENT	2.0	0.03
TVremote	livingroom	OM	152.0	0.03
circuit	yard	CURRENT	38.0	0.03
toilet	powderroom	FLOW	8.0	0.02
couch	livingroom	OM	17.0	0.02
cabinet	kitchen	OM	2.0	0.01
chair	diningroom	OM	2.0	0.01
closet	hallway	OM	1.0	0.009
...	...	...	...	...

Table 1: The information gain of each feature with respect to "watching TV" on Day 2 of the PlaceLab dataset. The count column refers to the number of times a feature occurred with respect to the situation.

For example, the weight of all RFID sensors is updated to 2.5, as this is seen as highly representative of activity. Finally, no processing is done on the solutions themselves, so each solution may contain multiple activities deliminated by an "AND". As discussed, this restricts the testing on pruned casebases to a K of 1. We deal with this problem by pruning the casebase even further.

Rule-Based Case Processing

Having determined the "weaker" sensor types, a rule-based reasoner is applied to the PlaceLab ontology to further reduce the casebase size. This additional step has two advantages:

- The complexity of the case base is reduced, as each case contains only features which are correctly representative of the solution.
- Case solutions are analysed and their complexity reduced. All cases where the solution has more than one situation are removed, leaving only single-activity cases. We cater for the occurrence of multiple activities by retrieving multiple cases from the casebase.

Two rules, defined using the Jena rule engine[4], are used to process the casebase. The *primary location* of each activity is found first. The primary location is the location in the apartment in which a situation is most likely to take place. The primary location is found by examining each feature of a case and determining the location of the sensor which created its associated reading. We use object motion, water flow, person motion and RFID as indicators of location. If a common location is determined, a property is asserted in the PlaceLab ontology which indicates the primary location of a situation.

Once the primary location of a feature has been determined, the ontology is queried for each sensor located in that place. These sensors are then used to create a case which represents that situation. For example, if the office is found to be the primary location of "using a computer", then all sensors based in the office are turned into features for that case.

Restricting activities to a single location is a broad criteria for creating cases which is prone to overfitting, but we use this approach to show how a model of the environment can help more accurately define cases.

Evaluation

As five days of training data was used, the stratified leave-one-out cross validation method was adopted, using four days of training data for one day of testing. Each day is tested, and the average of the results is found at the end. This approach was also adopted in (Logan et al. 2007; Ye et al. 2009). Accuracy is evaluated by recording the *precision*, *recall*, and calculating the *f-measure*.

Definition 1 Precision *refers to the number of times an activity was correctly identified versus the number of times it was chosen by the reasoner.*

Definition 2 Recall *refers to the number of activities which were classified versus the number of times they actually occurred.*

Definition 3 *The f-measure is used to combine the two aspects of a reasoner, defined by*

$$F = \frac{2 * Precision * Recall}{Precision + Recall}$$

Incremental learning

As this CBR engine learns cases incrementally, an evaluation was first performed with no initial training set. This examines the learning capabilities of the system. For the reasons discussed in Section 3, a threshold of 2 and a K value of 1 were used. Each day is run separately with no training data and average f-measure for a number of common activities is calculated.

An average of 100 cases were added each day. While 100 cases might seem small, this represents just one user over a five hour period. As the number of users and possible situations increases, this number will likely increase dramatically. In order to see how well these cases represented their solutions, they were used as training for the other days. Learning was disabled to ensure that no cases specific to that day were used.

Figure 1 shows the f–measure of incremental versus trained data. For the incremental approach, the casebase was initially empty, and cases were added as necessary. In almost every situation, the incremental technique performs better, showing that improved accuracy may be gleaned from the use of incremental learning on a daily basis. However the cases learned tend to be highly sensitive to the noise present when they were recorded.

[4]http://jena.sourceforge.net/inference/

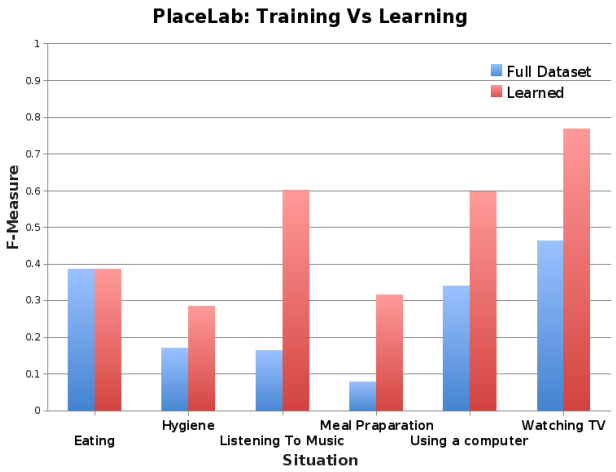

Figure 1: F-measure of Incremental CBR versus statically trained CBR

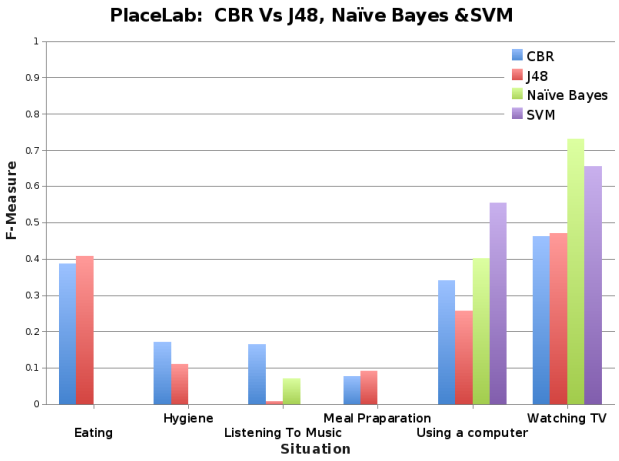

Figure 2: F-measure of CBR, Naïve Bayes and J48 for a number of activities

It is also clear that general accuracy tends to be quite poor, with only three activities having an f-measure of greater than 0.5. The events with the highest accuracy occur most often (over 10 hours are spent "using a computer" in the 25 hours we used), thus have far more reference cases. The "hygiene" activity, for example, happens in short intervals (just under 8 minutes are spent in this activity over the testing period), meaning significantly less learning is possible.

While using incremental learning yields encouraging results, the cases added to the casebase tend to be quite specific to the global circumstances in which they were recorded. It is therefore likely without some degree of casebases maintenance, the casebase will fill with highly specific and contextualised cases.

Full training set

Using a "full" training set allows for a more meaningful comparison between CBR and other techniques. As CBR learns incrementally, it is difficult to compare to other activity recognition techniques, which require training data. To perform this comparison appropriately, the same training set was used for CBR, J48 Decision Trees (J48) and Naïve bayes (NB) and support vector machines (SVM). This approach allows a direct comparison of the reasoners, regardless of learning capability. Figure 2 shows the f-measure comparison of CBR with NB, J48 and SVMs. From these results we can see that while good results are achieved by SVMs and NB for "using a computer" and "watching TV", they achieve poor accuracy or fail to classify the other activities. J48 and CBR are more consistent, providing a classification for each activity, and do not perform significantly worse than SVM or NB for any activity.

Overall we can say that CBR is comparable with the other techniques. This is an encouraging result, as this experiment was performed with no altered weights (all feature weights were set equally) and with learning disabled. With learning enabled, we see accuracy to increase, to similar levels as figure 1.

Pruning the Dataset

We pruned the full training set generated for the evaluation to contain just one case per solution. This pruned dataset was then used, as before, in a stratified cross validation (K=1, Threshold=2).

Finally, the full dataset was processed using both rule-based and ontological reasoning. Using the rules described in Section 4, the primary location of each activity was identified, e.g., kitchen was found to be the main location for the "meal preparation" activity. If no one location could be found, for example in the case of "listening to music in the background", the case was summarised without processing. For each primary location, all sensors were found, and all possible classifications from those sensors. Each of these classifications were used as features in the case corresponding to that location. This process left us with 6 cases, one for each activity. In simulation, K was set to 5. A K of 5 was chosen as this was in line with the largest number of concurrent cases which occurred in the PlaceLab dataset. Of the 5 returned cases, only cases with activations equal to the top neighbour were chosen as solutions. As there are no cases containing an "AND" in the casebase, the nearest neighbours are merged with an "AND" to cater for co-occurring situations.

Figure 3 shows the comparison of using a full, pruned and semantically processed training set. Using a pruned dataset showed improvements in "watching TV" and "hygiene" while only performed worse in "meal preparation" and "eating". This shows that by simply summarising the cases in a casebase, gains can be made in accuracy. Having processed the training set using the PlaceLab ontology, we see that the accuracy for "using a computer" rose by nearly 17% over a full dataset. This increase can be attributed to the high correlation between a specific location and the activity. Predictably, "meal preparation" rose also, but not significantly. Interestingly, "watching TV" and "hygiene", which one would expect to be highly location-specific perform poorly. For "watching TV" this can be attributed to

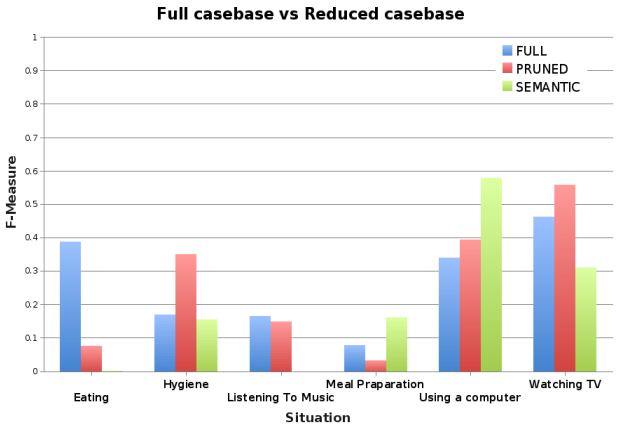

Figure 3: Accuracy using a full, pruned and semantically processed dataset

the fact that the PlaceLab apartment is open-plan, meaning it is difficult to pinpoint a user's location, but also watching TV tends to occur in conjunction with either eating or meal preparation. For "hygiene", the poor accuracy may be attributed to the fact that a small number of sensors are available in the powder room, thus making it slightly more difficult to distinguish.

Conclusion

This paper presented a CBR approach to activity recognition in a smart-home environment. We have shown that CBR is comparable to other techniques when using a static training set, and can perform better through incremental learning. As CBR engine learns, however, the casebase can become very large, causing scalability issues.

Using an ontology, which describes the application area, it is possible to perform effective case base maintenance while maintaining accuracy. We performed case base reduction first using a simple statistical technique, and then by semantically linking the case solutions with corresponding case features. These features were chosen based on the location of the sensor which created them.

By analysing each of these reduction techniques, we showed how the case base may be significantly reduced in size. While this reduction in training data caused a decrease in accuracy for some specific solutions, an increase was seen in others. This suggests that a more careful approach to feature selection can yield higher accuracy.

Creating a more efficient approach to feature selection will be the focus of future work. By creating a more comprehensive rule base and picking features more carefully, we hope to increase accuracy in recognising user activities.

We will also investigate how two solutions containing multiple situations may be merged. In this work, we were restricted to using a K value of 1, which limited the possible results. Increasing the value of K could result in a smoothing effect, whereby noisy data would become less influential, increasing accuracy.

Acknowledgment

This work was supported, in part, by Science Foundation Ireland grants 03/CE2/I303_1, 07/CE/I1147, and 05/RFP/CMS0062.

References

Aamodt, A. 2004. Knowledge-intensive case-based reasoning in CREEK. In *ECCBR*, 1–15.

Coyle, L.; Doyle, D.; and Cunningham, P. 2004. Representing similarity for CBR in XML. In Calero, P. A. G., and Funk, P., eds., *Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004 Madrid, Spain*, 119–127. Springer.

Doyle, D.; Loughrey, J.; Nugent, C.; Coyle, L.; and Cunningham, P. 2004. Fionn: A framework for developing CBR systems. *Expert Update* 8(1):11–14.

Kofod-Petersen, A., and Aamodt, A. 2009. Case-based reasoning for situation-aware ambient intelligence: A hospital ward evaluation study. In *ICCBR*, 450–464.

Kofod-Petersen, A. 2006. Challenges in case-based reasoning for context awareness in ambient intelligent systems. In *ECCBR Workshop Proceedings*, 287–299.

Leake, D. B.; Maguitman, A. G.; and Reichherzer, T. 2006. Cases, context, and comfort: Opportunities for case-based reasoning in smart homes. In *Designing Smart Homes*, 109–131.

Logan, B.; Healey, J.; Philipose, M.; Tapia, E. M.; and Intille, S. S. 2007. A long-term evaluation of sensing modalities for activity recognition. In *Ubicomp*, 483–500.

Smyth, B., and Keane, M. T. 1995. Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. In *IJCAI*, 377–383.

Stevenson, G.; Knox, S.; Dobson, S.; and Nixon, P. 2009. Ontonym: An ontology for pervasive application development. In *Workshop on Context, Information And Ontologies*. To Appear.

Stikic, M., and Schiele, B. 2009. Activity recognition from sparsely labeled data using multi-instance learning. In *LoCA*, 156–173.

Strang, T., and Linnhoff-Popien, C. 2004. A context modeling survey. In *Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp 2004*.

Sun, J.; Yu, X.; Wang, R.; and Zhong, N. 2009. A model for personalized web-scale case base maintenance. In *AMT*, 442–453.

van Kasteren, T.; Noulas, A. K.; Englebienne, G.; and Kröse, B. J. A. 2008. Accurate activity recognition in a home setting. In *UbiComp*, 1–9.

Weiser, M. 1999. The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.* 3(3):3–11.

Ye, J.; Coyle, L.; Dobson, S.; and Nixon, P. 2009. Using situation lattices in sensor analysis. In *IEEE International Conference on Pervasive Computing and Communications (PerCom 2009)*, 1–11.