Context Modulation of Sensor Data Applied to Activity Recognition in Smart Homes

Niall Twomey, Peter Flach

Department of Computer Science, University of Bristol.

Abstract. In this paper we present a method of modulating the context of data captured in smart homes. We show that we can dramatically adapt their sensor network topology and that this approach can be used to help understand various aspects of such sensor environments. We demonstrate how, with our software, we can discover the importance of individual sensors, clusters of sensors and sensor categories for resident identification and activity recognition. Finally, we validate the utility of context modulation in a number of experimental scenarios that show how the activity recognition is affected by each sensor topology elicited by these scenarios.

1 Introduction

Activity recognition is an important area of research and is a principal goal of many research groups [3,7,12,1,16]. Homes equipped with sensing technology that are capable of perceiving the activities being performed are called smart homes. In general these homes are equipped with passive sensors that do not require direct interaction of residents. Examples of these sensors are Passive Infra-Red (PIR) which detect movement, door sensors which identify whether a door is open or closed, light switch sensors which report whether a particular light is on or off, etc.

A critical requirement of smart homes is in the recognition of Activities of Daily Living (ADL) [4,7]. These are of interest because certain types of smart home may be able to interact and assist with ADLs. Assessment of ADLs can also shed light on the well-being, independence and cognitive abilities of the residents [4,1]. With ADL recognition tasks, it is important to find a balance between the number of sensors (people may not accept high numbers of these in their homes), robust classification capabilities and tolerance to faults that might occur in deployment. However, few research groups consider all of these points because investigating these may require that the context of already-existing data sets are significantly modified and such contextual modifications might be difficult to generate.

In this paper we describe a solution to the above problem and describe a technique that can easily manipulate already-existing smart home datasets into representing new user-defined contexts. Such new representations can be useful for understanding the sensor dynamics of smart homes, for building robust classification models and for assisting with transfer, multi-task and meta learning.

In our results section we demonstrate the utility of this pipeline with a number of different experiments.

This paper presents work in progress, yet we contribute to the literature in the following manners: 1) a powerful interface is presented which can dramatically modify the context of smart homes and allows a) investigation of the effect of different sensor layouts or failures; b) the change the form of the classification task; and c) feature engineering studies to be performed easily; 2) we demonstrate how modulating data can help us understand the importance and effect of sensor densities and topologies (our system can streamline this process); 3) we state that this work could be used in conjunction with learning algorithms so that robust, fault-tolerant classification models may be learnt from the data; and 4) our interface can simulate and quantify the effect of a number of scenarios that residents may insist on in deployment studies.

2 Context Modulation

We define context modulation as the act of modifying and adapting data in such a way that new data representations are obtained. These new representations are illustrative of original (user-defined) contexts. In this way, context modulation acts on the data itself and can generate new versions of existing datasets which depict new properties according to the parameters specified by an operator.

Employing multiple independent datasets is the traditional way to learn over multiple contexts. Our context modulation technique, however, generates new representations of the same datasets and these permit us to learn new lessons from the set of contexts individually and collectively. These lessons are interesting as they illustrate the effect of context change on a dataset with the same application domain target.

Throughout this paper we discuss context modulation with reference to activity recognition in a smart home. However, even under this application, context modulation can be applied to a number of different domains, and we will show later that it is useful for the software-defined assessment of sensor topologies, assessing importance of sensors for activity recognition, and it may also be used for learning robust classification models.

2.1 Motivation

We first motivate the idea of context modulation as an approach for simulating fault-like sequences that might occur in a smart home. In general, smart home datasets are "clean" (i.e. all sensors are always operational). However, this is an idealised view and may not always occur in a realistic deployment. Context modulation can provide protection against such scenarios by modulating training data in such a way that fault-like scenarios have already been contemplated by a learning algorithm. Our software can also expose the capability of modifying sensor and label topologies in ways that can translate the problem to a more appropriate form. For example, some smart home datasets are presented

as single-task classification problems to learning algorithms, but they may be more appropriately viewed in a multi-task setting.

Sensor Failure For any given activity, when a sensor has failed the sequence of events that would normally be perceived will likewise change. This poses a problem for activity classification models as they generalise by learning correlations between sensor activations and labelled activities.

Within the remit of a smart home, there exist a number of ways in which a sensor may fail. The sensor hardware may simply break, which might occur given the humid environments in a kitchen or shower in the bathroom. Alternatively, the battery of the sensor may fail. Sensors are not generally powered by mains electricity for reasons of expense, availability of power points and user-acceptance. In order to communicate to a database, the sensor must broadcast data wirelessly, and this is quoted as being the single most significant burden on the power budget in many applications [14].

An interesting case arises with PIR sensors. After replacement of batteries, for example, or perhaps if one has been knocked off alignment by a resident, the orientation of the sensor may be different and its field of view may now be focused over a new area. If the positioning of these sensors is critical for activity recognition, it is possible that classification accuracies may be adversely affected by this change of orientation. An example of where the PIR positioning may be important is within a kitchen where a number of activities (such as washing up, cleaning and cooking) take place in a confined area.

Sensor Density There is an intricate trade-off between the number of sensors that are found within a smart home and the user-acceptance of the presence of these. This is most important for projects whose eventual goal involves wide-scale deployment. In general, engineers will prefer to populate smart homes with as many sensors as possible in order that the most expressive description of activities is obtained. This high sensor density introduces redundancy to the testbed[1], but few studies investigate whether such high sensor densities would be accepted by users, and so it is possible that deployment-centred projects may limit recruitment populations with increasing numbers of sensors.

Further to user-acceptance considerations, few studies have investigated whether high sensor densities are required for accurate activity recognition. One could investigate this question with feature-selection methods to determine the sets which are most and least informative. However, this approach does not permit the researcher to investigate the effect of introducing new kinds of sensor that could be chosen to take the place of, perhaps, five of the current sensors, for example, and we will see later in this paper how this might be achieved with our software.

[1] Testbed are smart homes which are specifically designed for data capture. Residents may live in these locations for a period of time, but in general this will not be their permanent place of residence.

Table 1: Example of a (modified) sensor event file. The activities are identified by the final two columns.

Date/Time	Sensor ID	Sensor State	Label	Begin/End
2009-02-02 07:15:16	m_16	on	R1_Work	begin
2009-02-02 07:15:22	m_34	off	·	·
2009-02-02 07:15:23	m_22	on	R2_Sleep	begin
2009-02-02 07:15:28	m_26	on	·	·
... etc ...				
2009-02-02 07:20:55	m_22	on	·	·
2009-02-02 07:20:56	m_43	on	R2_Sleep	end
2009-02-02 07:21:00	m_13	on	·	·
2009-02-02 07:21:03	m_19	off	R1_Work	end

Transfer and Multi-task Learning When learning activity models from a set of testbeds the models must be able to cope with different floor plans of homes. Researchers in multi-task and transfer learning have stated (e.g. in [2]) that the best results are often obtained by sharing parameters between the different testbeds, tasks and resources. In order to do this, each testbed must translate their events to a common language that has been agreed upon. As an example of such a common language, a motion sensor with the identifier `m_17` which is found in a kitchen may be replaced with `m_kitchen`. While this is a trivial example, we show that this idea can be used to advantage in later sections.

2.2 Implementation and Utility

In this section we discuss a number of details about the implementation of our context modulation software and how all of the considerations in Section 2.1 have been addressed in our implementation. In brief, this section describes four ways by which we can alter the context of datasets under the guidance of a user. Each method can be used in isolation or in combination with others.

Input Sensor Event File Our sensor modulation technique is capable of generating data which can address the points raised in Section 2.1. In order to do so, our data generation algorithm reads from sensor event files. An extract of one such file is shown in Table 1. The first three columns of the table give the date/time, sensor identifier and the state of an event. The set of possible sensor states of doors are selected from the set $\{\text{open}, \text{closed}\}$ while motion sensors will select from $\{\text{on}, \text{off}\}$. The *Label* and *Begin/End* columns of the table identify the current activity being performed and the start and end points of this activity. The naming convention of activities, e.g. `R1_Work`, states that the resident, identified as `R1`, is working.

Multiple residents can perform activities in parallel in a smart home (e.g. one resident may be working while another sleeps). When this occurs the labelling

associated with new events is ambiguously reported, as these files effectively state that "when this event was recorded, these 2 activities were co-occurring" so no one activity is specified as being responsible for causing new events. As the class membership is not explicitly specified the annotations are somewhat ambiguous and weakly labelled (i.e. this event belongs to one element of this set of activities, and precisely which element is not known; see Table 1). The activity recognition research community can classify these problems in a multi-class manner [2] because even though multiple activities may occur in parallel, each event can only be generated by one activity from the set of the possible candidates.

Sensor Deletion and Merging Two main types of feature modulation can be performed on the sensor events. These modifications are defined by regular expressions within appropriate Extensible Markup Language (XML) files.

If a user wished to remove sensor `m_35`, for example, either to simulate a failure of the sensor or to introduce a persistent "blind spot" in the sensor topology, this may be done by specifying `^m_35$` within the sensor removal XML file. The software will then remove each event that matches this expression. Looking at Table 1 it can be seen that if the software simply ignored rows that satisfy this expressions, the beginning of the `R1_Work` activity would likewise be removed. Therefore, when this situation occurs the software will first "remember" the opened activity before shifting on to the next row of the data file.

Likewise, if one wished to merge sensors `m_35` and `m_34` and label these with a new "super" sensor identifier, `m_bedroom_1`, one may do so by specifying the following regular expression: `/^m_(34|35)$/m_bedroom_1/` and if no match is found the original string is returned. This procedure simulates the act of adding "new" sensors to the testbed which exhibit different properties from the original sensors, e.g. it may simulate PIR sensors with wider fields of view. This approach can also be used to assess the effect of investigating lower sensor densities for activity recognition. It is possible to use both deletion and merging expressions together and Algorithm 1 shows how this is achieved.

Feature Engineering Algorithm 1 provides an interface to modify the sensor activations of a dataset which can yield different user-defined sensor contexts. However, it also facilitates feature engineering on the full set of features that are extracted from the sensor event files. For example, it is possible to extract the day of the week and hour of the day from the date and time column of Table 1. Our software permits users to easily modify these other features (as opposed to the sensors that were affected previously) by specifying regular expressions. For example one may define breakfast time as occurring between 7 and 10 AM. This can be extracted with `/^[7-10]am$/breakfast_time/`. Similarly, identification of weekends can be specified by `/^(saturday|sunday)$/weekend/`. It is also possible to generate extra features with a regular expression such as `/^(saturday|sunday)$/weekend,\1/` which presents both the original day and the weekend as features.

Algorithm 1: Algorithm for processing sensor event files.

Input: A sensor event file, $\mathcal{D}$, a set of deletion regular expressions, d and a set of merging regular expressions m.

Output: The processed dataset $\mathbf{x}$

$\mathbf{x} \leftarrow$ empty list of tuples
a $\leftarrow \{\}$ `// Set of open activities`
for $i \leftarrow 1$ ***to*** $|\mathcal{D}|$ **do**
  $r_i \leftarrow i^{\text{th}}$ row of $\mathcal{D}$ (quantised if necessary)
  if r_i *defines start of new activity* `// e.g. rows 1/3 in Table 1`
  then
    $a \leftarrow a \cup$ activity
  else if r_i *defines end of activity* `// e.g. rows 7/9 in Table 1`
  then
    $a \leftarrow a \backslash$activity
  if *inDeletionSet*(r_i,d) **then**
    continue
  $\widehat{r_i} \leftarrow$ *replaceSensorID*(r_i,m)
  if *inDeletionSet*($\widehat{r_i}$,d) **then**
    continue
  $a_i \leftarrow a$ `// The set of possible activity labels responsible for` r_i
  Add the tuple ($\widehat{r_i}$,a_i) to end of list $\mathbf{x}$
return x

Our software will automatically detect real-valued features and an interface is provided which allows the user to easily quantise these data into user-defined bins.

Label Engineering Finally, the algorithm may operate on the labels. This is useful for a number of reasons. We can see from Table 1 that the data files are annotated with resident identifiers and activity labels, e.g. `R1_Work` specifies that the resident, `R1`, is working. Learning activities with many residents has been treated as a single-task multi-class problem, but it may be more naturally represented in a multi-task learning setting where the resident index and activity define each task. Our data generation software permits a user to define regular expressions for the activity labels. For example, by applying `/^(R\d+)_(.*?)$/\2/` to the labels, the dataset can be anonymised (i.e. resident identifiers are removed from the activity labels). Models learnt on this will learn to only classify activities and not the residents. Similarly `/^(R\d+)_(.*?)$/\1/` will remove the activity from the label yielding a classification problem for identifying the user by their interactions with the sensors.

Anonymising the labels, when applied to a multi-task learning approach, can be seen as "variable sharing" which is a method that can reduce the risk of over-fitting by reducing the number of learnt parameters. Indeed, smart-home

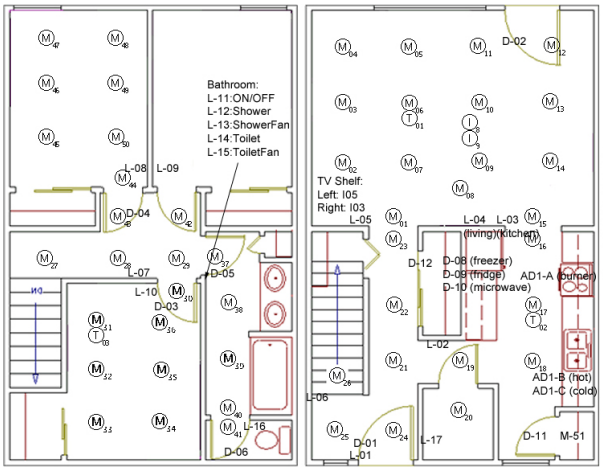


Fig. 1: Floorplan and sensor layout for the `twor.2009` dataset. Sensors are marked as circles.

based activity recognition is most frequently tackled as a sequential classification problem (see later). Using the terminology of Hidden Markov Models (HMMs), this means that emission and transition matrices must be learnt. Given a fixed number of features the size of the emission matrix scales by $\mathcal{O}(|\mathcal{Y}|)$, where $\mathcal{Y}$ is the set of possible labels. Likewise, the transition matrix scales by $\mathcal{O}\left(|\mathcal{Y}|^2\right)$. Therefore, by anonymising a two-resident label-space its size is halved and the transition matrix reduces in size by a factor of four.

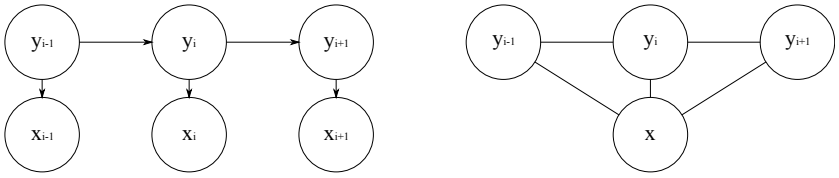
Label modifications are also necessary for transfer learning tasks where it is important that the labels are consistent over all testbeds. With the Centre for Advanced Studies in Adaptive Systems (CASAS) datasets[2] [3], for example, activities will labelled differently depending on the testbed investigated. Specifying expressions such as `/(cooking|meal_preparation)/cook/`, for example, will allow the dataset labels to be represented in a consistent language.

3 Experiments

3.1 Dataset

We use the CASAS datasets for our experiments. CASAS is a research group which primarily focuses on learning activity models for smart homes. For our analysis we investigated the `twor.2009` dataset[3]. This particular dataset was chosen because it poses a difficult multi-class and multi-resident problem. The

[2] CASAS homepage: http://ailab.wsu.edu/casas/
[3] This can be downloaded from: http://ailab.wsu.edu/casas/datasets/twor.2009.zip



(a) Linear-chain HMM. (b) Linear-chain CRF.

Fig. 2: Graphical structure of linear-chain models.

data was collected over a continuous period of three months and is built from unscripted, multi-resident home-living scenarios. In total, 13 activities are labelled, eight of which are personalised to specific residents.

The testbed floorplan and sensor layout are shown in Figure 1. This testbed consists of six families of sensor: 1) 51 motion sensors; 2) 9 door sensors; 3) 7 light switch sensors; 4) 2 water flow sensors; 5) 1 stove-top burner sensor; and 6) 1 item sensor. The motion sensors are PIR sensors that are distributed evenly throughout the testbed, and these are found at intervals of approximately 2 metres. Door sensors can be found at the entrances to the house, on the doors to rooms, and on the cupboards in the kitchen and wardrobes in the bedrooms. The water and stove-top sensors are found in the kitchen; these are Analogue to Digital Converter (ADC) sensors that measure the rotation of a fan or the temperature of a thermocouple.

3.2 Conditional Random Fields

For this work Conditional Random fields (CRFs) [10,15,8] were used for activity classification. CRFs are a sequential-based classification model that learn the conditional distribution of label sequences. As we require that our model only classifies data CRFs were selected over alternative models, such as HMMs, because they directly model the conditional distribution, $p(y|x)$, whereas HMMs model the joint (generative) distribution, $p(y, x)$. Furthermore, CRFs may utilise feature information from more than one single time step when predicting activities. With HMM models, however, such treatments are difficult to model and may violate the independence assumptions due to their methods of factorising probabilities [9,15].

In this work, we restrict ourselves to linear-chain CRFs. Let us formalise our notation by defining the training data, $\mathbf{x}$, as a sequence of N sensor events, and $\mathbf{y}$ to be the sequence of associated activities. Each $\mathbf{x}_i$ in $\mathbf{x}$ has K attributes and $\mathbf{x}_{i,k}$ identifies the k^{th} attribute of the i^{th} example. The graphical structure of linear-chain HMMs and CRFs are shown in Figure 2. Figure 2a shows the HMM structure where we can see that predictions for each time-slice only consider $\mathbf{x}_i$ and $\mathbf{y}_i$. However, Figure 2b, illustrating the graphical structure of a CRF, shows how each $\mathbf{y}_i$ may draw from the whole sequence of $\mathbf{x}$ to classify new examples.

The general model of the CRF is defined by

$$p\left(\mathbf{y}|\mathbf{x}\right) = \frac{1}{Z(\mathbf{x})} \exp\left\{\sum_{j=1}^{J} \mathrm{F}_j(\mathbf{x}, \mathbf{y})\right\}, \tag{1}$$

where

$$Z(\mathbf{x}) = \sum_{\mathbf{y}' \in \mathcal{Y}} \exp\left\{\sum_{j=1}^{J} \mathrm{F}_j(\mathbf{x}, \mathbf{y}')\right\}, \tag{2}$$

is called the partition function and it normalises the output of the CRF to a true distribution by summing over all combinations of $\mathbf{y}'$, and

$$\mathrm{F}_j\left(\mathbf{x}, \mathbf{y}\right) = \sum_{i=1}^{N} \lambda_j f_j\left(\mathbf{y}_{i-1}, \mathbf{y}_i, \mathbf{x}, i\right) \tag{3}$$

is the weighted sum of feature functions, f_i, over the sequence. Each feature function takes as arguments the previous and current labels ($\mathbf{y}_{i-1}$, $\mathbf{y}_i$), the full sequence of observations ($\mathbf{x}$) and the current position of the sequence (i). The reason for the final argument is because each prediction can depend on the entire sequence of observations, $\mathbf{x}$, and therefore the feature functions must be informed about the current position being labelled.

In general, the output of these feature functions is binary. An example of a feature function for activity recognition is given below (where $u, v \in \mathcal{Y}$):

$$f_j\left(u, v, \mathbf{x}, i\right) = \begin{cases} 1 & \text{if } u = \texttt{cook} \wedge v = \texttt{cook} \wedge \mathbf{x}_{i-3,k} = \texttt{m\_kitchen} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

This returns true if there was motion in the kitchen three time steps previously, and if the previous and current labels were both `cook`. The feature functions can be more expressive than sensor activations taken in isolation, as they may consist of a conjunction of tests that span multiple time steps.

Each feature function has a weight, λ_j, associated with it and these weights are learnt through optimisation procedures such as Stochastic Gradient Descent (SGD) or other quasi-Newtonian methods [11]. In each case, the partial derivative of the likelihood function, $\mathcal{L}\left(\mathcal{D}\right)$, is taken with respect to the j^{th} weight which yields

$$\frac{\delta}{\delta \lambda_j} \mathcal{L}\left(\mathcal{D}\right) = \widehat{\mathrm{E}}\left[f_j\right] - \mathrm{E}\left[f_j\right], \tag{5}$$

where $\widehat{\mathrm{E}}\left[f_j\right]$ is the expected value under the empirical distribution of the feature f_j, and $\mathrm{E}\left[f_j\right]$ is the expected value under the model distribution of f_j [9]. This will equal zero when learning has converged, and so the CRF will have learnt a model in which $\widehat{\mathrm{E}}\left[f_j\right] = \mathrm{E}\left[f_j\right]$, which implies that the model has been calibrated

against the training data. Introducing a Gaussian prior to the weights with a bandwidth σ^2 yields

$$\frac{\delta}{\delta\lambda_j}\mathcal{L}(\mathcal{D}) = \widehat{\mathrm{E}}[f_j] - \mathrm{E}[f_j] - \frac{\lambda_j}{\sigma^2}. \quad (6)$$

We choose σ^2 from a cross-validation subset of the training data which is never directly used to learn parameters or during testing.

3.3 Features

We use similar features to those used by other researchers in activity classification in smart homes. We extract five categorical features from the sensor data. The features are 1) the day of the week (7 possible values); 2) the hour or the day (24 possible values); 3) the second-to-last sensor event; 4) the previous sensor event; and 5) the current sensor event. The number of possible values in the last three attributes above depends on the user-specified deletion/merging configuration. Without any deletion/merging expressions 71 sensors are considered.

3.4 Experiments

We perform two types of experiments on the `twor.2009` dataset. These are summarised below:

- Activity recognition. Four different scenarios are investigated:
  - No sensors are removed, and all activity labels are preserved (termed `allSens_allAct`).
  - No sensors are removed, activity labels are anonymised (`allSens_someAct`).
  - Motion sensor activations are merged at the room-level[4], all activity labels are preserved (`someSens_allAct`.).
  - Motion sensor activations are merged at the room-level, labels are anonymised (`someSens_someAct`).
- Sensor removal. Three different scenarios are investigated:
  - Door sensors are removed (`no_door`)
  - Light sensors are removed (`no_light`)
  - Door and light sensors are removed (`no_door_or_light`)

3.5 Performance Assessment

Typically, researchers assess performance on the basis of positive predictive rates [3,5,13]. We use this metric here, but also discuss others. As is shown in Table 1, the dataset annotations are marked with `begin` and `end` statements. When multiple activities are co-occurring, it is unknown exactly which activity is responsible for generating sensor events as this is not explicitly annotated in the database. We, therefore, define the following metrics. The prediction of the i^{th} example, $\tilde{y}_i$, is deemed to be a

[4] With reference to Figure 1, this means that the sensors `m_(15|16|17|18|51)` would be converted to `m_kitchen`. We define eight rooms in total.

– True Positive (TP) if $\tilde{y_i} \in a_i$ (a_i is assigned in Line 16 of Algorithm 1); and
– False Positive (FP)if $\tilde{y_i} \notin a_i$.

We report our accuracy with the precision metric

$$\text{Precision} = \frac{\sum \text{TP}}{\sum \text{TP} + \sum \text{FP}} \tag{7}$$

which can be interpreted as the probability that a predicted label of the i^{th} instance of a dataset is in the set of open activities, a_i, at that time. This is a slightly ambiguous metric because it reports one precision metric for all activities, and because the activity that is truly responsible for the event is unknown.

We also report the False Negative (FN) metric. Given an activity, a, which spans the event indices $l \leq i \leq u$, FNs are counted if there exists no i over the duration of the activity where $\tilde{y_i}$ is predicted as a. The FN rate is then reported by normalising the number of FNs counted by the total number of activities in the dataset. The metric can be interpreted as follows: given that an activity a occurred, the FN rate gives the probability that a is never predicted by the classification model.

4 Results and Discussion

4.1 General Accuracy

Figure 3 shows an example of activity prediction on smart home sensor data[5]. In this figure, the ground truth labels are shown in the upper image and the predicted activity labels of the CRF are shown in the lower image. The colour at any time point is representative of the label that is predicted by the classification model. In the upper image when two colours are shown (e.g. region a, b, c, and d) this indicates that two residents are performing concurrent activities. This image shows good visual correspondence between the ground truth and the predicted activities. Some classification errors can also be seen in the region marked as b and d.

We can also see, at the regions marked by a and c, that the predictions can be relatively inconsistent and "jump" between the labels `watch_tv/meal_preparation` in region a, and between `work/sleep` in region c. The reason for this behaviour is because multiple residents are in the smart home performing different activities at this time. Predictions naturally "jump" between activities at these times because sensor events are dominated, for short time periods, by one activity. The dominant activity then changes, recorded events reflect this and the predictions will then change. Region d marks a FN of the `study` label.

[5] This image is generated using anonymised activity labels in order to increase the clarity for the reader. Similar images can be obtained with non-anonymised data, but the number of labels will have increased.

Fig. 3: Demonstration of the ground truth activity labels (top) and predictions from the CRF (bottom). Co-occurring activities are seen when multiple colours are seen the upper image. Regions marked with a and c identifies co-occurring activities, b shows mislabelled activities and d marks an FN activity.

4.2 Activity Recognition

Table 2 shows the results of the activity recognition experiments described in Section 3.4. In general it can be seen that good classification accuracy is obtained; the precision of the default label is approximately 36%. In all cases the default label accuracy is significantly improved upon. The results show that when the activity labels are anonymised the classification accuracy of activity recognition rises by 9%. This is intuitive because these activities should "look" the same from a sensor perspective regardless of which resident performed the activity. Indeed, it may be very difficult to attribute resident identification to activities, and perhaps specific identification sensors are necessary to achieve this. This is suggested by the fact that in all cases the `someSens` precision and FN metrics are inferior than the `allSens` precision results.

The results of these experiments suggest that, when it comes to activity recognition, recognition gains can be obtained by classifying anonymous activities. By performing this simple task we see that precision metric rose by approximately 9%, which is a reduction in error of approximately 35%. However, it is not surprising that the best results are always obtained when all sensors are considered, as our room-based reduction technique was relatively aggressive and yielded a reduction in overall precision of about 12%.

Interestingly, we can see from the FN column in this table that when the number of sensors in the rooms are reduced the FN rate rises by a factor of 2-3. At these rates approximately one activity in five are never classified. This can be understood by the fact that in the `twor.2009` testbed residents can perform

Table 2: Precision and FN rates for the first experiment.

Experiment Name	#labels	#sensors	Precision	FN Rate
allSens_allAct	13	71	76.2%	8.6%
allSens_someAct	9	71	85.2%	6.7%
someSens_allAct	13	29	64.0%	20%
someSens_someAct	9	29	71.1%	18%

Table 3: Precision and FN rates for the second experiment.

Experiment Name	#labels	#sensors	Precision	FN Rate
no_doors	9	62	82.9%	9.7%
no_lights	9	64	83.6%	10.8%
no_doors_or_lights	9	55	81.7%	9.8%

multiple activities in one room, and with a reduced number of sensors a less expressive description of the room is obtained. We can also see that FN rates seem to be consistently reduced when the activities have been anonymised.

4.3 Sensor Deletion

Table 3 show the classification accuracies that are obtained when door and light sensors are removed. For this task the activity labels were anonymised as this configuration was found to yield the highest precision in the previous allSens_someAct experiment.

It is very interesting to see that both the precision and FN rate are affected by less than 4% in comparison to the allSens_someAct experiment. This result is quite surprising in the no_door experiment because it was assumed that interaction with door sensors would have been important for discriminating between kitchen-centred events such as cooking and washing, for example.

The precisions and FN rates for these three experiments are relatively similar to one another and indeed are similar to the results obtained with the full complement of sensors. This indicates that, when comparing the results to those obtained in the someSens_* experiments, light and door sensors do not appear to be as critical as motion sensors for activity recognition.

4.4 Discussion

The results presented here shed light on the relative importance of the variety and density of sensors found within smart homes. In Section 4.2 it was seen that reducing the localisation accuracy of the motion sensors reduced the overall precision by approximately 13% and that the FN rate rose to 20%. This result

strongly suggests that a high density of motion sensors is very important for good recognition of activities and for ensuring low FN rates in the dataset investigated.

The context modulation software that we developed is capable of easily generating new and diverse contexts of the same testbed. We demonstrated this with a number of experiments where the sensing topology of the testbed has been dramatically modified. In Section 4.3, for example, two classes of sensor were completely removed from consideration by specifying a number of regular expressions in a text file. Even with these, potentially naïve, sensor modulations, however, we were still able to observe interesting trends on the predictive performance of the classification routine.

It is easy to imagine other scenarios that might be investigated. For example, a particular resident may not wish to have sensors in their bedrooms or bathrooms for reasons of privacy. Even though the sensors that we use here are non-invasive and should not make residents feel uncomfortable, our context modulation framework easily yields the ability of defining scenarios which will help us understand the effect that these constraints might have on activity recognition. Furthermore, quantitative assessment of these scenarios can be achieved without having to redeploy new sensor topologies in a smart home or having to gather new data.

5 Future Work

The work described in this paper is still in progress and future work will focus on the following. We will use context modulation as a critical node between the database and learning algorithms. We will allow training instances to be replicated, and each replication can have a randomised context modification. Presenting these to the learning algorithm, perhaps in a bagging or boosting fashion [6], should facilitate the learning of more robust classification models.

Furthermore, we will adapt the data generation and classification pipelines so that multi-task datasets can easily be learnt. Currently we are able to generate multi-task data, and we hope to introduce a mechanism that will permit multiple tasks to be learnt concurrently and, perhaps, to utilise meta classification on top of this. Finally, we hope to release this software to the open source community once it has been completed[6].

6 Conclusions

We presented a software tool that performs context modulation of sensor events and provides a common interface for performing a number of machine learning tasks (e.g. feature construction, multi-task data generation, feature elimination etc.), and all of this is achieved with little effort on the part of a user. This tool offers the opportunity to investigate and understand different aspects of the sensor configuration. When applied to activity recognition in the smart home

[6] Links to the software will be found at: http://www.irc-sphere.ac.uk/

we showed that modulating the context of the data itself can help us understand some aspects of the intricate relationship between the sensor set and the performance of activity recognition algorithms. Most importantly this process is streamlined by our software, which accelerates the understanding process.

The scenarios that were investigated in this paper offer a number of interesting findings, even given the relatively simple context modulation utilised here. In our experiments, we show that the most important sensors appear to be motion sensors, and that a higher densities of these yields highest precision and lowest false negative rates. We found that an absence of door and light sensors seem to affect activity recognition only marginally. We also conclude that identification of residents in smart homes is a difficult task and we speculate that only considering environmental sensors in isolation for this task is, perhaps, not optimal. Our software also provides the latitude to easily redefine datasets in many settings, and that it could be used to generate new data that might help to learn robust and fault-tolerant activity models for smart homes.

On the basis of our experiences using the context modulation software, we envisage that it will greatly facilitate deeper understanding of the relationship between sensors and prediction of activities of daily living in smart homes.

Acknowledgements

This work was performed under the SPHERE IRC funded by the UK Engineering and Physical Sciences Research Council (EPSRC), Grant EP/K031910/1.

References

1. Acampora, G., Cook, D., Rashidi, P., Vasilakos, A.: A survey on ambient intelligence in health care. Proceedings of the IEEE (2013)
2. Agarwal, A., Kataria, S.: Multitask learning for sequence labeling tasks. arXiv:1404.6580 [cs] (2014)
3. Cook, D., Schmitter-Edgecombe, M.: Assessing the quality of activities in a smart environment. Methods of information in medicine 48(5), 480 (2009)
4. Cook, D.J., Holder, L.B.: Sensor selection to support practical use of health-monitoring smart environments. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1(4), 339–351 (2011)
5. Crandall, A.S., Cook, D.J.: Coping with multiple residents in a smart environment. Journal of Ambient Intelligence and Smart Environments 1(4), 323–334 (2009)
6. Flach, P.: Machine learning: the art and science of algorithms that make sense of data. Cambridge University Press (2012)
7. Fleury, A., Noury, N., Vacher, M.: Improving supervised classification of activities of daily living using prior knowledge. International journal of E-Health and medical communications 2(1), 17–34 (2011)
8. Hsu, K.C., Chiang, Y.T., Lin, G.Y., Lu, C.H., Hsu, J.Y.J., Fu, L.C.: Strategies for inference mechanism of conditional random fields for multiple-resident activity recognition in a smart home. In: Trends in Applied Intelligent Systems, pp. 417–426. No. 6096 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (Jan 2010)

9. Klinger, R., Tomanek, K.: Classical probabilistic models and conditional random fields. TU, Algorithm Engineering (2007)
10. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning. pp. 282–289. ICML '01 (2001)
11. Lavergne, T., Cappé, O., Yvon, F.: Practical very large scale CRFs. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. p. 504–513 (2010)
12. Ofli, F., Chaudhry, R., Kurillo, G., Vidal, R., Bajcsy, R.: Berkeley MHAD: a comprehensive multimodal human action database. In: Applications of Computer Vision (WACV), 2013 IEEE Workshop on. p. 53–60 (2013)
13. Rashidi, P., Cook, D.J.: Transferring learned activities in smart environments. In: Intelligent Environments. p. 185–192 (2009)
14. Twomey, N., Walsh, N., Doyle, O., McGinley, B., Glavin, M., Jones, E., Marnane, W.: The effect of lossy ECG compression on QRS and HRV feature extraction. In: Engineering in Medicine and Biology Society (EMBC). pp. 634–637. IEEE (2010)
15. Vail, D.L., Veloso, M.M., Lafferty, J.D.: Conditional random fields for activity recognition. In: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems. p. 235 (2007)
16. Williams, J.A., Weakley, A., Cook, D.J., Schmitter-Edgecombe, M.: Machine learning techniques for diagnostic differentiation of mild cognitive impairment and dementia. In: Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence (2013)