IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received July 12, 2019, accepted July 22, 2019, date of publication August 8, 2019, date of current version August 22, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2933994

Predicting Sensor Events, Activities, and Time of Occurrence Using Binary Sensor Data From Homes With Older Adults

FLÁVIA D. CASAGRANDE[1], JIM TØRRESEN[2], (Senior Member, IEEE), AND EVI ZOUGANELI[1]
[1]Department of Mechanical, Electronics, and Chemical Engineering, OsloMet – Oslo Metropolitan University, 0166 Oslo, Norway
[2]Department of Informatics, University of Oslo, 0373 Oslo, Norway

Corresponding author: Flávia D. Casagrande (flacas@oslomet.no)

This work was supported by the Norwegian Research Council through the SAMANSVAR programme under Grant 247620/O70.

ABSTRACT We present a comprehensive study of state-of-the-art algorithms for the prediction of sensor events and activities of daily living in smart homes. Data have been collected from eight smart homes with real users and 13-17 binary sensors each – including motion, magnetic, and power sensors. We apply two probabilistic methods, namely Sequence Prediction via Enhanced Episode Discovery and Active LeZi, as well as Long Short-Term Memory Recurrent Neural Network, in order to predict the next sensor event in a sequence. We compare these with respect to the required number of preceding sensor events to predict the next, the necessary amount of data to achieve good accuracy and convergence, as well as varying the number of sensors in the dataset. The best-performing method is further improved by including information on the time of occurrence to predict the next sensor event only, and in addition to predict both the next sensor event and the mean time of occurrence in the same model. Subsequently, we apply transfer learning across apartments to investigate its applicability, advantages, and limitations for this setup. Our best implementation achieved an accuracy of 77-87% for predicting the next sensor event, and an accuracy of 73-83% when predicting both the next sensor event and the mean time elapsed to the next sensor event. Finally, we investigate the performance of predicting daily living activities derived from the sensor events. We can predict activities with an accuracy of 61-90%, depending on the apartment.

INDEX TERMS Binary sensor, probabilistic method, recurrent neural network, sequence and time prediction, transfer learning.

I. INTRODUCTION

Activity recognition and prediction are a prerequisite for the realisation of intelligent support functions in smart homes, including functions that support older adults with mild cognitive impairment or dementia (MCI/D) live a safe and independent life at home. MCI/D is a cognitive decline that can affect attention, concentration, memory, comprehension, reasoning, and problem solving [1]. A fair amount of research on smart home functions has aimed at assisting older adults with MCI/D in their everyday life [2]. Examples are prompting with reminders or encouragement [3], [4], diagnosis tools [5], [6], as well as prediction, anticipation, and prevention of hazardous situations [7], [8].

A number of algorithms for activity recognition and prediction have been reported in the literature. However, most of the work in the literature uses data collected in the lab or in testbeds based on scripted activities. In addition, there is no comparative study investigating state-of-the-art algorithms applied to data collected from real homes, different configurations for input of data, limitations, and suitable applications. This is the focus of this work, where we use data collected from real homes, analyze, and compare the performance of state-of-the-art prediction algorithms. The work has been carried out in an interdisciplinary project, the Assisted Living Project (ALP), that involves experts in health, technology, and ethics [9]. The aim of the project is to develop assisted living technology (ALT) to support older adults with MCI/D live a safe and independent life at home.

In this paper, we start our analysis by comparing the performance of state-of-the-art prediction algorithms – probabilistic methods and neural networks – for the prediction of the next sensor event based on previous sensor events. Their performance is assessed with regard to a number of

The associate editor coordinating the review of this article and approving it for publication was Qiuhua Huang.

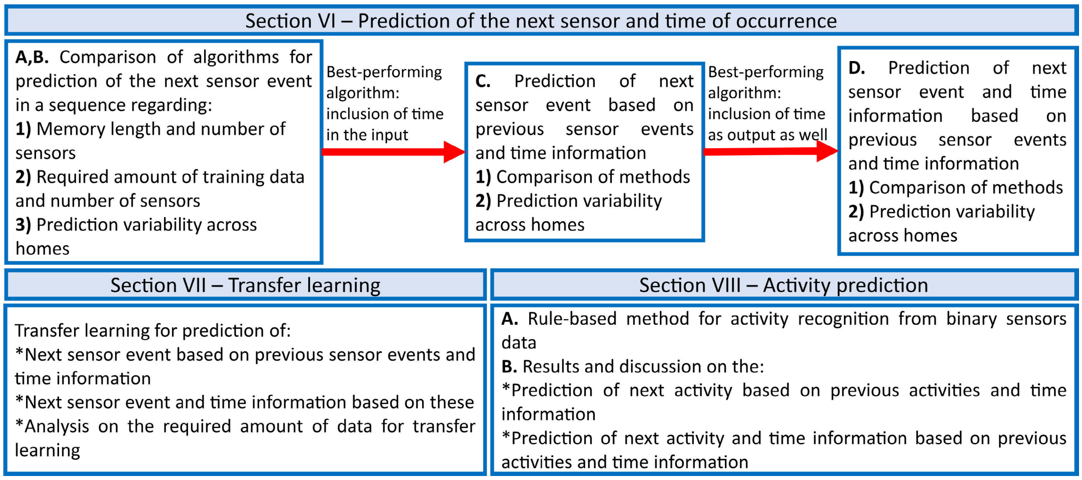

FIGURE 1. **Content of results and discussion sections in the paper.**

factors: the required number of preceding events to predict the next event from (which we refer to as ''memory length''), the necessary amount of data to achieve good accuracy and convergence, and the number of sensors in the dataset. The best-performing algorithm is further improved by including the time of occurrence information in several ways. Part of this work has been previously published [10]–[13], however, using data from one apartment only. We have also examined the prediction accuracy across some of the apartments and the performance when using transfer learning [14]. In the current paper, we expand the analysis to include all eight apartments in the field trial in order to analyze the variability of the prediction accuracy across residents. In addition, we analyze the feasibility of extracting daily living activities from the sensor events and predicting the next activity rather than the next sensor, as well as its time of occurrence.

The paper is organized as follows. Section II gives an overview of algorithms used for sequential sensor event and activity prediction in the literature, of work related to prediction of the time of occurrence, and of transfer learning. Section III presents our field trial, the sensor system in the apartments, and the format of the collected data. Section IV describes the prediction methods, followed by the description of data preprocessing in Section V. Sections VI, VII, and VIII present the results and discussion for the prediction of the next sensor event and its mean time of occurrence, transfer learning, and activity prediction, respectively. These are illustrated in Fig. 1, for better understanding. Finally, in Section IX, we discuss our findings and conclude the paper.

II. RELATED WORK

Activity prediction includes mainly two tasks: sequence prediction and time prediction. Such algorithms can for instance lead to an improved operation of automation functions (e.g. adjust the temperature sufficient time prior to the person waking up); enable the realization of prompting systems (e.g. prompt the resident if the predicted activity has not been performed) [15]; or identify changes and anomalies in certain behaviour patterns (e.g. movement, everyday habits, etc.) and thus indicate the onset or the progress of a condition [16].

A number of algorithms for sequence prediction have been studied in the past years [17]. These algorithms usually train a model based on a sequence of symbols to predict the next symbol. The Active LeZi (ALZ) is a probabilistic method that has been extensively employed for prediction of sequential data [18]. It achieved a peak accuracy of 47% when applied on the Mavlab testbed dataset, that includes 50 binary sensors [18]. The Sequence Prediction via Enhanced Episode Discovery (SPEED) algorithm has been implemented based on ALZ [19]. SPEED was applied on the Mavlab dataset and reached an accuracy of 88.3% when the same dataset was used both for training and for testing. Both algorithms convert the data of binary sensors to a sequence of letters and build a tree based on the observed patterns and corresponding frequency of occurrence. Neural networks have also been used for sensor event prediction with notable performance, typically recurrent neural networks (RNN) [11], [20]–[22]. Three RNN models – Echo State Network (ESN), Back Propagation Through Time (BPTT), and Real Time Recurrent Learning (RTRL) – were applied on a fourteen-day dataset with only six binary sensors (four motion and two magnetic). The ESN performed better with a root square mean error (RMSE) of 0.06 [20]. In these networks, the number of input and output values corresponded to the number of sensors in the dataset, and each assumed value ''0'' or ''1'' for being ''off'' or ''on'' at a certain time slot. The prediction in this case was computed for the next six hours. A similar study was carried out for a 16-room office environment [21]. The dataset in this case was collected through an app installed on the personal data assistant (PDA) of participating employees that had to register manually whenever they entered/left a certain room. An Elman network and a multilayer perceptron network were applied to predict the next room a person would go to. There were four participants in the study and the Elman network attained the best results, ranging from 70% to 91% accuracy, depending on the user. Each room was codified in four bits, as there were 16 rooms in total. The input corresponded to two rooms and the output to the predicted next room. This work also applied other methods – Bayesian network, state prediction, and Markov predictor – where comparable results were achieved [22].

In addition to sequence prediction, these algorithms should also be able to predict when the next symbol (representing either a sensor or an activity) will occur. The time series methods Autoregressive Moving Average (ARMA) and Autoregressive Integrated Moving Average (ARIMA) have been extensively applied in the literature [23]. Nevertheless, they assume the time series to be linear, which is not applicable to activities in a home [24]. Rule-based algorithms have been developed for time forecasting as well [15], [25]. They are quite useful, however they do not account for more complex activities. Non-linear time series models would be more suitable to time prediction in smart homes, e.g. artificial neural networks. A Non-linear Autoregressive Network (NARX) was compared to an Elman network to predict a sensor activation's start and end time [26]. In this study, each sensor had its own network trained and tested on a twenty-day dataset with six binary sensors. The NARX performed better, with a RMSE ranging from 0.06 to 0.09, depending on the sensor. Decision trees have been used to predict the time a certain activity would happen [24]. This method relies on several features extracted from sensor events sequences. It was applied on a dataset with 51 both binary and sampling sensors and achieved an average normalized RMSE of 0.01. Bayesian networks have been used to predict the next location, time of day, and day of the week a person would execute an activity [27]. This algorithm was employed in two apartments with about 30 binary sensors each, where the next location was predicted with 47% and 61%. Poisson process has also been applied to predict the time an observed activity would occur [28]. An RMSE of 3.9431 seconds was achieved in this work.

Taking into account that each individual has their unique habits, and smart homes may have different layouts and limitations for deployment of sensors, it is important that the prediction algorithm is able to adapt to each home and resident. Transfer learning can reveal whether the algorithm can adapt. This technique consists of training and learning parameters from a source dataset that is different yet related to a target dataset (e.g. different labels and data distributions [29]). Transfer learning has been used in several fields, e.g. image and language classification, computer networks, automated planning, mathematical problems, and activity recognition [29], [30]. This method has proved to provide many advantages. For instance, it allows that datasets with different feature spaces can transfer the knowledge between each other [31], [32]. In addition, transfer learning can dramatically decrease the required amount of data in the target dataset, as proved for a mortality prediction algorithm [33] and for activity recognition [32], [34]. Besides, it can be applied in combination with several algorithms: RNNs [35], Hidden Markov Models [36], statistical inference [33], support vector machine [34]. In smart homes, a cross-domain activity recognition algorithm combined with transfer learning and a similarity function between different activities was proposed [34]. In that work, three different datasets were used, where one was collected over 28 days from a real home of a 26-year-old man. A peak accuracy of 65% was achieved with seven activities. Another work transferred the knowledge of activities from multiple physical source spaces to a different target physical space [32]. The authors propose an algorithm that maps automatically activities from source to target environment and classifies the activity based on a weighted majority vote method. The data contained 5 to 11 activities, and were collected from six testbeds where volunteers lived for 2-3 months. A peak accuracy of about 80% was reported. Hidden Markov Models and transfer learning have also been combined and used across three apartments with five recorded activities and achieved a F1-score of 0.65 in the best case [37]. Transfer learning has its limitations. It has been shown that it can either improve or degrade the prediction accuracy of models depending on the dataset used for transfer, which is known as negative learning [29]. In these cases, it is important to detect which is the best source dataset to a problem, for example using Dynamic Time Warping to measure inter-dataset similarities [38].

Most datasets in the cited works were collected through scripted activities primarily in lab environments, whereas our dataset has been collected in real homes. It contains events from 13-17 binary sensors, i.e. twice as many as used in [20], [26], and less than one third of the number of sensors used in the Mavlab testbed [18]. The number of sensors is comparable to the work in [22] (16 rooms), however in that study the events were inserted by each user in their PDA rather than being generated using sensors, which may lead to a dataset with less artifacts. To our knowledge, no previous work has carried out a comparison of the performance of state-of-the-art sequence prediction algorithms, moreover applied to real data, nor have LSTM networks been previously used for the prediction of sequential sensor events, including the use of transfer learning. In addition, we predict both the next sensor and the mean elapsed time of occurrence within the same model. From the works cited above, [27] is the closest to ours in the sense that it predicts both the next event and its time information in the same model. That work predicts the next location, time of day (slots of 3 hours through the day), and day of the week using a Bayesian network with reported accuracy of 46-60%, 66-87% and 89-97%. Subsequently, the activity is predicted with an accuracy of 61-64% based on a combination of these features. The authors use data from testbeds collected over 6 and 4 months, and take into account 10 locations and 11 activities. Our work predicts the next sensor event and the time of occurrence for a set with about 15 sensors with better overall accuracy. In addition, activities are predicted with considerably higher accuracy.

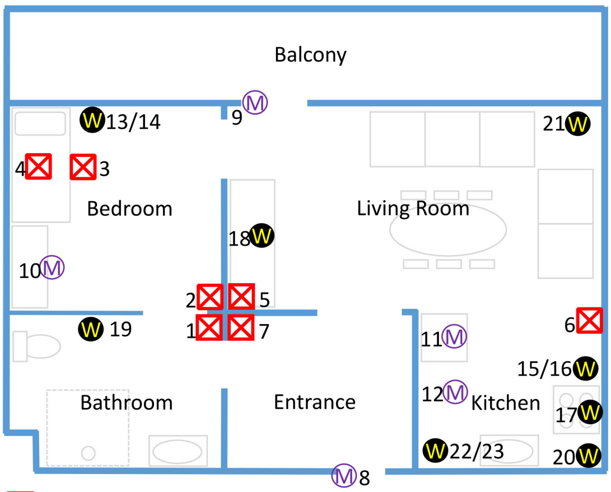

FIGURE 2. Proposed sensors system for field trial apartments.

III. FIELD TRIAL

Our field trial includes eight residents over 70 years old in a community care facility. The apartments have similar layouts – comprising a bedroom, a living room, an open kitchen area, a bathroom, and an entrance hall (Fig. 2). The purpose of the trial and the sensor system to be deployed have been decided upon in close collaboration with the residents [9]. A minimal number of binary sensors was installed in the apartments to minimize surveillance of the residents and comply with the technical and economical constraints imposed by the project. The set of sensors has been chosen so that it can potentially identify daily activities and possibly enable the realization of useful functions for older adults with MCI/D as these were indicated at dialogue cafés with the users [9]. Hence, our set of sensors contains motion, magnetic, and power sensors. These generate events that are able to indicate occupancy patterns (movement around the apartment), daily activities – kitchen related activities, dressing, being in bed –, and leisure activities – reading, watching TV, listening to radio. Motion sensors (Pyroelectric/Passive Infrared – PIR) detect motion through the change of the infrared radiation in its field of view. It generates an event with message "1" every time a motion is detected, otherwise it sends no event. In our dataset, we had to insert the "off" events ("0" message) so that the data are consistent for all sensors. Magnetic sensors indicate whether doors, windows, and drawers are open or closed, by generating events with messages "1" and "0", respectively. Power sensors measure the electricity usage of a certain appliance, and can therefore indicate whether it is turned on or off, and generate events with messages "1" and "0", respectively.

TABLE 1. Set of Sensors in each Apartment (complementing the standard set of motion sensors).

Apt. ID	Sensors
1	P[a]: night stand lamp, coffee machine, living room/reading lamp, TV; M[b]: cupboard/drawer, entrance door
2	P[a]: night stand lamp, coffee machine, living room/reading lamp, microwave, TV; M[b]: fridge, entrance door
3	P[a]: kettle, living room/reading lamp, microwave, toaster; M[b]: fridge, cupboard/drawer, entrance door
4	P[a]: night stand lamp, coffee machine, living room/reading lamp, TV; M[b]: fridge, entrance door
5	P[a]: kettle, TV; M[b]: fridge, cupboard/drawer, entrance door
6	P[a]: night stand lamp, coffee machine, kettle, living room/reading lamp, TV, microwave
7	P[a]: night stand lamp, coffee machine, kettle, living room/reading lamp, TV; M[b]: wardrobe, cupboard/drawer, entrance door
8	P[a]: night stand lamp, TV; M[b]: wardrobe, entrance door

[a]Power and [b]magnetic sensors.

TABLE 2. Sample of Binary Sensors Data.

Timestamp	Sensor ID	Sensor message
01.09.2017 07:58:05	2	1
01.09.2017 08:00:14	12	1
01.09.2017 08:01:01	4	1
01.09.2017 08:02:56	5	1
01.09.2017 08:03:05	12	0

Not all apartments could have the exact same set of sensors due to physical limitations (e.g. fridge door with too big gap to enable the use of a magnetic sensor) and/or different equipment (e.g. some residents have a coffee machine, others have a kettle). However, all the participants had the same initial proposal of set of sensors, as shown in Fig. 2. The eight apartments that provided data to this work have installed all the motion sensors, while the rest of the sensors vary between apartments, as summarized in Table 1.

The sensors are connected wirelessly through Z-Wave and xComfort protocols to a Raspberry Pi 3, which transfers the data for storage in a secure server. The data comprise timestamp (date and time with precision of seconds), sensor ID, and sensor message (binary). Table 2 shows a sample of the data collected.

IV. PREDICTION METHODS

This section describes the prediction methods applied in this work, probabilistic methods – Active LeZi (ALZ) and Sequence Prediction via Enhanced Episode Discovery (SPEED) – and recurrent neural network (RNN) with long short-term memory (LSTM). The probabilistic methods convert the data acquired from the sensors into a sequence of letters and identify sequence patterns. The patterns and their frequency of occurrence are used to generate a tree, which is then used to calculate the next most probable event to occur. This last step is performed by the Prediction Partial Matching algorithm (PPM) [39], [40]. The same converted data is used as input for the LSTM networks that are configured as text generation networks in this case.

Table 3 presents a possible scenario in our smart home with actions performed by the resident and the corresponding

TABLE 3. Actions scenario.

Action performed	Activated sensor
Wake up	PIR bedroom (on)
Go to living room	PIR living room (on)
Turn on TV	Power TV (on)
Go to kitchen	PIR kitchen (on)
Turn on coffee machine	Power coffee machine (on)
Go to living room and watch TV while coffee is being made	PIR living room (on)
Go to kitchen	PIR kitchen (on)
Turn off coffee machine	Power coffee machine (off)
Go to living room	PIR living room (on)

TABLE 4. Assignment of letters to sensors.

Sensor	Letter
PIR bedroom	a/A
PIR living room	b/B
Power TV	c/C
PIR kitchen	d/D
Power coffee machine	e/E

sensors triggered. As dictated by ALZ and SPEED, each sensor is assigned with a letter, as shown in Table 4.

A. ACTIVE LEZI

ALZ is a sequence prediction algorithm based on a text compression algorithm [18]. The input in ALZ consists of a sequence of lower-case letters, where each letter represents event from one sensor. For example, the sequence corresponding to the scenario described in Table 3 would be "abcdebdb". ALZ uses the procedure dictated by the LZ78 text compression algorithm to generate patterns that occur in a sequence and create a tree with these and their frequencies [41].

A given sequence $x_1, x_2, \ldots, x_i$ is parsed into n_i subsequences $w_1, w_2, \ldots, w_{ni}$ such that for all $j > 0$ the prefix of the subsequence w_j is equal to some w_i for $1 < i < j$. For example, if we have the sequence "abcdebdb", the patterns found by LZ78 would be "a", "b", "c", "d", "e", "bd". In addition, ALZ generates more patterns from their suffixes, if possible. For example, "bd" would also generate "d". This accounts for patterns that were not perceived by the LZ78 algorithm and that are possibilities in a smart home environment. This increases the convergence rate of the model [18].

When the sequence is parsed completely and the patterns are derived from it, their frequency of occurrence is counted. An order-k-1 Markov tree is then constructed based on the patterns and their frequencies, where k corresponds to the longest pattern found in a training sequence. Then PPM is used to calculate the next most probable event. The generated tree for the example scenario with sequence "abcdebdb" is shown in Fig. 3.

B. SEQUENCE PREDICTION VIA ENHANCED EPISODE DISCOVERY

SPEED is also a sequence prediction algorithm that is based on the occurrence of frequent patterns in home environments [19]. SPEED builds on the same procedure of ALZ, however, it introduces a different method for finding patterns in the sequence. SPEED defines an *episode* as the sequence between an initial and ending point of an activity. For example, the moment a coffee machine is turned "on" is the initial point of a coffee making episode, which lasts until the coffee machine is turned "off". An "off" event cannot happen unless an "on" event has preceded it. Therefore "off" events always happen after an "on" event of the same activity (or sensor), and vice-versa.

The data received from the sensors in the smart home are represented as a sequence of letters, where upper-case letters represent a sensor's "on" event and lower-case letters represent a sensor's "off" event. The sequence representing the example scenario presented in Table 3 would be "AaBCbDEdBbDedB".

The main idea of the SPEED algorithm is to extract episodes from a sequence of data and derive patterns from them. These patterns are used to generate a decision tree that keeps track of the learned episodes and their frequencies. The height of the tree is the length of the longest episode found in the sequence, defined as the maximum episode length. For every event in a sequence, the algorithm searches for its opposite event in the window and if it exists, an episode was found. In the previous sequence, the first episode found is "Aa", the patterns generated from it would be "A", "a" and "Aa". We keep track of these and count their occurrences to generate an order-k-1 Markov model, where k is the maximum episode length. A tree for the example sequence is presented in Fig. 4. Finally, the PPM algorithm is used for prediction.

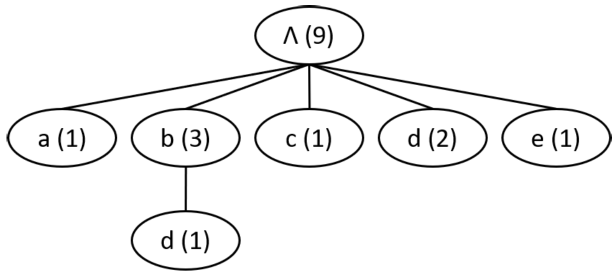

FIGURE 3. Tree generated by the ALZ algorithm for the sequence "abcdebdb".

C. PREDICTION PARTIAL MATCHING ALGORITHM

The PPM algorithm calculates the probability distribution of each possible event based on a given sequence by taking into consideration the different order Markov models with different weights [39], [40]. The weights are given by the escape probability, which allows the model to go from a higher-order to a lower one. The advantage of PPM is that it assigns a greater weight to the probability calculated in higher-order models if the symbol being predicted is actually found in the tree [18]. The predicted symbol is the one with the highest probability.

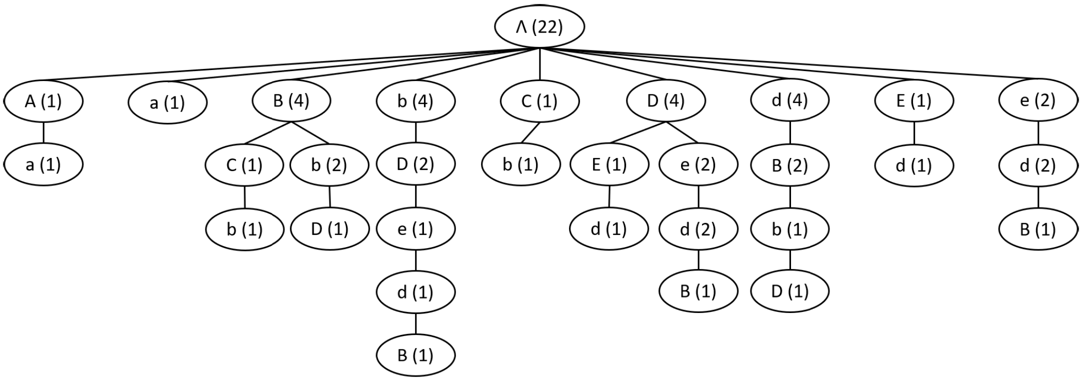

FIGURE 4. Tree generated by the SPEED algorithm for the sequence "AaBCbDEdBbDedB".

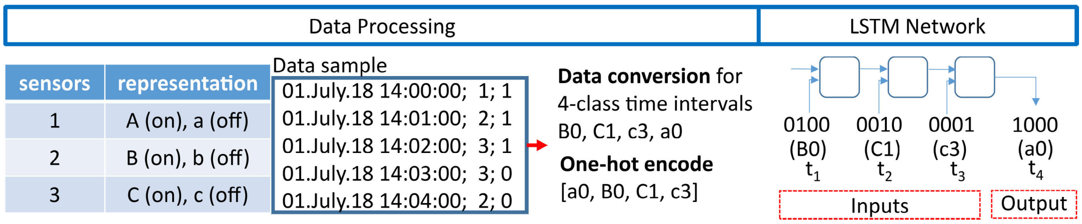

FIGURE 5. LSTM network configuration.

ALZ and SPEED use slightly different strategies of PPM. ALZ uses the exclusion strategy, which means the prediction is performed with the suffixes of the given sequence, except the sequence itself. Therefore, in the case of the sequence "bd", the patterns used to calculate the probability of each letter being the next would be "b" and the null context. Suppose we want to calculate the probability of having a "c" after "bd" using ALZ, based on the tree in Fig. 3. The probability would be given by (1): in an order-2 model, the probability of having a "c" after a "b" is 0/3 and we escape to the order-1 with 2/3 probability. In order 1, the probability of having a "c" after a null context is 1/9.

In the case of SPEED, the patterns used for calculating probabilities after a certain sequence would be all the suffixes, including the sequence itself. Suppose we have the sequence "dB". We would use patterns "dB", "d" and the null context. The probability of having a "b" after this sequence based on the tree in Fig. 4, would be given by (2): we start in order 2 model, where the probability of having a "b" after "dB" is 1/2 and escape to the lower order with probability 1/2. In order-1, the probability of having a "b" after "d" is 0/4 and we escape to the lower order with probability 2/4. Finally, in the lowest order, the probability of "b" after a null context is 4/22.

$$p(c, bd) = \frac{0}{3} + \frac{2}{3}\left(\frac{1}{9}\right) = 0.074 \tag{1}$$

$$p(b, dB) = \frac{1}{2} + \frac{1}{2}\left(\frac{0}{4} + \frac{2}{4}\left(\frac{4}{22}\right)\right) = 0.545 \tag{2}$$

D. LONG SHORT-TERM MEMORY NETWORK

RNN [42] is a neural network that has the property of keeping an internal memory, and has therefore been widely applied to inputs that are sequential in time [43], [44]. The LSTM [45] is a type of RNN designed to be better at storing and accessing information than the standard RNN.

We employ an LSTM network configured as a text generation network. The number of inputs is a certain number of sensor events – equal to the memory length – and the output is the predicted next event in the sequence (Fig. 5). The input and output are one-hot encoded. In the one-hot encoding representation, each symbol is represented by a vector of bits of length equal to the number of symbols in a sequence. All values are zero, except for the one corresponding to that symbol (Fig. 5).

A stateless LSTM network model was implemented in Python 3 using Keras open source library for neural networks. A number of parameters were tuned in order to find the optimal values. The model has one hidden layer with hyperbolic tangent activation and 64 neurons. Our batch size (i.e. number of samples used for training each iteration of the epoch) was 512. We used *Adam* as the optimization function with learning rate of 0.01 and categorical cross-entropy as loss function. The output layer was a softmax activation function. We used the early stopping method and dropout rate of 50% to avoid overfitting, allowing a maximum of 200 epochs for each model's training. In addition, during the training process we use weights for each sensor to balance the number of samples for each sensor. These are computed using the "`compute_class_weight`" function of the Scikit-learn

TABLE 5. Re-labeling of Sensors.

New labels	Sensors
Kitchen sensor	P[a]: toaster, microwave; M[b]: fridge, cupboard/drawer
Beverage sensor	P[a]: coffee machine, kettle

[a]Power and [b]magnetic sensors.

open source library. The weight corresponds to the total number of samples divided by the number of occurrences of the class.

V. DATA PREPROCESSING

A. SENSORS MAPPING

As described in Section III, some power and magnetic sensors differ within the eight apartments (Table 1). In the tests where we compare the prediction accuracy and transfer the learning across the apartments, we re-label the sensors that refer to the same activity. The new labels and the sensors assigned to these are shown in Table 5. Lamp power sensors and wardrobe door magnetic sensors' events were removed from the datasets since we did not manage to assign them to an activity that was common for most of the apartments.

B. DATA CORRECTION

Data acquired from binary sensors often contain faulty events e.g. erroneous activation of motion sensors by sunlight and switch-off delays of motion sensors [46]. Such noise can significantly affect the performance of the models. Hence, we have carried out a data correction preprocessing as follows. Occasionally the motion sensors do not send an activation event when they should. We therefore insert missing events to correct the data. For example, it is not possible to go to the bedroom directly from the kitchen without passing through the living room. When the living room activation event is missing, it is inserted. If there are two possible sensor events (e.g. two possible paths in the apartment), the choice of the inserted sensor event is done such that the final percentage distribution of the two options remains as observed in the original data. The time of the inserted event is the mean between that of the previous and of the next event. This does not compromise the data accuracy because the faulty events usually take place between relatively fast motions around the apartment, which means that the elapsed time between the events is quite short.

C. DATA CONVERSION

The corrected data are subsequently converted to both ALZ- and SPEED-text sequences, as explained in Section IV. The time inclusion was performed as follows. In all cases the generated sensor events are treated as independent events. In the case of the one-hot encoding for the LSTM, our input vector has as many values as the number of symbols in the sequence. For 15 sensors, we have 30 inputs to represent the "on" and "off" states of each of these.

1) SENSOR EVENT AND PERIOD OF DAY

In this case, we distinguish between four periods of the day: morning (from 7am to noon), afternoon (from noon to 6pm), evening (from 6pm to 10pm), and night (from 10pm to 7am). This is indicated by a number between 0 and 3 that is added to the letter that represents the event. For instance, an event of the motion sensor in the bedroom going "on" in the morning would generate the symbol "A0". E.g. when the time of day is taken into account, the number of inputs to the LSTM is multiplied by 4 (120 inputs in total) and similarly in the other cases. These are treated as independent events.

2) SENSOR EVENT WITH TIME ELAPSED TO THE NEXT EVENT

When predicting the next sensor event only, we use together with the sensor's letter a number that indicates the time elapsed to the next event. We define a set of 4-class time intervals: [$<$ 1min, 1-15min, 15min-1h, $>$ 1h]. Hence, we assign numbers 0-3 to the event. For example, if the motion in the bedroom (assigned letter a/A) were activated in the morning and 10 minutes later the person went to the bathroom, the generated symbol would be "A1".

3) SENSOR AND K-MEANS TIME-CLUSTER WITH HOUR OF THE DAY AND ELAPSED TIME TO THE NEXT EVENT

We apply an unsupervised learning method to cluster the sensor samples, where the K-means algorithm clusters each sensor event according to the hour of the day it has occurred and the time elapsed to the following sensor event. In the K-means algorithm, the samples of each sensor are classified into K clusters such that the sum of square distances (SSD) within the clusters is minimized [47]. Each cluster contains a centroid, given by the mean value of each feature of the algorithm. We perform K-means for a number of clusters (K) between 1 and 8 and choose the best K manually according to the elbow method [48]. This method consists of plotting an SSD vs. K graph and choosing the K that resembles an "elbow" (the point of inflection on the curve), which is the best fit for that problem. Fig. 6 shows an example of clustering the samples of the motion sensor in the kitchen. This sensor results in four clusters (represented by the different colors) – chosen by the elbow method based on Fig. 7. Suppose this sensor is represented by letter B, has had an "on" event at noon (blue cluster), and the next sensor event took place 3 minutes later. This would generate "B2" (where 2 represents the blue cluster).

VI. PREDICTION OF THE NEXT SENSOR AND TIME OF OCCURRENCE

Table 6 shows the number of sensor events in the dataset of each apartment and the number of days it has been collected. This section is organized as follows. We first explain the training and testing procedure for all methods. Subsequently, we perform tests for (i) predicting the next sensor event based on past sensor events, (ii) predicting the next sensor event based on past sensor events and time of occurrence

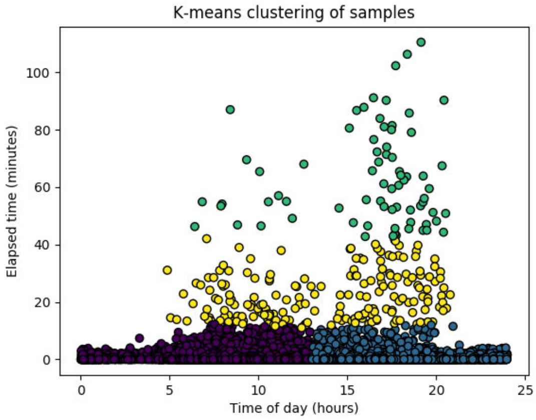

FIGURE 6. K-means clustering of samples of motion sensor events in the kitchen.

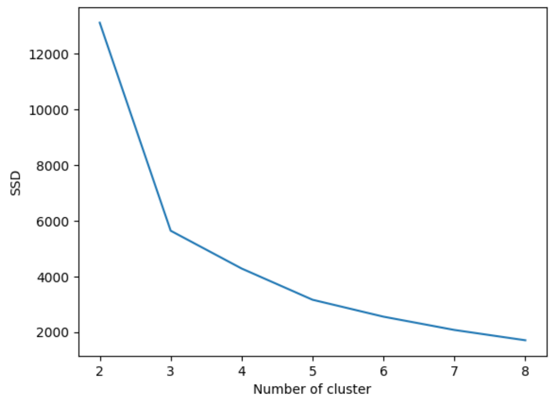

FIGURE 7. SSD vs. number of clusters for motion sensor events in the kitchen.

TABLE 6. Number of events per apartment.

Apt. ID	Number of Events	Number of Days
1	219921	358
2	137396	311
3	37108	163
4	147618	385
5	189468	260
6	19766	96
7	28129	75
8	21949	75

information, and (iii) predicting both sensor event and time of occurrence information based on input including these. In (i) the performance of the four algorithms is tested against a number of factors: the memory length, the amount of data required for good accuracy, and the number of sensors in the dataset. The best-performing algorithm is then further developed for tests (ii) and (iii), where we compare the methods and analyze the accuracy variability across apartments.

A. TRAINING AND TESTING CONFIGURATION

In the SPEED algorithm, the next event is predicted based on the last sequence of events with length equal to the maximum episode length [19]. In the work in [19], the authors use the same dataset for both training and testing, which may lead to overfitting and, in addition, may not lead to a generalized model that can be used on other datasets.

We have modified the testing procedure for both ALZ and SPEED by calculating the optimal number of last events to base the prediction on, i.e. the number of events that leads to the maximum overall prediction accuracy, which we refer to as the optimal memory length. Memory lengths up to the maximum pattern found have been considered. In a previous paper [12], we applied the SPEED method on our data that were obtained from one of the apartments reported over a period of two weeks. When using the same procedure as in [19], we achieved an accuracy of 82% – compared to 88% on the Mavlab dataset. When splitting the data into training (60%), validation (20%), and testing (20%), and optimizing the memory length as described above, we achieved an accuracy of 75% on our data obtained from a real home over two weeks. Similarly for ALZ we obtained 73% (compared to 47% in [18]) when using the same dataset for training and testing, and 53% when using separate datasets for training, validation and testing, and optimizing the memory length as described above. Hence, we use this modified method for SPEED and ALZ in the following sections.

In the case of SPEED and ALZ, the training set is used to build the tree, the validation set is used to find the optimal memory length, and the testing set is used to compute the model's accuracy. In the LSTM networks, the training set is used to train the network, the validation set is used for tuning the parameters and the testing set to calculate the accuracy. All models were trained based on a certain number of events, validated on 3000 random events, and tested on 3000 random events. This process is repeated three times and the accuracy values in the graph correspond to the mean accuracy. The fact that the testing set is always random produces some instability in the accuracy when the model is trained with little data, which is evidenced by the instability shown in the lower range in some of the graphs.

B. PREDICTION OF THE NEXT SENSOR EVENT BASED ON PAST EVENTS

1) CHOICE OF MEMORY LENGTH

We examine the accuracy achieved on the validation set for values of memory length ranging from 1 to 30 events. This is performed first for the dataset of apartment 1 that contains events from fifteen sensors (including magnetic, power and motion sensors) – Fig. 8 – and then for the dataset containing only the seven motion sensors – Fig. 9.

When using the dataset with fifteen sensors (Fig. 9), ALZ achieved a best accuracy of 69.15% while SPEED reached 79.87%. The optimal memory length was four events for ALZ and three for SPEED. The LSTM networks achieved accuracies of 72.02% and 84.12% when using ALZ- and SPEED-text, respectively. In both cases the optimal memory length is equal or larger than eight. The larger optimal

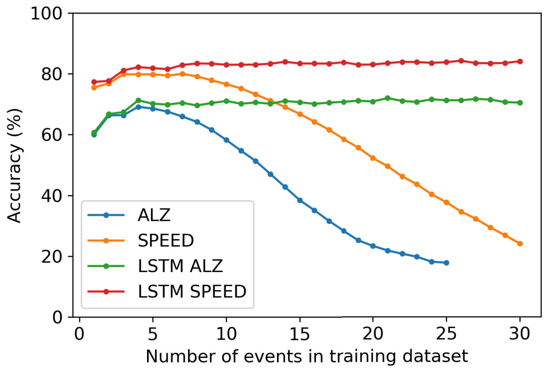

FIGURE 8. Accuracy vs. memory length for all algorithms on a dataset with all fifteen sensors.

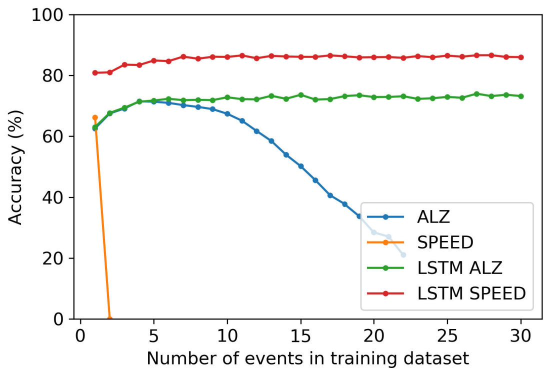

FIGURE 9. Accuracy vs. memory length for all algorithms on a dataset with seven motion sensors.

memory length for LSTM indicates that these are very efficient at detecting patterns and correlations over a longer sequence, in opposition to probabilistic methods.

It is also interesting to notice how the accuracy is affected by memory lengths larger than the optimal. The accuracy of the probabilistic methods drops substantially as the memory length gets larger. In contrast, the LSTM networks roughly stabilize at the peak accuracy for larger memory length values than the optimal. A reason for this is that probabilistic methods are based on certain patterns happening quite frequently. Since our dataset has few sensors, short patterns are more likely to happen more often, and therefore they provide better predictions. The LSTM, on the other hand, has the ability to find patterns in long sequences and can therefore predict the next event based on many past events and longer term patterns and dependencies. Increasing the memory length further does not improve the accuracy, however, which can imply that the model has reached its best performance for this configuration.

Subsequently, we compare the accuracy results of the dataset with fifteen sensors (Fig. 8) to the accuracy results for the dataset that contains only the seven motion sensors (Fig. 9). The accuracy curves for the LSTM network models show a similar dependency to memory length. The optimal memory length is eight or larger. The LSTM with SPEED-text achieves 86.64% while with ALZ-text achieves 74.00%.

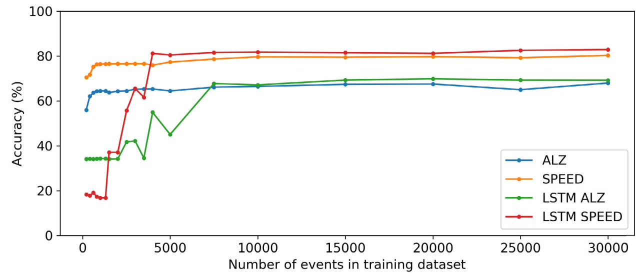

FIGURE 10. Accuracy vs. size of training set for all algorithms on the dataset of apartment 1 with all sensors (15).

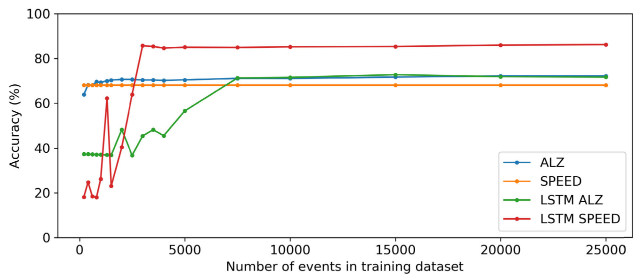

FIGURE 11. Accuracy vs. size of training set for all algorithms on the dataset of apartment 1 with seven motion sensors.

The ALZ method also shows similar behaviour, and the same optimal memory length of four, with a peak accuracy of 71.45%. SPEED presents a very peculiar behaviour. The maximum memory length is two. This is a consequence of the fact that SPEED builds the tree based on episodes, and the longest episode in this case is two events. For example, if the resident would go from the bedroom to the living room and then to the kitchen, the resulting sequence would be "AaBbCc". There are no intertwined events, since when one motion sensor activates, another deactivates. Hence, the "off" events are easily predicted. When it comes to "on" events, the sensor that is most frequently activated will always be the one predicted to activate next, leading to lower accuracy for the "on" events.

2) ACCURACY PER TRAINING SET SIZE

In the following, we investigate the behavior of the accuracy with respect to the size of the training dataset. The accuracy results are computed using the optimal memory length found in the previous analysis. Fig. 10 and 11 show the results when the algorithms are applied to the dataset with all fifteen sensors and with seven sensors, respectively. Since there is no significant improvement in the accuracy for larger datasets, we show the plots for training dataset sizes up to 30000 events for better clarity on the low range of the graph.

We first examine the accuracy in the dataset with all sensors (Fig. 10). A peak accuracy of 83.26% was achieved by LSTM with SPEED-text, while the SPEED algorithm achieved a peak accuracy of 80.65%. The accuracy achieved by the LSTM with ALZ-text was considerably lower at 70.43%. In this case, stability is achieved much later than with the other methods. Finally, the ALZ method reached a peak accuracy of 68.00%. Note that the probabilistic methods attain a good accuracy (close to their peak accuracy) with only 1000 events in the training set. By comparison, the LSTM with ALZ- and SPEED-text require 7500 and 4000, respectively.

Next we examine the accuracy results for the dataset using only the seven motion sensors (Fig. 11). As seen in the previous analysis, the top accuracy is higher since there are fewer sensors in this set. Moreover, motion sensor events happen sequentially, without intertwined events. Hence "off" events can be predicted more easily. The LSTM with SPEED-text achieved an accuracy of 87.21%, by far the best among the methods. The stability was achieved with about 4000 events. Stability is reached with a similar amount of data compared with the case in Fig. 10. The LSTM with ALZ-text and the ALZ achieved very similar accuracies of 73.24%, and 72.32% respectively. The SPEED method, however, achieved a poor accuracy in this case. This is due to the short memory length and lack of intertwined events, as discussed when presenting Fig. 9. Also here, it is confirmed that probabilistic methods require a rather small amount of data to achieve a considerable accuracy, close to the peak accuracy that can be reached by these methods.

3) PREDICTION VARIABILITY ACROSS APARTMENTS

In the following, we apply the two prediction methods with higher accuracy – LSTM with SPEED-text and SPEED – on the dataset of each apartment of our field trial. In this case, we perform the mapping (Section V-A) so that the comparison is fair. Table 7 presents the obtained results. SPEED achieved accuracies in the range 74-82% and LSTM with SPEED-text in the range 75-85%. In all cases, the LSTM had an accuracy 1.5-5% higher than SPEED, with one exception (apartment 4), where the accuracies are about the same. On the other hand, in most of the apartments SPEED required less events for a good accuracy and convergence of the model. We noticed that apartments 6, 7, and 8 have have not achieved stability completely yet as the curves keep rising, indicating that higher accuracy can be achieved. They are indeed the apartments with less collected data (Table 6).

It is interesting to notice that SPEED presents less variability across the apartments. This may be due to the fact that SPEED builds a tree where the predictions will be based on the patterns that happen more often, and these are in fact similar to all the apartments since they have similar layouts. The LSTM network, however, is better able to adapt to the resident in this case, taking into account also patterns that do not happen often.

4) SUMMARY AND DISCUSSION

We have compared the performance of two probabilistic methods – ALZ and SPEED – with LSTM networks using ALZ-text and SPEED-text in apartment 1. The best accuracy was achieved by the LSTM network with SPEED-text, 83% with all the fifteen sensors and 87% with seven motion sensors.

TABLE 7. Prediction accuracy of the next sensor event.

Apt. ID	Top Mean Accuracy (Number of Events for Convergence)	
	SPEED	LSTM with SPEED-text
1	82.24% (2000)	83.66% (5000)
2	77.67% (2000)	79.13% (5000)
3	82.21% (3000)	84.54% (3000)
4	75.01% (2000)	75.42% (4000)
5	76.64% (3000)	79.17% (4000)
6	74.60% (5000)	76.13% (4000)
7	81.78% (7500)	82.20% (3500)
8	79.95% (7500)	84.50% (3000)

The probabilistic methods achieved a high prediction accuracy (close to their peak accuracy) with a relatively small amount of training dataset (about 1000 events). LSTM networks required a larger training dataset (about 4000 event with SPEED-text and 7500 events with ALZ-text) to reach an accuracy close to the peak. Also, probabilistic methods are found to base the prediction on a relatively small number of previous events – an optimal memory length of four for ALZ and three for SPEED was established. On the other hand, LSTM networks base the prediction on a sequence of eight previous events or more. This indicates that such networks are better at finding longer-term dependencies and patterns in a sequence of events. In addition, in the LSTM the attained accuracy is quite stable for memory lengths that are larger than the optimal. On the other hand, probabilistic methods have an optimum memory length, hence the accuracy decreases both for shorter and for longer memory lengths than the optimal.

For the dataset containing events from the fifteen sensors, our best result was achieved by the LSTM network with SPEED-text (83%). SPEED achieved only 2% lower accuracy, however, after considerably longer training time [10]. Hence, in applications where it is an advantage to model with a small amount of data where in addition execution time is not too critical, SPEED may be a good choice, since it can achieve an accuracy close to its peak with little data. In general, our results have shown that it is possible to achieve good accuracy with much less data than thought previously. SPEED and LSTM with SPEED-text achieve better results than ALZ and LSTM with ALZ-text. This is not surprising since the conversion of data to SPEED-text sequences contains more information (both "on" and "off" events). This can also be confirmed by the trees formed by ALZ and SPEED (Fig. 3 and 4).

For a dataset with no intertwined events though – the case of our dataset with only the seven motion sensors – the best choice is the LSTM with SPEED-text. SPEED does not work well in this case, since the tree has a height of two so that only "off" events can be predicted reliably.

Another interesting finding is that when applying these algorithms in different apartments, LSTM with SPEED-text has shown a larger range of accuracies. This indicates that the

LSTM can adapt better to the different patterns in the home of each resident than SPEED does. This fact, in addition to the higher accuracy and the shorter execution time, have shown that the LSTM network with SPEED-text is the best model for our smart homes setup. We therefore further develop only this method in the following analysis.

C. PREDICTION OF THE NEXT SENSOR EVENT BASED ON PAST EVENTS AND TIME INFORMATION

It is important to observe in the results of the previous section that having more than 10000 events in the training set did not improve significantly the results for any of the applied methods. Hence, a change in the algorithms and/or in the way the data are input, or additional information, is required to improve the prediction accuracy. In this section, we include the time of occurrence information in the input of the LSTM network with SPEED-text to investigate whether this leads to an improvement of the prediction accuracy.

1) COMPARISON OF METHODS

We predict the next sensor event based on the three proposed input sequences with time information (Section V-C). Fig. 12 shows the performance of the prediction according to the amount of data in the training set in apartment 1. We include the accuracy when using only the previous sensors for comparison purposes. We achieved an accuracy of 83.26% when predicting the next sensor event using previous sensor events as input (i.e. no time information). When we include the period of the day, the class time intervals, and the K-means time-cluster in the input, the models achieve 84.07%, 85.05%, and 84.01%, respectively. The small improvement of 0.8-2% was initially somewhat surprising, as we had expected that the time information would increase accuracy significantly. However, on second thoughts, the apartments are quite small – limiting the number of possible patterns – and there is a limited number of sensors, and hence a lot of information (including for example time information about movements and actions in the home) is not "visible" for the model. The standard deviation of the LSTM models is about 0.02-0.06%, hence the model is quite stable. A significant improvement, however, is in the convergence of the model that occurs with training set sizes of 2500 events, almost half of the events needed for when no time information is included.

2) PREDICTION VARIABILITY ACROSS APARTMENTS

The input using the 4-class time intervals has shown marginally better results than the other two methods, and we therefore apply this method on the other apartments. The results are shown in Table 8. Including the time has led to improved accuracy in all the apartments, in a range of 0.5-4%.

The 10% variability between apartments for the prediction of the next sensor could be due to the amount of data available in each apartment. In Table 9, we present the average number of events per day and the average time spent out of the apartment per day. This indicates that the degree of activity varies significantly and/or that some of the residents are more active when in the apartment (e.g. apartments 1 and 5) than others. Nonetheless, the average number of events per day does not seem to have a direct influence on the achieved prediction accuracy. For instance, relatively high prediction accuracy (86%) has been achieved for apartment 3 that only has 227 events per day, whereas much lower prediction accuracy (81%) is achieved in apartment 5 that has the highest number of events per day (729). Also, comparable accuracy is attained in apartment 1 (87%) as to apartment 3, although there are on average more than twice as many events per day in the former than in the latter. Hence, there is no correlation between attained accuracy and the average number of events per day here. Another hypothesis for the prediction accuracy variability is the noise originated by different sources in the data for the apartments. For example, the resident in apartment 6 has often family members visiting. This noise cannot be measured in our setup at this moment.

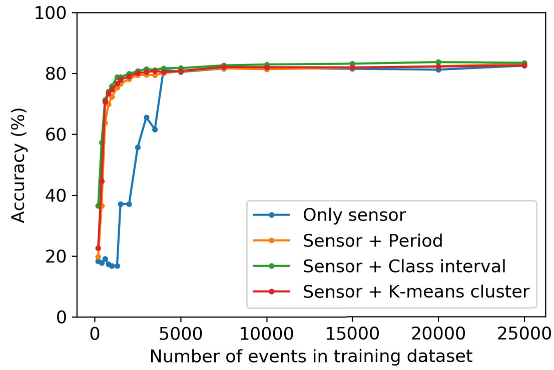

FIGURE 12. Accuracy of prediction of next sensor event vs. the number of events in the dataset.

TABLE 8. Prediction accuracy of the next sensor event based on past sensor events and 4-class time intervals with LSTM with SPEED-text.

Apt. ID	Top Mean Accuracy (Number of Events for Stability)
1	86.58% (4000)
2	83.76% (5000)
3	86.38% (5000)
4	79.29% (10000)
5	80.91% (10000)
6	76.63% (5000)
7	82.91% (5000)
8	86.39% (5000)

TABLE 9. Number of events per day and time spent outside the apartment for each resident.

Apt. ID	Avg Number of Events per Day	Avg Time Out of Apt (h)
1	614	1.5
2	442	1.0
3	227	0.6
4	383	3.6
5	729	1.1
6	206	1.75
7	375	1.85
8	293	7.3

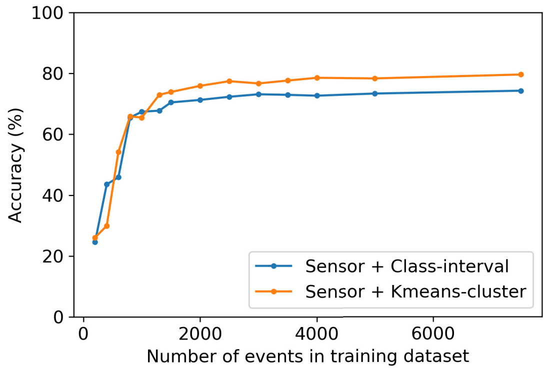

FIGURE 13. Accuracy of prediction of next sensor event and time information vs. the number of events in the training dataset.

Furthermore, the coefficient of variation (standard deviation divided by the mean) in this case is about 0.04, which is lower than 1 and therefore, a low variance. The different predictions in this case may simply indicate some people are more predictable in their patterns around the apartment than others.

D. PREDICTION OF THE NEXT SENSOR EVENT AND ITS MEAN TIME OF OCCURRENCE

1) COMPARISON OF METHODS

In the following we examine the accuracy of predicting both the next sensor event and the time of occurrence information. In this case, only the input sequences of class time intervals and K-means time-cluster are considered. Lower accuracy is attained (Fig. 13) than when predicting only the next sensor event, as expected, since now more information is being predicted within the same model. The best accuracy is achieved by the K-means time-cluster (82.00%), 6% better than the class time-intervals (76.63%). For both methods, convergence is achieved with about 2000 events in the training set (Fig. 13). Our hypothesis is that the K-means algorithm clusters the samples in a more balanced way than the 4-class intervals, and this leads to a better prediction accuracy.

2) PREDICTION VARIABILITY ACROSS APARTMENTS

The K-means time-cluster method attained the highest accuracy when predicting both the next sensor event and time information, and therefore we apply this on the dataset from all apartments. The obtained results are shown in Table 10. The attained accuracy is 3-6% lower than when predicting the next sensor only (Table 8), as expected.

TABLE 10. Prediction accuracy of the next sensor event and time-cluster based on past sensor events and time-cluster with LSTM with SPEED-text.

Apt. ID	Top Mean Accuracy (Number of Events for Convergence)
1	82.97% (5000)
2	79.36% (10000)
3	80.59% (10000)
4	74.87% (10000)
5	77.67% (10000)
6	72.73% (10000)
7	78.09% (5000)
8	79.21% (5000)

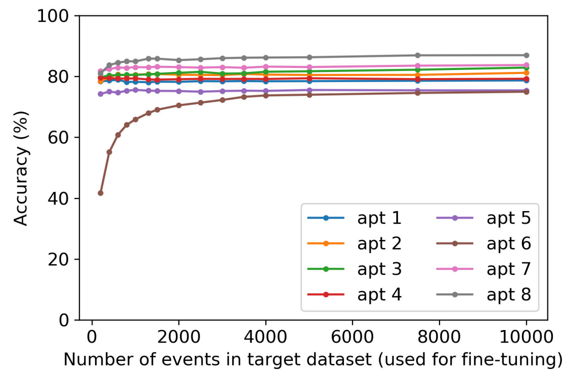

FIGURE 14. Accuracy of prediction of the next sensor vs. number of events used for fine-tuning, using as input both sensor event and 4-class time interval. Transfer learning – training the model with data from seven apartments, fine-tuning with and testing on the target apartment.

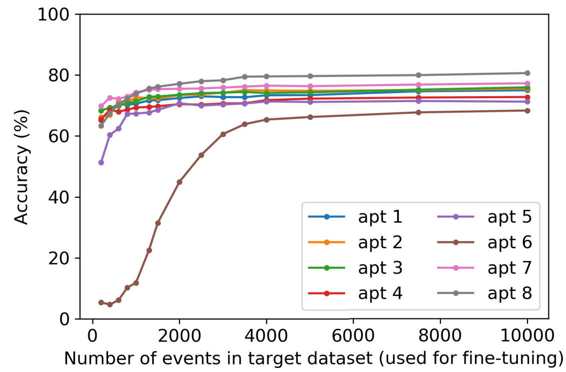

FIGURE 15. Accuracy of prediction of the next sensor and time-cluster vs. number of events used for fine-tuning, using as input both sensor event and time-cluster. Transfer learning – training the model with data from seven apartments, fine-tuning with and testing on the target apartment.

VII. TRANSFER LEARNING ACROSS APARTMENTS

In this section, we investigate whether the transfer learning technique is feasible and beneficial across the apartments in our field trial. We use transfer learning as follows. We first train an LSTM network with data from seven source apartments and fine-tune and test with one target apartment. In this case, the data from the target apartment – that have not been used in the training – are split to be used in the fine-tuning of the network (keeping the weights of the best-fit model), and in the testing (3000 events). We compute the accuracy of predicting the next sensor event only based on input about previous sensor events, as well as time information using 4-class time of occurrence intervals (Fig. 14). In Fig. 15, we present the obtained accuracy when predicting both the next sensor event and the time-cluster based on inputs about both these.

Accuracies from about 80% can be achieved straight away with very little data from the target apartment. There is one exception, apartment 6, which takes much longer time to

TABLE 11. Prediction accuracy with transfer learning without and with Fine-tuning (FT).

Apt. ID	Top Mean Prediction Accuracy			
	Next Sensor		Next Sensor and Time	
	No FT	FT	No FT	FT
1	75.17%	82.52%	60.40%	78.96%
2	77.46%	84.38%	58.17%	79.37%
3	68.67%	85.79%	52.27%	79.14%
4	79.03%	80.44%	60.20%	75.29%
5	71.87%	79.16%	49.87%	75.39%
6	2.93%	75.44%	0.00%	69.16%
7	80.57%	84.11%	59.00%	78.11%
8	78.90%	87.72%	53.43%	81.68%

achieve good accuracy. However, also this apartment required less data (about 4000 events) compared to the case without transfer learning (about 5000 events). For larger training datasets, the prediction accuracy is approximately the same as when each apartment is modelled with its own data. In fact, in most cases is it marginally higher when each apartment is modelled individually, except for apartments 4 and 8.

Note also that when predicting the time-cluster in addition to the next sensor, a larger amount of data is required to transfer the learning as effectively as when not predicting the time. This is due to the fact that when predicting only the next sensor, the layout of the apartment is what is mostly taken into account given that the apartments are very small and all have the same layout (section III). When predicting the time-cluster, we account in addition for the individual habits of each resident, and hence, additional data are required to fine-tune the network.

Table 11 presents the top accuracy obtained for each apartment. We have also computed the accuracy without fine-tuning prior to testing when applying transfer learning, i.e. when the network has been trained with data from seven apartments and subsequently tested directly on the target dataset. For the prediction of the next sensor only, the accuracy is 4-8% lower than when using fine-tuning. When predicting both the next sensor and the time information, the accuracy is 15-30% lower without as compared to with fine-tuning. This is in accordance with what has been mentioned earlier in this section, i.e. that predicting the time takes into account individual patterns, and therefore needs additional data to fine-tune the network to each resident. In either prediction case, the fine-tuning of the model is indeed required to achieve good prediction accuracy when using transfer learning across apartments.

Subsequently, we investigate how much data are required to transfer learning from the base model (with data from several apartments) such that the target apartment will obtain a good accuracy with very little data. We chose apartment 8 to be the target apartment in this case, since it has shown to have higher accuracy with transfer learning rather than when being modeled with its own data. We use 100 events from the target apartment to fine-tune the network and test on 3000 random events. When predicting only the next sensor, about 40000 events from seven different apartments are required so that in apartment 8 80% prediction accuracy can be achieved with only 100 events, as shown in Fig. 14. For predicting the next sensor and the time-cluster much more data are needed, about 500000 from the seven different apartments. In this case, the accuracy with only 100 events in the target apartment is about 60%. As discussed earlier and presented in Fig. 15, when predicting the time, more data are required from the target apartment to achieve the peak accuracy.

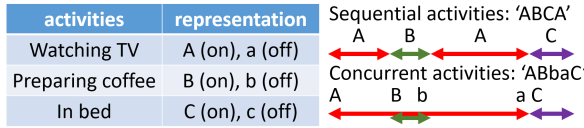

FIGURE 16. Types of activities sequences implemented – sequential and concurrent. The example corresponds to a scenario where the resident watches TV, goes to the kitchen to prepare a coffee while watching TV, and then goes to bed.

VIII. ACTIVITY PREDICTION

A. METHOD

Ultimately, the binary sensor events indicate activities of daily living. In this section, we associate the binary sensor events with activities and predict these. We are only able to register high-level activities as the number of sensors in our set-up is quite limited. Our dataset comprises the following classes: watching TV, being in bed, being out, bedroom activities, living room activities, kitchen activities, bathroom activities, transitions in bedroom/bathroom/entrance/living room – 11 in total.

We implemented two rule-based algorithms for deriving activities from binary sensors that we refer to as *sequential* activities and *concurrent* activities. We decided for a set of rules as described in Table 12. In the case of *sequential* activities, we assume that no more than one activity takes place at the same time, so that as soon as one activity ends, another starts. The time information is the elapsed time to the next activity, which in this case is the duration of the activity. In the case of *concurrent* activities, each activity has a start and an end – indicated by a "1" and a "0", respectively –, allowing several activities to be happening in parallel. For example, the resident can be in the kitchen preparing coffee and still be watching TV. This implies that, in many cases, the duration of the activities will be longer compared to the sequential activities. The time information is inserted such that activity start contains the duration of the activity (time elapsed until the end of the activity) and activity end contains the elapsed time to the start of the next activity event. Fig. 16 shows an example of the two sequences without including the time, for simplicity.

As we do for the sensor events, each activity is assigned a letter, and the time information with K-means time-cluster is selected due to its best performance. The transition classes are only used in the input of the LSTM network, thus the output classes are in fact only 7. The LSTM network

TABLE 12. Rules for deriving activities from sensor events.

Activities	Rules
Kitchen activities	Whenever power and magnetic sensors located in the kitchen are activated or motion sensor in the kitchen is active for more than 1 minute.
Living room activities	Whenever power and magnetic sensors located in the living room (except TV) are activated or motion sensor in the living room is active for more than 5 minutes.
Watching TV	Whenever the resident is in the living room for more than 5 minutes and the power in the TV is on.
Bedroom activities	Whenever power and magnetic sensors located in the bedroom (except sensors around the bed) are activated or motion sensor in the bedroom is active for more than 5 minutes.
Being in bed	Whenever motion sensors around the bed are consecutively activated for more than 5 minutes.
Bathroom activities	Whenever the motion sensor located in the bathroom is active for more than 1 minute.
Being out	Whenever the entrance door "off" and "on" events happen consecutively and for more than 5 minutes; or when the entrance door is the last active motion sensor for more than 10 minutes (in one of the apartments where the entrance door was not installed).
Transitions	Being in the entrance is always considered as a transition as there are no relevant activities in that area. Other rooms have a subjective transition time chosen based on the distance between rooms and conditions of the residents (e.g. walking speed, use of rollator, etc.).

TABLE 13. Number of activities per apartment.

Apt. ID	Number of Activities	
	Sequential	Concurrent
1	70931	66136
2	26460	52920
3	19344	38700
4	53984	61222
5	49665	99330
6	10577	11382
7	10031	20090
8	5446	8162

has the same configuration parameters as the one used for the prediction of sensor events. In addition, we use the Synthetic Minority Oversampling Technique (SMOTE) since our data is imbalanced. SMOTE is an over-sampling technique that creates synthetic samples for the minority classes [49]. The library Imbalanced-Learn was used for this implementation [50].

B. RESULTS AND DISCUSSION

Table 13 shows the number of activity events in each dataset for each apartment. The LSTM network was trained based on a certain number of events and tested on either 3000 random events or 10% of the total number of events (for the apartments with very few activity events, e.g. 6-8). This process is repeated three times, and the accuracy values in the graphs correspond to the mean of the best test accuracy of each training.

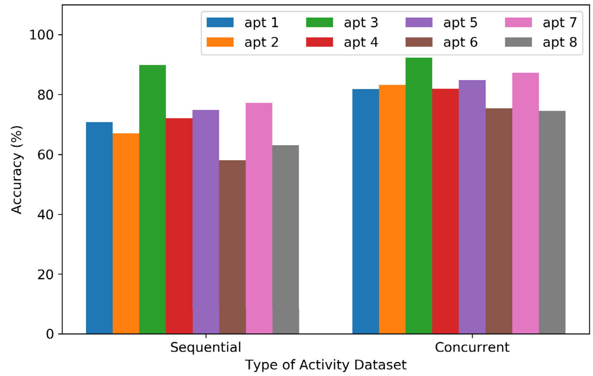

FIGURE 17. Prediction accuracy of next activity based on previous activities per apartment and type of activity dataset.

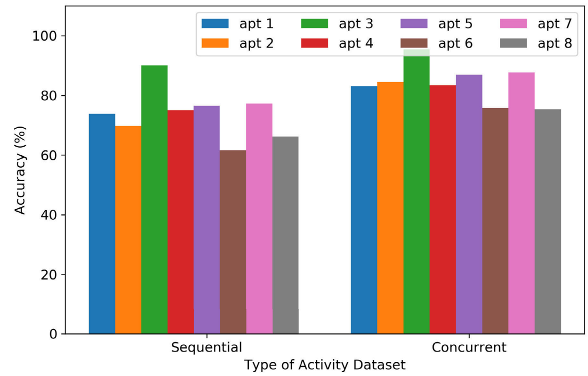

FIGURE 18. Prediction accuracy of the next activity based on previous activities and K-means time-cluster per apartment and type of activity dataset.

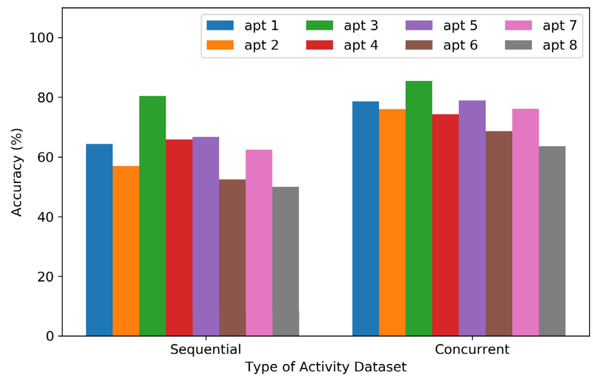

FIGURE 19. Prediction accuracy of both next activity and K-means time-cluster per apartment and type of activity dataset.

Firstly, we predict the next activity based on previous activities (Fig. 17), and subsequently when including the K-means time-cluster (Fig. 18), for both types of activity sequences. Table 14 presents the prediction accuracy for these. For the sequential activities dataset, the prediction accuracy varies between 58-90% without using the time information in the input, and between 61-90% when including the time information. Including the time in the input resulted in 0 (apartment 7) to 3% improvement. In the case of the

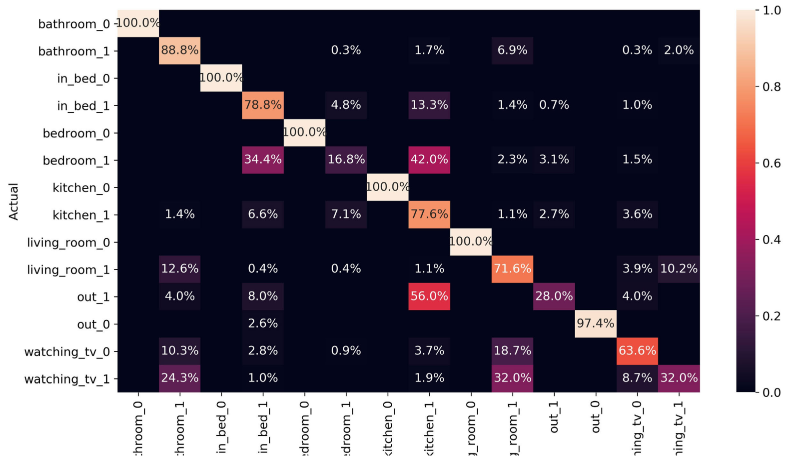

FIGURE 20. Confusion matrix of prediction of the next activity based on previous activities and K-means time-cluster for apartment 1, using the concurrent activity dataset.

TABLE 14. Prediction of the next activity.

Apt. ID	Without time		With time	
	Sequential	Concurrent	Sequential	Concurrent
1	70.80%	81.82%	73.78%	83.04%
2	66.94%	83.18%	69.72%	84.51%
3	89.79%	92.28%	90.02%	95.38%
4	72.13%	81.89%	75.02%	83.40%
5	74.77%	84.81%	76.51%	86.94%
6	58.07%	75.34%	61.67%	75.67%
7	77.23%	87.18%	77.23%	87.72%
8	63.00%	74.56%	66.20%	75.34%

TABLE 15. Prediction of the next activity and K-means Time-Cluster.

Apt. ID	Sequential	Concurrent
1	64.34%	78.54%
2	56.98%	75.93%
3	80.33%	85.44%
4	65.91%	74.28%
5	66.75%	78.98%
6	52.50%	68.59%
7	62.43%	76.12%
8	50.00%	63.58%

concurrent activities dataset, the prediction accuracy varies between 75-92% without using the time information in the input, and between 75-95% when including the time information. Thus the accuracy has improved from 0.3-3.1% across the apartments when the time information is included in the input. Fig. 19 and Table 15 present the accuracy results when predicting both the next activity and its duration/time elapsed to the next activity. The obtained accuracy varies between 64-85% for the concurrent activities, i.e. it is 4.5-11.8% lower compared to above. Similarly, for the sequential activities, an accuracy of 50-80% is achieved, i.e. 9.4-16.2% lower than above. This is expected since now the model has many more classes to predict from and it is in addition predicting more information.

We can notice that apartment 3 has achieved the best accuracy in all tests. One observation is that this resident did not have the power sensor in the TV, so that this model has one less class to predict (*watching tv*). In addition, it is a class that usually presents much confusion with others, especially with *living room activity*. Apartments 6 and 8 have shown similar and poor accuracies in the tests, however, they do not have enough data for conclusive results (see number of activity events in Table 13). The other apartments – 1, 2, 4, 5, and 7 – present comparable results.

The accuracy results for the concurrent activities dataset were better in all cases – 5.4-14% improvement when predicting only the next activity based on previous activities and time; and 5.1-16.1% when predicting the next activity and K-means time-cluster. However, since there is only one resident and a relatively small number of sensors in each apartment, that moreover do not relate to other sensors, there are in reality only few concurrent activities. Hence, most of the "start" activity events in the concurrent dataset are immediately followed by the "end" of the same activity. Therefore, most of the "end" of activities is predicted with 100% accuracy, which explains the higher accuracies of this method. This can be confirmed by the confusion matrix obtained with the prediction accuracy results in apartment 1 – Fig. 20. Nevertheless, this may be a good implementation in smart home environments where several activities can happen at the same time, e.g. multi-resident smart homes. This is not the case of our setup, hence the sequential activity dataset is probably a fairer algorithm. An example confusion matrix for this dataset (apartment 1), is shown in Fig. 21. The confusions within classes are similar for both types of datasets. *Bedroom activities* are mostly predicted as *in bed* and *kitchen activities*. This is understandable since bedroom activities happen often after having been in bed or in the living room, which has access to the kitchen. As mentioned before, *living room activities* are confused with *watching TV*, and a little with *kitchen activities*, as the previous comment. And finally, *being out* has been predicted most of the times as *kitchen activities*, as the entrance door also has a connection to the living room. An interesting result is that in this

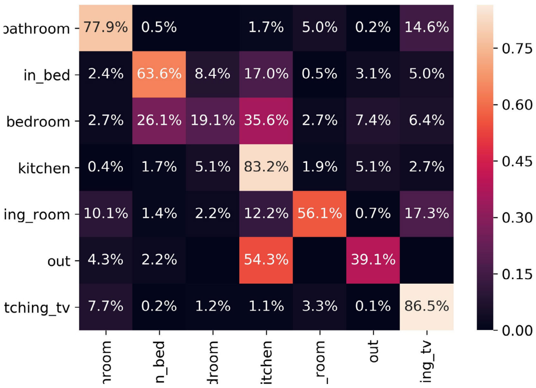

FIGURE 21. Confusion matrix of prediction of the next activity based on previous activities and K-means time-cluster for apartment 1, using the sequential activity dataset.

apartment the *watching tv* activity has been very well predicted – 86.5%. This could be useful for smart functions involving the TV, e.g. if the resident has difficulties operating the remote control. *Bathroom* and *kitchen activities* have also shown a considerably good accuracy (77.9% and 83.2%). The range of accuracy may be useful for analyzing patterns in the home and potentially for anomaly detection.

IX. CONCLUSION

Sequential sensor events, time prediction, and activity recognition and prediction algorithms can enable the development of a number of support functions in smart home environments. Most of the research work in the literature has been carried out using data collected in lab environments and testbeds, typically including a quite large number of binary sensors (e.g. 50 sensors [18]). We collected data from eight apartments in a community care facility, with one resident each (over 70 years old). Data were collected from 13-17 sensors per apartment, over a period of time ranging from 75 to 385 days, depending on the apartment.

To our knowledge, there is no comparative study investigating state-of-the-art sequence prediction algorithms applied to sensor data acquired in homes of real users, as we do in this paper. We compare the performance of these methods regarding factors such as memory length and the required amount of data for good accuracy. When applying two probabilistic methods (ALZ and SPEED) and LSTM networks with both SPEED- and ALZ-text sequence inputs for prediction of the next sensor in a sequence, LSTM with SPEED-text has achieved the highest accuracy of 85%. SPEED achieved 3% lower accuracy and required much longer time to execute. On the other hand, the LSTM required about 4000 events in the training set to reach an accuracy close to its peak, whilst the probabilistic methods only needed about 2000 events. Hence, for datasets with little data SPEED may be beneficial. If there is a considerable amount of data (5000 events in this work), LSTM with SPEED-text is more suitable – it provides a higher accuracy and in much faster execution time than probabilistic methods. When tested in all the apartments, LSTM with SPEED-text achieves results in the range 76-85%.

There is quite limited work in the literature on the prediction of the time of occurrence in addition to the sensor events in smart homes. We study the possibility of improving the best performing algorithm (LSTM with SPEED-text) by including the time component in three different ways: period of the day (morning, afternoon, evening, night), 4-class time interval (elapsed time) to the next sensor event, and K-means time-cluster including information about the mean hour of the day and the mean time elapsed to the next sensor event. Our best performing model for predicting the next sensor event included the 4-class time interval input and attained a peak average accuracy of almost 87%. This is 2% better than without including the time information. Hence, the time elapsed between events contains some information that improves prediction, however, only marginally. In other apartments the improvement varied from 0.5-4.5%. We also predict both the next sensor event and the time of occurrence information, obtaining best results by using K-means time-cluster input. This implementation attained an accuracy of 83%. Other apartments had accuracies in the range 73-83%. Furthermore, we evaluate the variability of the prediction accuracy across the apartments and investigate the feasibility of transfer learning between these. Transfer learning has been shown to work successfully up to a certain number of events. For a low number of events in the training dataset, up to about 4000 events, transfer learning leads to higher prediction accuracy than when each apartment is modelled individually. This means that when a new apartment is added to the study, the prediction algorithm can work well straight away, and attain a relatively good accuracy (70-80%) from the first day in most cases. However, for larger training datasets, the prediction accuracy is approximately the same. In fact, in most cases it is marginally higher when each apartment is modelled individually.

A last analysis carried out activity recognition in a rule-based manner from the binary sensors events and performed activity prediction with the LSTM with SPEED-text algorithm. Two types of activity datasets were analyzed: sequential and concurrent. For the concurrent activity dataset, when predicting the next activity only, our best model achieved 95% accuracy, whilst when predicting the next activity and the mean duration and time of occurrence information, the best model achieved an accuracy of 85%. For the sequential activity dataset, the results are worse. When predicting the next activity, our best model achieved 90% accuracy, whilst when predicting the next activity and its duration and time of occurrence information, the best model achieved 80%. However, we indicate that this latter method may be fairer for our dataset where there are relatively few activities happening concurrently. Additional sensors could have been an advantage for better activity recognition and prediction. Our set of sensors proved to be somewhat limited for the task since it can only imply high-level activities. A small number of sensors like ours may, however, be preferable both in terms of reduced surveillance for the user, lower cost, and less nuance for the aesthetics of the home. Our work shows that it is possible to achieve acceptable prediction accuracy with few sensors. In addition, the findings of our study can be useful for deciding which analysis and prediction methods to use in accordance with project constraints (e.g. the number of available sensors, user privacy, etc.) and the area of application.

ACKNOWLEDGMENT

The authors would like to thank the residents and the housekeepers at the seniors' care unit Skøyen Omsorg+ for their invaluable contribution to this work; Torhild Holthe and Dr. Anne Lund (OsloMet) for recruiting participants for the trial and communicating with the residents throughout the trial; Dejan Krunić and Øyvind Width (Sensio AS) for installations of the sensors; and the rest of the participants of the Assisted Living Project for a fruitful interdisciplinary collaboration.

REFERENCES

[1] B. Winblad *et al.*, "Mild cognitive impairment–beyond controversies, towards a consensus: Report of the international working group on mild cognitive impairment." *J. Internal Med.*, vol. 256, no. 3, pp. 240–246, 2004.

[2] S. Blackman, C. Matlo, C. Bobrovitskiy, A. Waldoch, M. L. Fang, P. Jackson, A. Mihailidis, L. Nygård, A. Astell, and A. Sixsmith, "Ambient assisted living technologies for aging well: A scoping review," *J. Intell. Syst.*, vol. 25, no. 1, pp. 55–69, Mar. 2015.

[3] A. Mihailidis, J. Boger, M. Canido, and J. Hoey, "The use of an intelligent prompting system for people with dementia," *Interactions*, vol. 14, no. 4, pp. 34–37, 2007.

[4] J. Hoey, A. V. Bertoldi, P. Poupart, and A. Mihailidis, "Assisting persons with dementia during handwashing using a partially observable Markov decision process." in *Proc. ICVS*, 2007, pp. 1–10.

[5] A. Ashraf and B. Taati, "Automated video analysis of handwashing behavior as a potential marker of cognitive health in older adults," *IEEE J. Biomed. Health Inform.*, vol. 20, no. 2, pp. 682–690, Mar. 2016.

[6] S. Karaman, J. Benois-Pineau, R. Mégret, J. Pinquier, Y. Gaëstel, and J. F. Dartigues, "Activities of daily living indexing by hierarchical HMM for dementia diagnostics," in *Proc. 9th Int. Workshop Content-Based Multimedia Indexing*, 2011, pp. 79–84.

[7] P. J. McCullagh, W. Carswell, M. D. Mulvenna, J. C. Augusto, H. Zheng, and W. P. Jeffers, "Nocturnal sensing and intervention for assisted living of people with dementia," in *Healthcare Sensor Networks: Challenges Toward Practical Implementation*. Boca Raton, FL, USA: CRC Press, 2012, pp. 283–303.

[8] M. A. Feki, J. Biswas, and A. Tolstikov, "Model and algorithmic framework for detection and correction of cognitive errors," *Technol. Health Care*, vol. 17, no. 3, pp. 203–219, 2009.

[9] E. Zouganeli, F. D. Casagrande, T. Holthe, A. Lund, L. Halvorsrud, D. Karterud, A. Flakke-Johannessen, H. Lovett, S. K. Mørk, J. Strøm-Gundersen, E. Thorstensen, R. Norvoll, R. ter Meulen, M.-R. Kennedy, R. J. Owen, M. Ladikas, and E.-M. Forsberg, "Responsible development of self-learning assisted living technology for older adults with mild cognitive impairment or dementia," in *Proc. 3rd Int. Conf. Inf. Commun. Technol. Ageing Well E-Health (ICT4AWE)*, 2017, pp. 204–209.

[10] F. D. Casagrande, J. Tørresen, and E. Zouganeli, "Comparison of probabilistic models and neural networks on prediction of home sensor events," in *Proc. Int. Joint Conf. Neural Netw.*, 2019.

[11] F. D. Casagrande, J. Tørresen, and E. Zouganeli, "Sensor event prediction using recurrent neural network in smart homes for older adults," in *Proc. Int. Conf. Intell. Syst.*, 2018, pp. 662–668.

[12] F. D. Casagrande and E. Zouganeli, "Occupancy and daily activity event modelling in smart homes for older adults with mild cognitive impairment or dementia," in *Proc. 59th Conf. Simulation Modelling (SIMS)*, 2018, pp. 236–242.

[13] F. D. Casagrande, J. Tørresen, and E. Zouganeli, "Prediction of the next sensor event and its time of occurrence in smart homes," in *Proc. 28th Int. Conf. Artif. Neural Netw. (ICANN)*, 2019.

[14] F. D. Casagrande and E. Zouganeli, "Prediction of next sensor event and its time of occurrence using transfer learning across homes," in *Proc. ACS/IEEE Int. Conf. Comput. Syst. Appl. (AICCSA)*, 2019.

[15] L. B. Holder and D. J. Cook, "Automated activity-aware prompting for activity initiation," *Gerontechnology*, vol. 11, no. 4, pp. 534–544, 2013.

[16] 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," *Artif. Intell. Med.*, vol. 67, pp. 57–74, Feb. 2016.

[17] S. Wu, J. B. Rendall, M. J. Smith, S. Zhu, J. Xu, H. Wang, Q. Yang, and P. Qin, "Survey on prediction algorithms in smart homes," *IEEE Internet Things J.*, vol. 4, no. 3, pp. 636–644, Jun. 2017.

[18] K. Gopalratnam and D. J. Cook, "Online sequential prediction via incremental parsing: The active LeZi algorithm," *IEEE Intell. Syst.*, vol. 22, no. 1, pp. 52–58, Jan./Feb. 2007.

[19] M. R. Alam, M. B. I. Reaz, and M. A. M. Ali, "SPEED: An inhabitant activity prediction algorithm for smart homes," *IEEE Trans. Syst., Man, Cybern.-A, Syst. Hum.*, vol. 42, no. 4, pp. 985–990, Jul. 2012.

[20] A. Lotfi, C. Langensiepen, M. M. Sawsan, and M. J. Akhlaghinia, "Smart homes for the elderly dementia sufferers: Identification and prediction of abnormal behaviour," *J. Ambient Intell. Hum. Comput.*, vol. 3, no. 3, pp. 205–218, 2012.

[21] L. Vintan, A. Gellert, J. Petzold, and T. Ungerer, "Person movement prediction using neural networks," in *Proc. KI2004 Int. Workshop Modeling Retr. Context*, 2004, vol. 114, no. 4, pp. 1–12.

[22] J. Petzold, F. Bagci, W. Trumler, and T. Ungerer, "Next location prediction within a smart office building," *Cogn. Sci. Res. Paper-Univ. Sussex (CSRP)*, vol. 577, p. 69, May 2005.

[23] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco, CA, USA: Holden-Day, 1990.

[24] B. Minor and D. J. Cook, "Forecasting occurrences of activities," *Pervasive Mobile Comput.*, vol. 38, pp. 77–91, Jul. 2016.

[25] A. Aztiria, J. C. Augusto, A. Izaguirre, and D. Cook, "Learning accurate temporal relations from user actions in intelligent environments," in *Proc. 3rd Symp. Ubiquitous Comput. Ambient Intell.*, in Advances in Soft Computing, vol. 51, J. M. Corchado, D. I. Tapia, and J. Bravo, Eds. Berlin, Germany: Springer, 2009.

[26] S. Mahmoud, A. Lotfi, and C. Langensiepen, "Behavioural pattern identification and prediction in intelligent environments," *Appl. Soft Comput.*, vol. 13, no. 4, pp. 1813–1822, Apr. 2013.

[27] E. Nazerfard and D. J. Cook, "CRAFFT: An activity prediction model based on Bayesian networks," *J. Ambient Intell. Humaniz. Comput.*, vol. 6, pp. 193–205, 2015. doi: 10.1007/s12652-014-0219-x.

[28] T. Mahmud, M. Hasan, A. Chakraborty, and A. K. Roy-Chowdhury, "A poisson process model for activity forecasting," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 3339–3343.

[29] S. J. Pan and Q. Fellow, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[30] D. Cook, K. D. Feuz, and N. C. Krishnan, "Transfer learning for activity recognition: A survey," *Knowl. Inf. Syst.*, vol. 36, no. 3, pp. 537–556, Sep. 2013.

[31] D. H. Hu and Q. Yang, "Transfer learning for activity recognition via sensor mapping," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2011, pp. 1962–1967.

[32] P. Rashidi and D. J. Cook, "Activity knowledge transfer in smart environments," *Pervasive Mobile Comput.*, vol. 7, no. 3, pp. 331–343, 2011.

[33] T. Desautels *et al.*, "Using transfer learning for improved mortality prediction in a data-scarce hospital setting," *Biomed. Inform. Insights*, vol. 9, 2017, Art. no. 1178222617712994. doi: 10.1177/1178222617712994.

[34] D. H. Hu, V. W. Zheng, and Q. Yang, "Cross-domain activity recognition via transfer learning," *Pervasive Mobile Comput.*, vol. 7, no. 3, pp. 344–358, 2011.

[35] P. Gupta, P. Malhotra, L. Vig, and G. Shroff, "Transfer learning for clinical time series analysis using recurrent neural networks," pp. 4–7, 2018, *arXiv:1807.01705*. [Online]. Available: https://arxiv.org/abs/1807.01705

[36] P. Jaini, Z. Chen, P. Carbajal, E. Law, L. Middleton, K. Regan, M. Schaekermann, G. Trimponias, J. Tung, and P. Poupart, "Online Bayesian transfer learning for sequential data modeling," in *Proc. 5th Int. Conf. Learn. Represent.*, Paris, France, 2017, pp. 1–20.

[37] T. L. M. van Kasteren, G. Englebienne, and B. J. A. Kröse, "Transferring knowledge of activity recognition across sensor networks," in *Proc. 8th Int. Conf. Pervasive Comput. (Pervasive)*, Helsinki, Finland, 2010, pp. 283–300.

[38] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Transfer learning for time series classification," in *Proc. IEEE Int. Conf. Big Data*, Dec. 2018, pp. 1367–1376.
[39] J. G. Cleary and I. Witten, "Data compression using adaptive coding and partial string matching," *IEEE Trans. Commun.*, vol. C-32, no. 4, pp. 396–402, Apr. 1984.
[40] J. G. Cleary, W. J. Teahan, and I. H. Witten, "Unbounded length contexts for PPM," in *Proc. Conf. Data Compress. (DCC)*, 1995, pp. 52–61. [Online]. Available: https://www.cs.waikato.ac.nz/ml/publications/1995/Cleary-Teahan-Witten-ppm.pdf and http://ieeexplore.ieee.org/document/515495/
[41] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," *IEEE Trans. Inf. Theory*, vol. IT-24, no. 5, pp. 530–536, Sep. 1978. [Online]. Available: http://ieeexplore.ieee.org/document/1055934/
[42] A. Graves, "Generating sequences with recurrent neural networks," Aug. 2014, *arXiv:1308.0850*. [Online]. Available: https://arxiv.org/abs/1308.0850
[43] I. Sutskever, J. Martens, and G. Hinton, "Generating text with recurrent neural networks," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, L. Getoor and T. Scheffer, Eds. Madison, WI, USA: Omnipress, 2011, pp. 1017–1024.
[44] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, vol. 3, May 2013, pp. 6645–6649.
[45] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[46] N. E. Elhady and J. Provost, "A systematic survey on sensor failure detection and fault-tolerance in ambient assisted living," *Sensors*, vol. 18, no. 7, p. 1991, 2018.
[47] K. M. Bataineh, M. Najia, and M. Saqera, "A comparison study between various fuzzy clustering algorithms," *Jordan J. Mech. Ind. Eng.*, vol. 5, no. 4, pp. 335–343. 2011.
[48] K. D. Joshi and P. S. Nalwade, "Modified k-means for better initial cluster centres," *Int. J. Comput. Sci. Mobile Comput.*, vol. 2, no. 7, pp. 219–223, 2013.
[49] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, no. 1, pp. 321–357, 2002.
[50] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *J. Mach. Learn. Res.*, vol. 18, no. 17, pp. 1–5, 2017. [Online]. Available: http://jmlr.org/papers/v18/16-365.html

FLÁVIA D. CASAGRANDE was born in Vitoria, Espirito Santo, Brazil, in 1990. She received the B.S. degree in electrical engineering from the Federal University of Espirito Santo, Brazil, in 2013, and the M.S. degree in computer vision and robotics from the Erasmus Mundus joint programme from the University of Burgundy, France, Heriot-Watt University, Scotland, and the University of Girona, Spain, in 2016. She is currently pursuing the Ph.D. degree in informatics with the University of Oslo, Norway, externally employed at OsloMet—Oslo Metropolitan University, Norway.

Her research interests include machine learning and computer vision, with a focus on medicine, assisted living, and health applications.

JIM TØRRESEN received the M.Sc. and Dr.Ing. (Ph.D.) degrees in computer architecture and design from the Norwegian University of Science and Technology, Trondheim, in 1991 and 1996, respectively.

He was a Senior Hardware Designer with NERA Telecommunications, from 1996 to 1998, and Navia Aviation, from 1998 to 1999. He was a Visiting Researcher with Kyoto University, Japan, from 1993 to 1994, the Electrotechnical Laboratory, Tsukuba, Japan, in 1997 and 2000, and Cornell University, USA, from 2010 to 2011. Since 1999, he has been a Professor with the Department of Informatics, University of Oslo, where he was an Associate Professor, from 1999 to 2005. He is also the Principle Investigator with the RITMO Centre for Interdisciplinary Studies in Rhythm, Time and Motion. His research interests include artificial intelligence, ethical aspects of AI and robotics, machine learning, robotics, and applying this to complex real-world applications, which have resulted in approximately 190 scientific peer-reviewed papers in international journals, books, and conference proceedings. He is a member of the Norwegian Academy of Technological Sciences (NTVA) and the National Committee for Research Ethics in Science and Technology (NENT).

EVI ZOUGANELI received the B.Sc. degree in applied physics from the University of Patras, Greece, in 1985, the M.Sc. degree in telecommunications, in 1988, and the Ph.D. degree in optoelectronics (multiple quantum well devices), in 1992, from the Department of Electrical Engineering, University College London, U.K., and the master's degree in management from the Norwegian School of Management (BI), Oslo, Norway, in 2002.

She held a post doctorate position at the Swiss Federal Institute of Technology (ETH) Zürich, from 1992 to 1994. She joined the Norwegian Telenor Research, incumbent telecom operator, in Oslo, Norway, where she carried out research in a series of European collaborative projects—among others in semiconductors, optical switching, broadband networks, and the Internet of Things. From 2004 to 2010, she was a Virtual Department Leader for the European Network of Excellence in Photonics—ePhotonONE, BONE. In parallel, she carried out consulting work within telecom, technology strategy, and roadmap for Telenor, as well as engaged in business modeling and new business work. From 2010 to 2011, she was with the Norwegian Radium Hospital, where she worked with medical informatics. She joined Oslo Metropolitan University, in April 2011, where she is currently an Associate Professor. She leads the research group in Automation, Robotics, and Intelligent Systems (ARIS). Her current research interests include machine learning and cognitive systems for industrial applications, robotics, and personalized health. She is a board member of the Norwegian Artificial Intelligence Society (NAIS).