Comparison of Probabilistic Models and Neural Networks on Prediction of Home Sensor Events

Flávia Dias Casagrande
Dept. of Mechanical, Electronics, and Chemical Engineering
OsloMet – Oslo Metropolitan University
Oslo, Norway
flacas@oslomet.no

Jim Tørresen
Dept. of Informatics
University of Oslo
Oslo, Norway
jimtoer@ifi.uio.no

Evi Zouganeli
Dept. of Mechanical, Electronics, and Chemical Engineering
OsloMet – Oslo Metropolitan University
Oslo, Norway
evizou@oslomet.no

***Abstract*—We present results and comparative analysis on the prediction of sensor events in a smart home environment with a limited number of binary sensors. We apply two probabilistic methods, namely Sequence Prediction via Enhanced Episode Discovery – SPEED, and Active LeZi – ALZ, as well as Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) in order to predict the next sensor event in a sequence. Our dataset has been collected from a real home with one resident over a period of 30 weeks. The binary sensor events are converted to two different text sequences as dictated by SPEED and ALZ, which are also used as inputs for the LSTM networks. We compare the performance of the algorithms regarding the number of preceding sensor events required to predict the next one, the required amount of data for the model to reach peak accuracy and stability, and the execution time. In addition, we analyze these for two different sets of sensors. Our best implementation achieved a peak accuracy of 83% for a set with fifteen sensors including motion, magnetic and power sensors, and 87% for seven motion sensors.**

***Index Terms*—smart home, sensor data prediction, binary sensors, recurrent neural network, probabilistic models**

I. Introduction

The Assisted Living Project (ALP) is an interdisciplinary project involving health, ethics, and technology experts [1]. The aim is to develop assisted living technology (ALT) to support older adults with Mild Cognitive Impairment (MCI) or Dementia (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 [2]. A fair amount of research on smart home functions has aimed at assisting older adults with MCI/D in their everyday life. Examples are functions such as prompting with reminders or encouragement, diagnosis tools, as well as prediction, anticipation and prevention of hazardous situations. These require quite robust and reliable activity recognition and prediction algorithms in order to be deployed in real homes.

Activity recognition and prediction can be performed by various algorithms that have been reported in the literature. Most of this work has used data collected in the lab based on scripted activities. In addition, there is no comparative study investigating different configurations for input of data, the required data size for accurate predictions, or providing guidelines as to the applicability of these. In this work, we apply state-of-the-art sequence prediction algorithms, both probabilistic methods and recurrent neural networks, to binary sensor data acquired from a real home with a relatively small number of sensors over a period of 30 weeks. We compare the performance of these methods for sensor event prediction with regard to the amount of data, the time used for training and testing the models, and the number of preceding events required to predict the next event (memory length). We further analyze the performance of the algorithms for two different sets of sensors – one with events from fifteen sensors (motion, magnetic and power) and one with events from seven motion sensors only.

Section II gives an overview of algorithms used for sensor sequential prediction in the literature. Section III describes our field trial. Section IV presents the methods used in the current work. In section V we present our results and discuss our findings. The paper concludes in Section VI with a short summary and ideas for improvement and future work.

Financed by the Norwegian Research Council under the SAMANSVAR programme (247620/O70).

II. Related Work

Several sequential data prediction algorithms have been investigated in the past years [3]. These have a broad range of application areas, including sensor event and activity prediction – the basis of several functions in smart homes. Such algorithms can for instance lead to an improved operation of automation functions (e.g. turn on the heater a sufficient time prior to the person arriving at home); enable the realization of prompting systems (e.g. prompt the resident if the predicted activity has not been performed) [4]; 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 [5].

The Active LeZi (ALZ) is a probabilistic method that has been extensively employed for prediction on sequential data [6]. It achieved a peak accuracy of 47% when applied on the Mavlab testbed dataset, that includes 50 binary sensors [6]. Based on the ALZ, the Sequence Prediction via Enhanced Episode Discovery (SPEED) algorithm was implemented [7]. 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. The tree is Markov model-based, where at any given point in time the next state depends solely on the previous one [8]. Hence, the most probable next event can be estimated based on the current state, by using the Prediction by Partial Matching algorithm (PPM) [9].

Neural networks have also been used for sensor event prediction with notable performance, typically recurrent neural networks (RNN). 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 [10]. 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. In a subsequent work, a Non-linear Autoregressive Network (NARX) was compared to an Elman network. Both used as input and output the start and end time of a sensor's activation [11]. In this study, each sensor had its own network trained and tested on a twenty-day dataset with the same six binary sensors. The NARX performed better when predicting only the next step, with a RMSE ranging from 0.06 to 0.09, depending on the sensor.

A similar study was carried out for a 16-room office environment [12]. The dataset in this case was collected through an app the employees had installed on a personal data assistant (PDA). They would register themselves 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 [13].

Other related research includes prediction of the next activity as well as the time, location, and day it would occur using Bayesian networks, which achieved 74% of activity prediction [14]. Prediction of the time when a certain activity will take place has also been investigated using decision trees [15] and time series [16].

Our dataset was collected from a real home, while most datasets from the cited works have been collected through scripted activities primarily in lab environments. In addition, it contains events from fifteen binary sensors, i.e. twice as many as used in [10], [11], and less than one third of the number of sensors used in the Mavlab testbed. The number of sensors is comparable to the work in [12] (16 rooms), however in that study the events were inserted by each user in their PDA rather than being generated automatically.

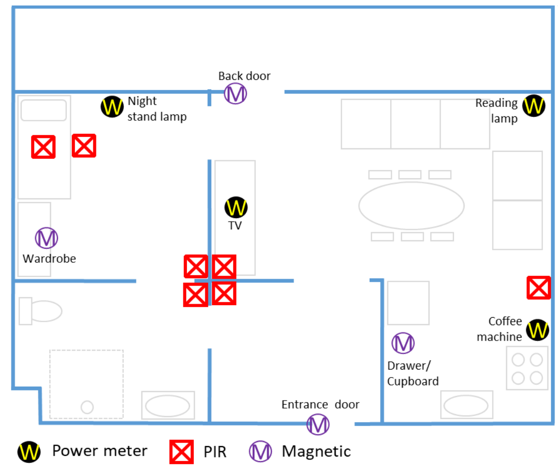

Fig. 1. Sensors system installed in the field trial apartment.

III. Field Trial

Our field trial includes nine apartments in a community care facility with residents over 65 years old. In this work we use data from one of the apartments where we have collected 30 weeks of data. The apartments comprise a bedroom, a living room, an open kitchen area, a bathroom, and an entrance hall (Fig. 1).

The purpose of the trial and the sensors system deployed in the apartments have been decided in close collaboration with the participants [1]. We installed a minimal number of binary sensors in order to both minimize surveillance of the residents and comply with the technical and economic constraints imposed by the project. The set of sensors has subsequently been chosen so that it can enable the realization of useful functions for older adults with MCI/D as these were indicated at dialogue cafés with the users [1]. Hence, our set of sensors contains motion, magnetic, and power sensors. These enable inference of occupancy patterns (movement around the apartment) and some 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 sends a message "1" every time a motion is detected, otherwise it sends no other message. 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 sending 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 send messages "1" and "0" respectively.

Fig. 1 shows the schematic of the apartment we collected 30 weeks of data from, with 15 sensors in total:

- Seven motion sensors: one in each room of the apartment, and two over and by the bed to indicate whether the person is in bed;
- Four magnetic sensors: entrance and back doors, wardrobe, and cutlery drawer;
- Four power sensors on appliances: night stand lamp, coffee machine, TV, and living room/reading lamp.

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 I presents events generated by the following example scenario: the resident wakes up (PIR bedroom "on"), goes to the living room (PIR living room "on"), turns on the TV (power TV "on"), goes to the kitchen (PIR kitchen "on"), starts the coffee machine (power coffee "on"), goes back to the living room (PIR living room "on") while coffee is prepared, goes back to kitchen (PIR kitchen "on") to get the coffee (power coffee "off") and drink it in the living room (PIR living room "on").

IV. Sensor Data Prediction Methods

A. Preprocessing

The preparation of the data includes two steps: data correction and data conversion. The data correction is necessary because the data acquired from binary sensors often contain faulty events e.g. erroneous activation of motion sensors by sunlight, bouncing of contact sensors, or switch-off delays of motion sensors [17]. Such flawed data may substantially affect the performance of the models that will learn erroneous patterns. In our system, we observed that sometimes the motion sensors do not send an activation event, as they should. Missing sensor events have been inserted to correct for this. For example, it is not possible to go to the bedroom directly from the kitchen without passing through the living room. If the living room motion sensor activation event is missing, it is inserted. In the case where 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 distribution of the inserted events corresponds to the percentage distribution of the two options as observed in the data. This process had a significant effect on the obtained accuracy.

TABLE I
Binary sensors data

Timestamp	*Sensor ID*	*Sensor message*
01.09.2017 07:58:05	2	1
01.09.2017 07:58:40	4	1
01.09.2017 07:59:02	10	1
01.09.2017 07:59:50	5	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
01.09.2017 08:03:33	4	1

Subsequently, the corrected data is converted to two sequences of letters, as dictated by the ALZ and SPEED algorithms. The resulting sequences are also fed into LSTM networks that are configured as text generation networks.

The conversion assigns a dedicated letter to each of the sensors. In the case of ALZ, only "on" events are taken into account, and hence only lower-case letters are used. SPEED, on the other hand, differentiates "on" and "off" events of the same sensor by using upper- and lower-case letters, respectively. Table II presents the assigned letters corresponding to the example scenario in a smart home described in Table I.

B. Active LeZi

ALZ [6] is a largely used algorithm for sequence prediction. From the sequence of lower-case letters, ALZ derives several patterns and their frequency of occurrence. This is based on the LZ78 text compression algorithm [18]. Given a certain sequence $x_1, x_2, \ldots, x_i$, the LZ78 will parse it 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, ALZ would generate the sequence "abcdebdb" for the scenario in Table I. The derived patterns according to LZ78 would be "a", "b", "c", "d", "e", "bd". ALZ generates these and even more patterns from the original ones, if possible. For example, "bd" also generates the pattern "d". This addition accounts for patterns that were not perceived by the LZ78 algorithm and that are still possible in a smart home environment. This modification increases the convergence rate of the model [6]. Besides the patterns, their frequency of occurrence is also counted. An order-k-1 Markov tree is then constructed based on the patterns and their frequencies. Note that k corresponds to the longest pattern found in a training sequence. Fig. 2 shows the generated tree for the example scenario with sequence "abcdebdb".

Subsequently, the PPM algorithm is used for predicting the next event. The PPM algorithm calculates the probability distribution of each possible event based on a given sequence

TABLE II
Assignment of letters to sensors

Sensor (ID)	*Letter*
PIR bedroom (2)	a/A
PIR living room (4)	b/B
Power TV (10)	c/C
PIR kitchen (5)	d/D
Power coffee machine (12)	e/E

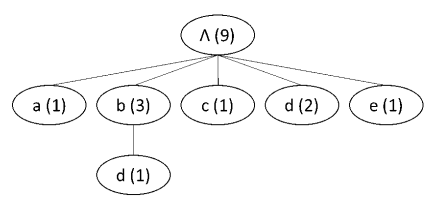

Fig. 2. Tree generated by the ALZ algorithm for the sequence "abcdebdb".

by taking into consideration the different order Markov models in the formed tree with different weights [9].

C. Sequence Prediction via Enhanced Episode Discovery

SPEED is, like ALZ, a sequence prediction algorithm based on the occurrence of frequent patterns in home environments [7]. SPEED defines an *episode* as the sequence between an "on" and an "off" event of the same sensor, or vice-versa. For example, the events that occur between the TV turned "on" and "off", these included, is an *episode*.

Upper- and lower-case letters represent a sensor's "on" and "off" events. For the example scenario presented in Table I, SPEED would generate the sequence "AaBCbDEdBbDedB".

SPEED extracts episodes from a given sequence and derive patterns from them. In the previous sequence, the first episode that is found is "Aa" and the patterns derived from it would be "A", "a" and "Aa". These are used to generate a decision tree that keeps track of the learned episodes and their frequencies, as performed by ALZ. A tree for the example sequence is presented in Fig. 3. Note that the height of the tree is the length of the longest episode found in the sequence. The PPM algorithm is also used for the prediction of the next event.

D. Long Short-Term Memory Network

RNN has been broadly applied to sequence prediction due to its property of keeping an internal memory. Hence, it attains a good performance for inputs that are sequential in time. Examples of applications include text generation [19], speech recognition [20] and pattern recognition in music [21]. The LSTM [22] is an RNN architecture designed to be better at storing and accessing information than the standard RNN [23].

In this work the LSTM network is 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. 4). The input and output are one-hot encoded. In the one-hot encoding representation, each letter is represented by a vector of bits of length equal to the number of letters. All values are zero, except for the one corresponding to that letter (see Fig. 4).

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. Memory length (i.e. number of events that are used to predict the next event) was set to 9. The model has one hidden layer with 64 neurons. The number of samples used for training each iteration of the epoch (i.e. batch size) was 512 and learning rate of 0.01. Adam was used as the optimization function, categorical cross-entropy as loss function, and the activation functions in the hidden layer and output layer were set as hyperbolic tangent and softmax, respectively. We used the early stopping method to avoid overfitting and unnecessary computations, allowing a maximum of 200 epochs for each model's training.

V. Results and Discussion

Data have been collected from one apartment over a period of 30 weeks. Table III shows the number of sensor events for ALZ- and SPEED-text sequences, after data correction and conversion, for the two sets of sensors we analyze (all 15 sensors and only the 7 PIR sensors).

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 [7]. In [7], the authors use the same dataset for both training and testing, which leads to overfitting.

We have modified the testing procedure 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 episode length have been considered. In a previous paper [24], we applied the SPEED method on our data that were obtained from the same home as reported here over a period of two weeks. When using the same procedure as in [7], 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.

Similarly for ALZ we obtained 73% (compared to 47% in [6]) when using the same dataset for training and testing, and 53% when using different 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.

We use the same split rates for the sets used in the LSTM network, where 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. We can notice from Table III that the majority of the events are from motion sensors. Therefore, during the training process in the neural networks, we use weights for each sensor to compensate the fewer samples from the magnetic and power sensors. These are computed using the "`compute_class_weight`" function of the Scikit-learn open source library. The weight corresponds to the total number of samples divided by the number of occurrences of the class. In addition, for all the methods the results show the mean accuracy achieved using a 5-fold cross-validation process (using 60% of the data for training, 20% for validation, and 20% for testing).

TABLE III
NUMBER OF EVENTS IN DATASET

Set of Sensors	*Number of Events* *ALZ*	*SPEED*
All sensors (15)	60961	121922
PIR sensors (7)	55302	110604

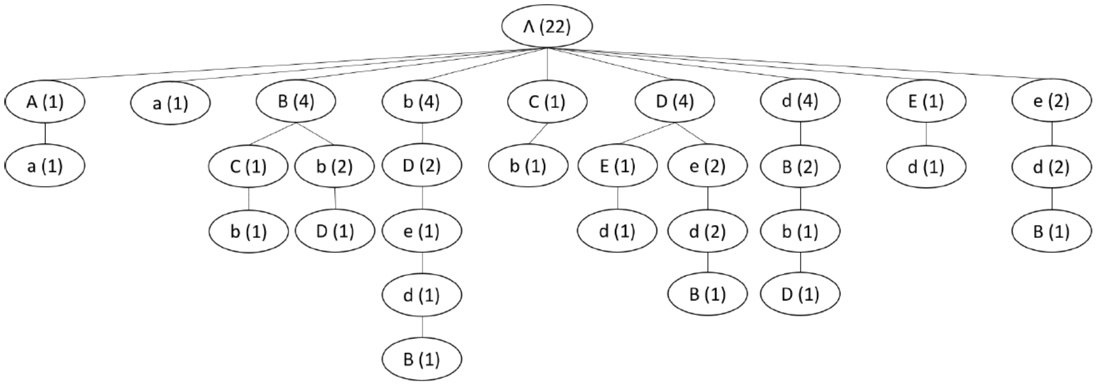

Fig. 3. Tree generated by the SPEED algorithm for the sequence "AaBCbDEdBbDedB".

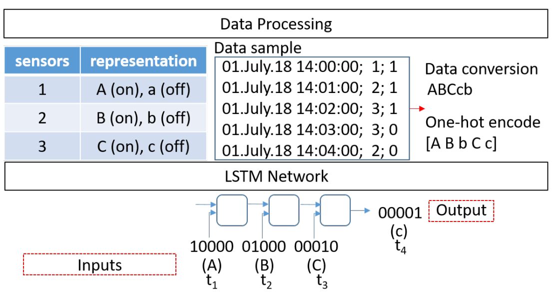

Fig. 4. LSTM network configuration.

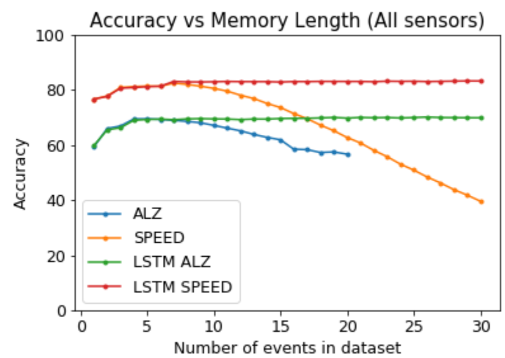

Fig. 5. Accuracy vs memory length for all algorithms on a dataset with all fifteen sensors.

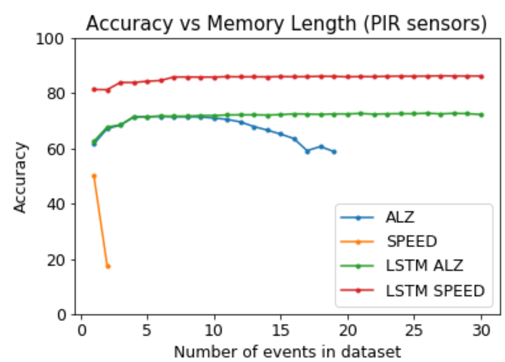

Fig. 6. Accuracy vs memory length for all algorithms on a dataset with seven motion sensors.

In the following, we examine the accuracy attained by the four algorithms (ALZ, SPEED, LSTM with ALZ-text and LSTM with SPEED-text) first against the memory length and then against the size of the dataset given in weeks. Further we compare our results to previous related work and summarize our discussion in this section.

B. Optimum memory length

We examine the accuracy achieved on the validation set for several values of memory length ranging from 1 to 30 events. This is performed first for a dataset containing events from all fifteen sensors (magnetic, power and motion) – Fig. 5 – and then for a dataset containing only the seven motion sensors – Fig. 6.

When using a dataset with fifteen sensors (Fig. 5), ALZ achieved a best accuracy of 69% while SPEED reached 82%. The optimal memory length was 4 events for ALZ and 7 for SPEED. The LSTM networks achieved accuracies of 70% and 83% when using ALZ- and SPEED-text, respectively. In both cases the optimal memory length is equal or larger than 8. The larger optimum memory length for LSTM indicates that probabilistic methods predict the next event based on fewer previous events, in other words the LSTM is more efficient at detecting patterns and correlations over a longer sequence.

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. 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 a dataset with fifteen sensors (Fig. 5) to the accuracy results of a dataset that contains only the seven motion sensors (Fig. 6). The accuracy curves for the LSTM network models show a similar dependency to memory length. The optimal memory length is 9 or larger. The LSTM with SPEED-text achieves 87% while with ALZ-text achieves 73%. The ALZ method also shows similar behaviour, and same optimal memory length of 4, with a higher accuracy of 71%. SPEED presents a very peculiar behaviour. The maximum memory length is 2. 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 "on" events.

C. Required amount of data for good accuracy

In the following, we investigate the behaviour of the accuracy with respect to the size of the dataset used for the complete process of training, validating, and testing the models. The accuracy results are computed within the testing set and using the optimal memory length found in the previous analysis. Fig. 7 and 8 show the results when the algorithms are applied to a dataset with all fifteen sensors and with only seven sensors, respectively.

The best accuracy is achieved for 10 weeks of data or above. There is no significant improvement in the accuracy for larger datasets, we therefore show the plots for dataset sizes up to 10 weeks for better clarity on the lower range of the graph.

We first examine the accuracy in the dataset with all sensors (Fig. 7). A peak accuracy of 83% was achieved by LSTM with SPEED-text, while the SPEED algorithm achieved a peak accuracy of 82%. The accuracy achieved by the LSTM with ALZ-text was considerably lower at 69%. In this case, stability is achieved much later than with the other methods. Finally, the ALZ method reached a top accuracy of 70% with 4 weeks of data. However, this method does not seem to be as stable as the other algorithms.

Note that the probabilistic methods attain a good accuracy (close to the peak accuracy) with only 2 days of data. By comparison, the LSTM networks need approximately 2-3 weeks of data to start approaching their top accuracy. This correlates well with the previously discussed ability of the LSTM to learn longer term patterns and dependencies, and attain better accuracy based on these.

Next we examine the accuracy results for the dataset using only the seven motion sensors (Fig. 8). As expected, the accuracy is higher since there are fewer sensors in this set. Moreover, motion sensor events happen sequentially, without intertwined events. The LSTM with SPEED-text achieved an accuracy of 87%, by far the best among the methods. The peak accuracy was achieved with slighlty less than 2 weeks of data. In addition, stability is reached with less data compared with the case in Fig. 7. The LSTM with ALZ-text and the ALZ achieved very similar accuracies of 73%, and 74% 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. 6. 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. The LSTM with SPEED-text also achieved a good accuracy with only a few days of data. However, the LSTM network with ALZ-text needed considerably larger amounts of data to attain acceptable prediction accuracy.

Most of the models reached a peak accuracy with 10 weeks of data or more. It may appear somewhat surprising that the best accuracy was reached for the same amount of data – 10 weeks – for both sets of sensors. However, as we pointed out earlier, the majority of the events in the dataset is in fact from motion sensors, and therefore, the two datasets are of similar size.

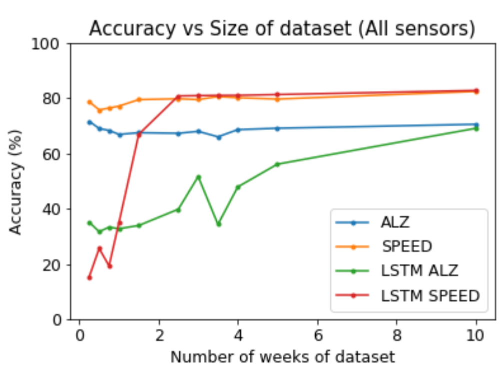

Fig. 7. Accuracy vs size of dataset for all algorithms on a dataset with all sensors.

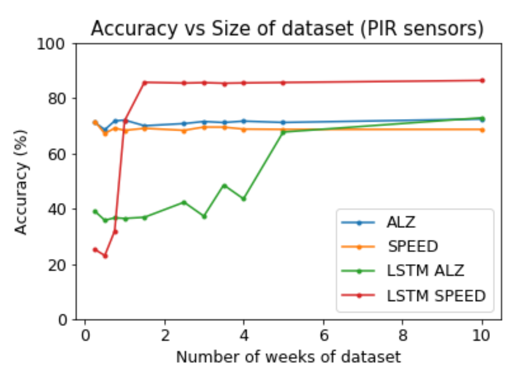

Fig. 8. Accuracy vs size of dataset for all algorithms on a dataset with motion sensors.

D. Execution time

Lastly, we examine the execution time to train and test the models. Table IV shows the results for the set with all sensors. In general, the probabilistic methods require longer processing time, although the ALZ needs only slightly longer time than the LSTM networks. SPEED requires eight times longer time to model than the LSTM with SPEED-text. Note that when using all the sensors these two models achieve similar prediction accuracy. However, SPEED reaches a high accuracy with much less data.

E. Discussion

We have applied two probabilistic methods on our data and have achieved comparable results to those obtained from the Mavlab testbed dataset. That testbed includes 50 sensors while our dataset was obtained from a real home with fifteen sensors, i.e. considerably fewer than the Mavlab testbed. In addition, in that work the same dataset was used both for training and for testing, which results in overfitting and overestimating the accuracy of the model. We use separate datasets for training, validation, and testing.

We have compared the performance of these two probabilistic methods with LSTM networks. To our knowledge this is the first time LSTM networks have been applied to this specific task. ESN, that is an RNN like LSTM, has shown good results [10], [11]. It is also the first time that probabilistic methods are compared to LSTM neural networks for sensor event prediction.

In our work, the best accuracy was achieved by the LSTM network with SPEED-text, 83% with all the fifteen sensors and 87% with seven motion sensors. In [12] an Elman network was applied to a dataset with 16 rooms and achieved peak accuracy of 91%, which is higher than our results. However, the dataset in that study was generated by the users themselves rather than being collected by sensors, a fact that is expected to lead to considerably fewer faulty events.

Our work showed that probabilistic methods can achieve a high prediction accuracy (close to their peak accuracy) with a relatively small amount of data (typically 2 days of data). LSTM networks require a larger dataset (about 3 weeks with SPEED-text and 10 weeks with ALZ-text) to reach good accuracy. 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 seven for SPEED was established in this work. On the other hand, LSTM networks base the prediction on a sequence of eight last 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 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 optimum.

TABLE IV
EXECUTION TIME OF ALGORITHMS

Algorithm	*Execution time (min)*
ALZ	2.8
SPEED	16.5
LSTM with ALZ-text	2.1
LSTM with SPEED-text	1.5

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 1% lower accuracy, however, after considerably longer training time. 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. 2 and 3).

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 2 so that only "off" events can be predicted reliably.

Another interesting finding is that more data than 10 weeks does 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.

Finally, regarding the number of sensors. A larger number of sensors can lead to better prediction accuracy to the extent that it entails more information to base the prediction on. A smaller number may, however, be preferable both in terms of reduced surveillance for the user, lower cost, and less nuance for the esthetics of the home. Our work shows that it is possible to achieve acceptable prediction accuracy with much fewer sensors than thought previously.

VI. Conclusions and Future Work

Activity recognition and prediction algorithms in smart home environments using binary sensors have been indicated to be useful for a number of functions. Most of the work reported in the literature has been carried out using data collected in lab environments and testbeds, with scripted activities. Such smart home testbeds typically include a quite large number of sensors, e.g. the Mavlab testbed deployed around 50 sensors [6].

In this paper we presented results on sensor sequence prediction using state-of-the-art methods: two probabilistic methods (ALZ and SPEED) and LSTM networks with both SPEED- and ALZ-text sequence inputs. Our dataset was obtained from a real home with an older adult ($>$ 65 years old) and with a relatively small number of sensors (15).

We compared all the methods with regard to a number of factors: the required number of preceding events to predict the next event (memory length), the necessary amount of data to achieve good accuracy and stability, the time used for training/testing, and the number of sensors in the dataset. To the extent of our knowledge, this is the first time such a comparison has been carried out. Our best implementation achieved an accuracy of 83% with LSTM with SPEED-text for a set with fifteen sensors in total – motion, magnetic and power sensors – and 87% with LSTM with SPEED-text as input for seven motion sensors. For the most accurate models using the SPEED-text, the LSTM required around 1/7 of the time SPEED required to do the modelling. On the other hand, the LSTM required about 3.5 weeks of data before reaching considerable (close to its peak) accuracy, whereas the probabilistic methods only needed 2 days of data for reaching considerable accuracy. The findings of our study can be useful for deciding which methods to use in accordance with project constraints (e.g. the number of available sensors, user privacy, etc.) and the area of application.

Clearly, a higher prediction accuracy is required before such algorithms can be applicable to real homes. Future work will include the time information as part of the input in order to improve the accuracy of our models. In addition, we will investigate the reproducibility of the best prediction model in other apartments with similar sensors and hence the variability of the predictions. Moreover, we will examine the possibility of using transfer learning methods across the apartments. These will be published in future work.

Acknowledgment

The authors would like to thank the residents and the housekeepers at the seniors' care unit Skøyen Omsorg+; 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] E. Zouganeli, F. Casagrande, T. Holthe, A. Lund, L. Halvorsrud, D. Karterud, A. Flakke-Johannessen, H. Lovett, S. Mørk, J. Strøm-Gundersen, E. Thorstensen, R. Norvoll, R. Meulen, M.-R. Kennedy, R. Owen, M. Ladikas, and E.-M. Forsberg, "Responsible development of self-learning assisted living technology for older adults with mild cognitive impairment or dementia," *ICT4AWE 2017 - Proceedings of the 3rd International Conference on Information and Communication Technologies for Ageing Well and e-Health*, no. Ict4awe, pp. 204–209, 2017.

[2] B. Winblad, K. Palmer, M. Kivipelto, V. Jelic, L. Fratiglioni, L.-O. Wahlund, a. Nordberg, L. Bäckman, M. Albert, O. Almkvist, H. Arai, H. Basun, K. Blennow, M. de Leon, C. DeCarli, T. Erkinjuntti, E. Giacobini, C. Graff, J. Hardy, C. Jack, a. Jorm, K. Ritchie, C. van Duijn, P. Visser, and R. C. Petersen, "Mild cognitive impairment–beyond controversies, towards a consensus: report of the International Working Group on Mild Cognitive Impairment." *Journal of internal medicine*, vol. 256, no. 3, pp. 240–246, 2004.

[3] 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 of Things Journal*, vol. 4, no. 3, pp. 636–644, 2017.

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

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

[6] K. Gopalratnam and D. J. Cook, "Online Sequential Prediction via Incremental Parsing: The Active LeZi Algorithm," *IEEE Intelligent Systems*, vol. 22, no. 1, 2007.

[7] M. R. Alam, M. B. Reaz, and M. A. Mohd Ali, "SPEED: An inhabitant activity prediction algorithm for smart homes," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, vol. 42, no. 4, pp. 985–990, 2012.

[8] L. R. Rabiner and B. H. Juang, "An Introduction to Hidden Markov Models," *IEEE ASSP Magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[9] J. G. Cleary and I. H. Witten, "Data Compression Using Adaptive Coding and Partial String Matching," *IEEE Transactions on Communications*, vol. 32, no. 4, pp. 396–402, 1984.

[10] A. Lotfi, C. Langensiepen, S. M. Mahmoud, and M. J. Akhlaghinia, "Smart homes for the elderly dementia sufferers: Identification and prediction of abnormal behaviour," *Journal of Ambient Intelligence and Humanized Computing*, vol. 3, no. 3, pp. 205–218, 2012.

[11] S. Mahmoud, A. Lotfi, and C. Langensiepen, "Behavioural pattern identification and prediction in intelligent environments," *Applied Soft Computing Journal*, vol. 13, no. 4, pp. 1813–1822, 2013.

[12] L. Vintan, A. Gellert, J. Petzold, and T. Ungerer, "Person movement prediction using neural networks," in *Proceedings of the KI2004 International Workshop on Modeling and Retrieval of Context*, vol. 114, no. 4, 2004, pp. 1–12.

[13] J. Petzold, F. Bagci, W. Trumler, and T. Ungerer, "Next Location Prediction Within a Smart Office Building," *Cognitive Science Research Paper Unversity of Sussex CSRP*, vol. 577, p. 69, 2005.

[14] E. Nazerfard and D. J. Cook, "CRAFFT: An Activity Prediction Model based on Bayesian Networks," vol. 33, no. 4, pp. 395–401, 2015.

[15] B. Minor and D. J. Cook, "Forecasting occurrences of activities," *Pervasive and Mobile Computing*, 2016.

[16] M. T. Moutacalli, A. Bouzouane, and B. Bouchard, "Sensors activation time predictions in smart home," *Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, p. 37, 2015.

[17] N. E. Elhady and J. Provost, "A Systematic Survey on Sensor Failure Detection and Fault-Tolerance in Ambient Assisted Living," *Sensors*, vol. 18, no. 1991, p. 19, 2018.

[18] J. Ziv and A. Lempel, "Compression of Individual Sequences via Variable-Rate Coding," *IEEE Transactions on Information Theory*, vol. 24, no. 5, pp. 530–536, 1978.

[19] J. Martens, "Generating Text with Recurrent Neural Networks," *Neural Networks*, vol. 131, no. 1, pp. 1017–1024, 2011.

[20] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech Recognition With Deep Recurrent Neural Networks," *Icassp*, no. 3, pp. 6645–6649, 2013.

[21] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *Journal of Machine Learning Research*, vol. 3, no. 1, pp. 115–143, 2003.

[22] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[23] A. Graves, "Generating Sequences With Recurrent Neural Networks," *arXiv*, 2014.

[24] 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 *Proceedings of The 59th Conference on Simulation and Modelling (SIMS 59)*, 2018, pp. 236–242.