Occupant Behaviour Pattern Modelling And Detection In Buildings Based On Environmental Sensing

Author:
Jose Luis Gómez Ortega
Director of Studies:
Liangxiu Han
Supervisor:
Nicholas Bowring

A thesis submitted in partial fulfilment of the requirements of the Manchester Metropolitan University for the degree of Doctor of Philosophy

School Of Computing, Mathematics And Digital Technology

2017

Abstract

Occupant presence and behaviour have a significant impact on building energy performance. An occupant present in a building generates pollutants like CO_2, odour, heat, which can directly change the indoor environment. Because of this change, the occupant may interact with the building environment to maintain the comfort level, for example, he or she may turn on air conditioning systems. Today's Building Energy Management Systems (BEMS) are usually operated based on a fixed seasonal schedule and maximum design occupancy assumption but fail to capture dynamic information. This is both costly and inefficient.

Recent efforts on exploitation of environmental sensors and data-driven approaches to monitor occupant behaviour patterns, have shown the potential for dynamically adapt BEMS according to real user needs. Furthermore, this occupant information can also be used for other applications such as home security, healthcare or smart environments. However, most of existing models suffer from inaccuracy and imprecision for occupant state classification, could not adaptively learn from real-time sensor input and they mainly focused on single occupant scenarios only. To address these issues, we present a novel data-driven approach to model occupant behaviour patterns accurately, for both single occupant and multiple occupants with real-time sensor information. The contributions can be summarised as follows:

Firstly, we have conducted a thorough benchmark evaluation of classification performance of state-of-the-art Machine Learning (ML) methods and occupant related publicly available datasets. **Secondly**, based on the findings in literature and our own experimental evaluations, we have developed a novel dynamic hidden semi-Markov model (DHSMM), which can accurately detect occupant behaviour patterns from sensor data streams in real-time. **Thirdly**, built upon the online DHSMM model, we have developed a novel incremental learning approach to allow dynamically learning over streaming data. **Finally**, we have conducted an experimental evaluation of our proposed model Online DHSMM Multi-Occupant for occupancy detection for both single and multiple occupants. We have validated our approach using real datasets and the experimental results show our proposed approach outperforms existing methods in terms of classification accuracy and processing time/scalability.

To the best of our knowledge, we have first developed a HSMM-based incremental online learning approach to fast and accurate learn building occupant patterns over streaming data for both single and multiple occupants in a holistic way. Additionally, our approach significantly improves the classification accuracies of traditional Markov models (over 10% accuracy increase, while maintaining the model complexity and performing multi-occupant detection).

Acknowledgements

Firstly, I would like to express my sincere gratitude to my director of studies, Dr. Liangxiu Han. Her sustained support throughout this research has proven to be both endless and priceless. I am fortunte for having had her insight and guidance to proceed through this thesis successfully.

I also want to thank my supervisor Prof. Nicholas Bowring and Mr. Nick Whittaker for their help and support, and the staff at the SCMDT with an especial emphasis to the other PhD candidates at the FUNDS group.

This has been a long and challenging process, but I am blissful for having had my 'Spanish MMU Fellows' at all times beside me. Thanks for the endless talks, for the help, support and friendship, and basically for making things a lot easier and enjoyable. This can be also extended to my *Creature Comfort* family, whose fabulous members have taken care of me wonderfully con queso.

To the rest of my family and friends sending me their love and support (and ham) from Spain (especially to Marc and Diego: thanks for hanging on without your Tito). You are far, but at the same time you are always close. Thanks everyone.

And finally, to the person that made all this possible: Sonia. It is plainly unthinkable to imagine all this process without you in the picture. Thanks for making things real, for standing beside me in the hardest moments and for laughing with me in the good ones. Gracias amorín.

Contents

List of Figures

List of Tables

Publications

1. J. L. G. Ortega, L. Han, N. Whittacker and N. Bowring, "A machine-learning based approach to model user occupancy and activity patterns for energy saving in buildings," *2015 Science and Information Conference (SAI)*, London, 2015, pp. 474-482.

2. J.L.G. Ortega, "Occupancy Pattern Detection for Energy Efficient Buildings", *MMU Scientist Research Bulletin*, Issue June 2014, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester, 2014, pp. 11.

3. J.L.G. Ortega,L. Han, "Modelling Occupant Activity Patterns For Energy Saving In Buildings Using Machine-Learning Approaches", *ZEMCH International Conference Proceedings*, Lecce, 2015, pp. 241-253.

4. J.L.G. Ortega, L. Han, N. Bowring, "Modelling and Detection of User Activity Patterns for Energy Saving in Buildings", *Chapter 9, Springer Book Series - Studies in Computational Intelligence*, 2016, pp. 165-185.

5. H. Altan et al., "Chapter 7 – Energy Use in Housing, ZEMCH: Towards the Delivery of Zero Energy Mass Custom Homes"; *Springer Tracts in Civil Engineering*, 2016, pp. 175-207.

6. J.L.G. Ortega, L. Han, N. Bowring, "A Novel Dynamic Hidden Semi-Markov Model (D-HSMM) for Occupancy Pattern Detection from Sensor Data Stream", *2016 8th IFIP Conference on New Technologies, Mobility & Security (NTMS)*, Larnaca, 2016, pp. 1-5. **(Best Paper Award, New Technologies Track).**

Abbreviations

ADL	**A**ctivities of **D**aily **L**iving
ANN	**A**rtificial **N**eural **N**etwork
APR	**A**ctivity **P**attern **R**ecognition
BEMS	**B**uilding **E**nergy **M**anagement **S**ystems
BW	**B**aum-**W**elch **A**lgorithm
CDA	**C**hunk **D**ata **A**pproach
CDF	**C**umulative **D**ensity **F**unction
DHSMM	**D**ynamic **H**idden **S**emi-**M**arkov
FB	**F**orward-**B**ackward **A**lgorithm
HMM	**H**idden **M**arkov **M**odels
HSMM	**H**idden **S**emi-**M**arkov **M**odels
KNN	**K**-**N**earest **N**eighbours
ML	**M**achine **L**earning
MLP	**M**ulti-**L**ayer **P**erceptron
NB	**N**aive **B**ayes
OBPM	**O**ccupant **B**ehaviour **P**attern **M**odelling
OPD	**O**ccupancy **P**attern **D**etection
PDF	**P**robability **D**ensity **F**unction
PIR	**P**assive **I**nfra-**R**ed
RFID	**R**adio **F**requency **ID**entification
SVM	**S**upport **V**ector **M**achines
TSA	**T**ime-**S**lice **A**pproach

Chapter 1

Introduction

This chapter presents an overview of the background of this research introducing current limitations and research questions to be solved as well as our novel contributions.

1.1 Background

Buildings are one of the major sources of CO_2 emissions, accounting for over 40% of the total [3]. Occupants have a major impact on the total energy consumption performance of buildings, both from internal gains and from the interactions with their surroundings. Many works e.g. [4][5][6][7][8], reported that energy cost savings could be significantly reduced when considering occupant data in the loop of Building Energy Management Systems (BEMS).

Today's building systems such as lighting or heating, ventilation and air-conditioning (HVAC) systems are still regulated based on fixed schedules or peak assumptions that overestimate occupancy which cause large amounts of energy wastage [4]. Therefore, it is crucial to develop strategies to regulate BEMS based upon real occupant needs, in pursuance of maximising user comfort [9] while maintaining high levels of building energy efficiency. To address these issues, 'live' occupancy detection is important for improving energy efficiency and consequent reduction of greenhouse gases emissions.

For example, there can be scenarios with several sensors (e.g. indoor and outdoor temperature, humidity, CO_2 levels and information about user thermal comfort) as in

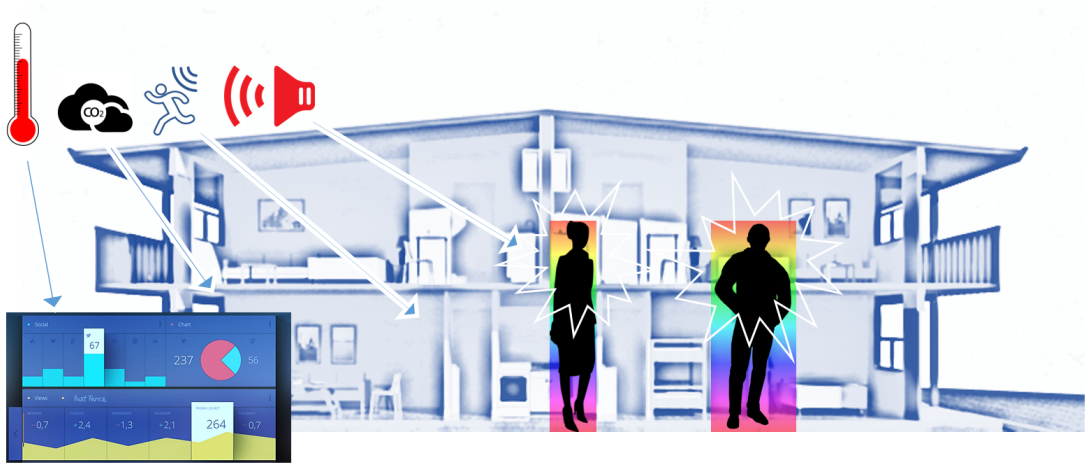


FIGURE 1.1: Sensors and ML models will help us to model occupant/ambient interactions.

the diagram shown in Fig. 1.1. The different sensors will capture information related to indoor conditions and variations generated by occupants and the interactions with their surroundings. That information will be used to regulate energy systems, monitor occupant states or detect abnormal or potentially hazardous situations among others.

Much effort has been devoted to occupant behaviour pattern modelling (such as occupancy patterns or activities) over the last years [10][11][12][13][14]. Particularly, models based on Machine Learning (ML) techniques coupled with ambient sensors have demonstrated potential to capture occupant behaviour patterns information that can be used to adaptively regulate BEMS according to realistic users' needs. The aim of these systems is to understand what are the real occupant levels of occupancy or the activities they are performing in a building. This information will provide insights that will enable more efficient and effective decision making on how to regulate the building systems.

The existing approaches incorporate various sensor data into models to detect user behaviour in the form of presence/absence, number of occupants or activity recognition; always within the context of indoor building environments. These models are designed to analyse sensor readings and classify occupant states from those sensor signals. Fig. 1.2 illustrates how sensor signals could be used to train a mathematical model. Here, we can see that the data can be used to train/evaluate the model and make predictions of the potential actions and occupant/building interactions. These models need to be able to overcome the specific practical problems that arise from the use of datasets containing

this type of data such as data imbalance, missing values, incompleteness data among others.

Due to these, there are occasions where a deeper and more complex analysis is needed to find the underlying relationships between the data features and the expected outcomes. In order to address this, ML techniques can be used to capture the potential patterns in the data (provided that we have significant amounts of data and there are actually patterns in the data that the algorithms can learn from). For example, a rainy day can alter the patters of occupants in terms of the clothes they wear, subsequently affecting the way they operate HVAC systems, windows, etc. However, if in the available training datasets there are no samples containing rainy days data, the ML model will not be able to capture these patterns.

Among these approaches, Hidden Markov Models (HMM) [15] and Hidden Semi-Markov Models (HSMM) [16] have gained increasing popularity for their ability to model sensor data from buildings as observable discrete temporal sequences and being able to infer occupant (hidden) states like presence or absence, based on the analysis of all sensor events. For example, early approaches such as the work of Page et al. [10] modelled occupancy profiles in an office building applying Markov chains or the HMM-based model in [17], which was designed to regulate different smart-home systems in context-aware scenarios. Other models based on HMM models and more recent extensions also became increasingly popular for occupant behaviour pattern recognition as described in [18]. Hongeng et al. [19] proposed a video based event recognition where an adaptation of HMM algorithms was used to detect human activities through video data. Dong's work [14] presented a HSMM-based occupancy modelling approach for energy saving and comfort management by detecting events extracted from multiple sensor data. Later, in [20], they used a various models including HMM and other machine learning approaches to detect occupants in an office building based on multiple ambient sensors, noting that different features (sensors) are potentially more significant for the final state classification decision and that HMM can also be used for predictive purposes. The work in [21] provided an interesting activity recognition benchmark work, where HMM and HSMM models were evaluated and compared with other state-of-the-art approaches to study modelling issues and performance based on contextual aspects such as time granularity.

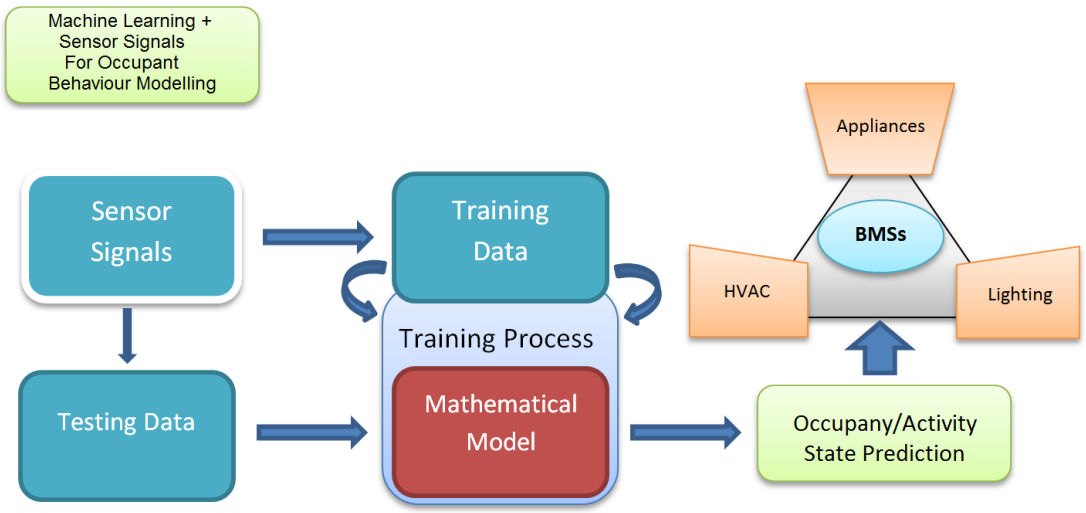


FIGURE 1.2: Machine learning techniques modelling schematic data flow.

1.2 Current Limitations

In spite of recent advances, most of the aforementioned existing models suffer from inaccuracy and imprecision in terms of the classification of occupant states based on sensor information. This is due to the fact that typical datasets from occupants in buildings present challenges for modelling states due to issues such as the variety of data nature (e.g. use of different sensing devices, multiple sensor network topologies, data acquisition techniques, etc.). In addition, many approaches were built upon static datasets and could not adaptively learn from real-time sensor input and focus on single occupant only. In the case of works using HMM models, some specific limitations arise since HMM-based approaches model state dwelling time by allowing the system to self-transition each time the system makes an inference (from state S_i to S_i) based on the probability of remaining in the same state. This means that remaining in the same state for a period of time involves multiplying the probability of remaining on that state each time the self-transition occurs. Hence, state duration is inherently exponentially distributed. However, in real scenarios, the state duration should be better captured by using different temporal distributions other than exponential distribution [22][16].

HSMM expands HMM capabilities by introducing state duration. HSMM-based approaches allow to calculate the most probable dwelling time d based on explicitly modelled duration distributions, during which the state remains unchanged. However,

HSMM-based approaches present limitations when attempting to make occupancy detection in an online fashion. This is due to the fact that these models traditionally infer probable states based on whole sequences of observable data. This poses a real challenge for state classification using streaming data as future observable data is not available when a new state is reached. Therefore, state prediction has to be decided with the uncertainty of what observable sequence follows. In addition, Markov models treat all sensor readings as a unique feature, thus assuming all sensors potentially provide the same 'amount' of relevant information to the system. However, in real world scenarios, individual sensor contributions to state prediction is often different and can be conditioned by factors such as sensor spatial location, system/sensor resolution, data nature or sensor reading ranges to name a few.

Moreover, existing models were mainly used for single occupant only, thus never attempting to model scenarios including more than one occupant at the same time. There is therefore little work done in relation to multi-occupant behaviour pattern detection, which still remains one of the most important challenges for this type of modelling approaches [23].

1.3 Research Questions

All the limitations discussed above motivate the following research questions:

- How can we improve the classification accuracy of the existing approaches?
- How can we detect occupant behaviour patterns using real-time streaming data?
- How can we improve the efficiency of the model parameter updating using online incremental learning to accommodate new data observations? and, finally
- How can we ensure these models will perform well when detecting more than one occupant?

1.4 Contributions

This research mainly focuses on the development of novel approaches to fast & accurately occupant behaviour pattern detection over stream data in real-time. Our contributions lie in the following aspects:

1) Benchmark experiments

   To improve the accuracies of the existing works, we need to understand the current state-of-the-art approaches and limitations. To do so, we have conducted a thorough benchmark evaluation using data from sensors in buildings for occupant behaviour pattern modelling and detection. Based on data from multiple publicly available datasets, we have applied well-known machine learning approaches to evaluate different model performances.

2) Development of a novel online dynamic hidden semi-Markov model (DHSMM)

   We have developed a novel dynamic hidden semi-Markov model (DHSMM), which can detect occupant behaviour patterns from sensor data streams in real-time, increasing the prediction accuracy compared to traditional Markov models. Our approach extends HMM and HSMM including a new dynamic state duration estimation and a new observation model including a weighted function designed to improve the model performance when using partially available observations instead of using the whole set of observations. The proposed model has been validated in an online setting using real datasets which contained multiple sensor data and occupancy states labels.

3) Development of a new incremental learning approach based on DHSMM

   To enable online parameter updating, we have developed a novel incremental learning approach to dynamically learn over streaming data and fast, accurately predict occupant behaviour patterns. As this model is based on a HSMM approach, our online DHSMM needs to incrementally update an additional parameter: duration model. To do so, we have developed a novel approach to update the DHSMM parameters based on an approximation of the β variable in the Baum-Welch algorithm. For the duration model, we have applied a function based on Bayesian

inference techniques to update the different duration model parameters in an incremental way. The incremental updating is applied to the parameters that define the probabilistic distributions used to model state duration. We have used this approach with two of the most well-known probabilistic functions: Gaussian and Gamma. We have validated the results with real datasets and the experimental result shows the proposed incremental learning approach significantly outperforms non-increment approach over streaming data while maintaining the same level of accuracy as the non-incremental learning approach.

4) Development of online multiple occupancy pattern detection

We have developed a model framework for occupant pattern detection for both single and multiple occupants. To do this we present two different on-line DHSMM multi-occupant approaches including a multi-layered model and a single-layered model for multi-occupant activities detection. We used a publicly available dataset that includes human activities performed by two individuals in the same space to see how our model handles this challenging setup. Results have shown our approach is able to accurately detect occupancy patterns in the context of both single occupant and multiple occupants.

1.5 Restrictions

Occasionally, sensor data have the potential to contain personal information about the occupants present in the spaces these sensors are monitoring and extracting the information from. When this happens, it can be considered as an intrusion in people's privacy. This is particularly relevant in public spaces where people never gave their consent to be privately monitored or when dealing with vulnerable populations such as children or patients from hospitals or other health related institutions. Moreover, sometimes the legislation is unclear and is different in each country.

In order to not having to deal with potential intrusiveness issues, and for the sake of developing systems that will never pose a risk to people's privacy, this research mainly focuses on non-intrusive, off-shelf sensors such as motion or temperature. Due to low cost and effectiveness and the absence of the mentioned privacy issues common when

using rich sensors such as video-cameras, RFID or wearables, we limit our study and proposed methodologies to the use of only non-intrusive sensor data [24].

1.6 Thesis Outline

The rest of this thesis is organised as follows:

Chapter 2 contains a comprehensive literature review concerning all types of occupancy detection and activity recognition models and potential scenarios of application for these approaches. We define the scope of the research, provide a description of contextual factors such as scenarios, discuss about the datasets including sensors, data nature and granularity and indicate publicly available related datasets. We have critically analysed and evaluated the current advances and their limitations.

Chapter 3 presents a thorough benchmark evaluation on existing well-known machine learning approaches. We conducted a series of experiments using publicly available datasets to evaluate the performances of several machine learning techniques and how these can be affected by other factors aside mode parameters such as data nature and pre-processing techniques and evaluated their performance and suitability when processing occupancy data in different building scenarios.

Chapter 4 introduces our novel online dynamical HSMM model (DHSMM). We analyse the pros and cons of using HMM and HSMM models and how our DHSMM can be used to address the original methods' limitations, particularly for a good generalisation of the model when used in different scenarios and for real time occupancy and activity detection.

Chapter 5 includes a new approach to perform DHSMM online incremental learning parameters. We introduce how the Markov models can be used to solve the so-called three Markov problems and how we extended these traditional approaches and developed our incremental online DHSMM approach, enabling online incremental learning of model parameters over streaming data.

Chapter 6 presents a multi-occupancy model framework based upon our online incremental DHSMM. Our approach can detect occupancy patterns from both single and multi-occupant data addressing the issues with data association and state interaction.

Chapter 7 summarises the contributions of this work and provides directions for future work.

Chapter 2

Literature Review

In this chapter, we present a survey on previous research regarding Occupant Behaviour Pattern Modelling (OBPM), paying special attention to the mathematical modelling methodologies presented. We have focused on previous works addressing human behaviour pattern modelling in buildings using only non-intrusive sensors (e.g. motion, humidity, sound, CO_2,...).

2.1 Occupant Behaviour Pattern Modelling Definition

We define Occupant Behaviour Pattern Modelling (OBPM) as the modelling of user presence and absence and activity patterns in buildings based on sensor information. These models are fed with data collected through ambient sensors, and occupant patterns are modelled using different techniques such as stochastic processes or machine learning algorithms.

OBPM include all works that, based on supervised learning approaches, use sensor data information collected by sensors deployed in buildings in order to predict occupant patterns. The models will be trained with the available data, and then other training data subsets or new collected data will be processed by the model. The model outputs will consist in the classification of several occupant traits. Once this information is available, it will be included in the loop of decision of automatic building systems that will adapt any their operation according to the information and therefore to real occupant needs.

2.1.1 Types Of OBPM

After a comprehensive survey on the OBPM literature, we found a noticeable differentiation between approaches. Following this intuition, we have proposed the categorisation of these models into two types:

a) Occupancy Models:

- Presence/Absence: Binary classification on whether or not there are any occupants in a space at a determined time.
- Number of occupants: Multiple classification of as many classes as occupants are present in a determined location at a time.

b) Activity Models: Activities of Daily Living (ADLs) are defined as the low level activities occupants can be performing on a daily basis in a domestic building scenario [25][26]. Approaches based on activity modelling need to define beforehand which are the activities expected (e.g. *sleeping* or *having_breakfast*) and models are used to predict what activity is being performed based on sensor readings (classification of as many activities defined in the dataset).

2.1.1.1 Occupancy Modelling (OM)

Occupancy pattern detection models (occupancy models for short) definition comprises modelling approaches that focus on detecting presence/absence and/or number of occupants and the identification of occupancy patterns, such as time of arrival or departure or peak occupational times.

2.1.1.2 Activity Modelling (AM)

Activity models include those which perform recognition of occupant's activities of daily living (ADLs) such as *sleeping*, *leaving_home*, *watching_television*, *having_shower* or any user activity which might have a link to the final energy consumption.

2.1.2 Scenarios

Looking into previous works' typical scenarios, we noted a usual natural gap between scenarios for OM and AM. Occupancy models tend to be used in non-domestic buildings (smart buildings) including general purpose spaces such as offices, labs or lecture rooms. However, activity models are usually intended to be used in domestic buildings (smart homes). This is basically due to the fact that many ADLs are not expected to occur in a smart building scenario (people working in an office are not expected to cook, sleep or have a shower) whereas they are totally normal in a smart home. Conversely, big building scenarios tend to focus on the presence of occupants as the information will be used to regulate lighting or HVAC systems, which are typically more concerned about people being there rather than what they are actually doing.

Nevertheless, there is not a major restriction saying that necessarily all occupancy models are based on non-domestic scenarios and activity models are based on domestic scenarios. Yet, in the majority of the works surveyed this is a clear trend. In fact, another natural consequence of this discernment is that in normal houses it is usual to consider just one occupant (one user performs various activities) while multi-occupancy (diverse occupants) is normally addressed in non-domestic scenarios.

2.1.3 Sensors

In the previous section, we have discussed how model type (occupancy/activity) had a correlation with the typical scenario. Supporting this idea, we found that the nature of the sensors used in previous works also points towards this hypothesis.

We define ambient sensors as the devices used to collect information about our surroundings and transform it into signals, which can be later processed and incorporated into mathematical models for their training and further operation. In this work, we focus on the most popular non-intrusive choices; which are usually simple ambient sensors such as passive infraRed PIR (for motion), radio (for proximity), ambient light, humidity, temperature, or contact sensors. Although there is no formal constraints about the use of different sensors for any type of models, most previous works suggest that there are some sensors that are not suitable for occupancy models. For example, a sensor to monitor a microwave open door or to detect water from the shower is useful to predict activities

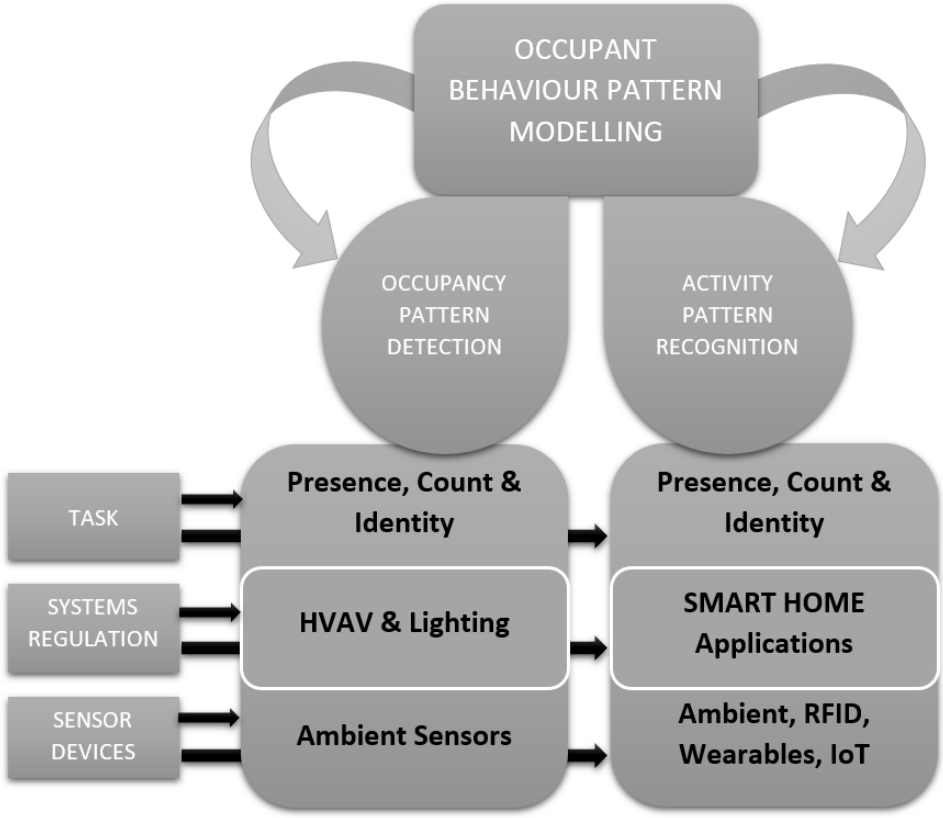


FIGURE 2.1: The literature shows a natural differentiation between occupancy and activity models in terms of the classification task, scenario and sensors commonly used.

within a house preformed by an occupant, but it will make a meagre contribution if we want to monitor occupancy patterns of an entire office building section.

As it will be discussed in following sections on this chapter, generally simple ambient sensors as the ones mentioned above, although with some limitations, can be used to detect occupancy and activities successfully. However, sensor combinations have shown potential to mitigate some of the issues present in systems using just one sensor -this will be discussed in detail in the following sections. A detailed list of sensors and applications can be found in [13] and [27]. Fig. 2.1 shows the category of occupancy and activity model features in terms of typical sensors and systems to regulate, as well as their usual classification tasks.

2.1.4 General Approaches For Occupant Behaviour Pattern Modelling (OBPM)

The existing OBPM approaches are mainly based upon the detection of occupants and activities using sensor data in buildings. Mathematically, this translates into a classification problem in which the inputs are sensor readings and the outputs could be presence/absence, number of occupants or ADLs.

The works covered in this chapter are mainly based on stochastic mathematical models and machine learning based algorithms; used to process occupant and ambient data and to classify occupancy and activity states. Models such as naive Bayes [28], hidden Markov and semi-Markov models [15] and conditional and semi-conditional random fields [29] are probabilistic approaches, which have been extensively used for pattern recognition problems in the past. However, other machine learning methods based on numeric optimisation and marginality have been proposed with promising results such as support vector machines [30] or artificial neural networks [31]. Other common techniques are algorithms based on multi-agent systems, which are used to divide the automation of BEMS in several agents and where the learner agent is usually based on a machine learning classifier.

The most frequent machine learning algorithms for pattern modelling found in the literature reviewed are:

- Naive Bayes (NB)
- Decision Trees (DT)
- Logistic Regression (LT)
- Hidden Markov Models (HMM)
- Hidden Semi-Markov Models (HSMM)
- Artificial Neural Networks (ANN)
- Support Vector Machines (SVM)
- k-Nearest Neighbours (kNN)

Although these might be considered as the most popular OBPM approaches, other successful methods also found in previous researches include conditional random fields (CRF), genetic programming (GP) or models based in stochastic processes or heuristics.

In the following sections, we will present a detailed literature review on existing occupant behaviour pattern model approaches.

2.2 Occupancy Modelling Approaches

2.2.1 Occupancy Modelling Definition

Many occupancy modelling previous works are focused on the modelling of the so called spatio-temporal properties which include: location, number of occupants or tracking; since these can be captured through ambient sensing devices [13] unlike other human behaviour traits such as psychological or emotional. The main attractiveness of these models is that the outputs generated can be used to automate BEMS, meaning that the building systems can be regulated more effectively and efficiently.

2.2.2 Classification Of OM

Based on the literature surveyed and attending to the task the models are intended to perform, we propose a classification of occupancy models into two types:

- ***Offline Models:*** These approaches include models that use data to recreate time-related occupancy sequences of the most likely states in an offline setting. Models can be later embedded into energy related or building simulation tools such as ESP-r or Energy+, or used to regulate BEMS adapting their functioning to the occupancy patterns suggested by the model. Offline models provide: 1) Information that can be used to create accurate occupancy profiles in order to make assumptions and predictions based on previous findings; 2) Information which will be used to monitor 'live' occupancy and regulate systems in real-time. Although these models are designed to adapt to real user needs, once built their predictions are used in a static way. Since these approaches are intended to study the process that generates the data in that particular way, most of them are based

on generative models where the signal producing the data is studied to better understand relationships between inputs and outputs and the 'true' signal that originated the data. Offline models are based on batch learning approaches, which have the advantage that they can analyse information from data such as class balance, prior probabilities or how the inputs distribute for all the data (supposing that we have enough samples to assume that the total of the data accurately represents the properties and underlying patterns of the population from which has been originated).

- ***Online Models:*** This category includes the models designed to perform 'live' occupancy detection. This approach consists in feeding data in real-time to estimate at any given time the most likely state or activity. After the model have been trained, the data is processed in a streaming fashion. Although most of these models could work in an offline setting, these are specifically designed to be able to cope with scenarios in which the data is fed on an stream fashion into the model. Therefore, these models don not have access to future data in the moment of making an state prediction, so they need to make predictions with partial data available. These models are often equipped with ways where they can gradually learn from each new sample by updating the model parameters evaluating the model response step by step (incremental learning). While offline models are useful to study and reproduce previous occupant patters in buildings, online methods allow to use OBPM in real world applications that can benefit from a real-time state prediction. Systems such as smart homes can be made responsive of current occupant needs to improve the user experience or critical decisions can be supported by these models in applications such as elderly monitoring or security systems.

In following sections will see that similarly to occupancy models, activity models can be divided in offline and online approaches (See Fig. 2.2).

2.2.3 Offline Modelling Approaches Of OM

One the of pioneering approaches to measure the relationship between occupancy and environmental aspects was proposed in [32]. In this study, PIR sensor data was fed into

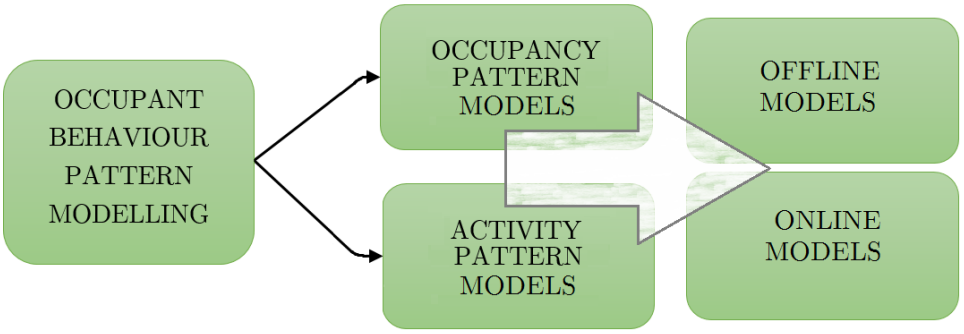


FIGURE 2.2: OPBM can be divided into occupancy models (OM) and Activity Pattern Recognition (AM), which can be subdivided into Offline and Online models.

an algorithm based on heuristic rules to control tasks in an offices building environment. At the time, the major limitations were the ineffectiveness of the sensors in terms of the high number of false detections (noisy data) and the technical problems for their implementation [33][10][34][35]. However, despite these problems, these early approaches showed that addressing occupancy could be helpful to improve building systems usage efficiency through a more realistic use of the BEMS.

In the work in [33], a model for the regulation of lighting systems was proposed. Using motion sensors, they claimed this model could reduce lighting operating costs between 35% and 75% when adapting its use to an occupancy schedule based on real user occupation data. The modelling approach consisted in the use of pareto distributions for lighting times delay modelling and Grey systems to model lighting sequence data.

These initial developments motivated a growing interest in studying the building systems actual usage rather than assuming fixed schedules, noting a potential to make BEMS use more effective and efficient. Based on this, following approaches included stochastic processes techniques to build their models, as in Dodier et al. [34] and Page et al. [10]; which also used data from PIR sensors. The idea of modelling human occupancy patterns as stochastic processes, was intended to provide the algorithms with the ability to incorporate the random nature of human behaviour.

Dodier et al. [34] approach consisted of an algorithm based on Bayesian theory and Markov Models. Motion sensors such as PIR were the first most popular choice due to their ubiquity and technical simplicity. However, they complemented the motion data with sensors connected to the phone in the office to overcome periods of low user

mobility, making the predictions on those intervals highly dependant on specific sensor events (potentially solved with rules).

Page et al. also used motion sensors (PIR) deployed in the LESO building. In this case, the dataset contained almost three years of occupancy data from sensors deployed in over 50 office spaces. Page's work was specially relevant due to the large amount of data they managed to collect (occupancy data was really scarce at the time) and the inclusion of Markov chains. In this work, they studied the behaviour of occupants in an office under an stochastic scope, thus pioneering the research of time-related aspects such as arrival and departure time, weekends and periods of long absence. This work was particularly relevant due to the amount of data used for training the model. However, these approaches were only concerned in the creation of simulated profiles, and the use of this data was limited to simulation tools.

These researches however, showed that occupancy patterns were dynamic and dependent on human behaviour uncertainty, and therefore crucial to understand occupation and its impact in a more realistic way. This ideas represented an important breakthrough since they marked the starting point of the more advanced techniques we have nowadays.

Following these previous research lines, other authors also combined sensor data and random processes for human behaviour modelling in the future. The work in [36] presented an algorithm constructed from LESO building data and Markov decision processes; in this occasion focused in window openings regulation. They identified windows as a critical element in the final performance of buildings, hence the need to address their apertures in order to improve energy efficiency. Although they claimed high levels of performance, these models were only focused of profile creation and only suitable for scenarios where windows were a determinant factor in terms of triggering the occupants responses.

More recently, [37] incorporated novel techniques based on a genetic programming algorithm to model occupancy based on the LESO building dataset as in [10][36]. This model was designed to perform presence/absence classification and improved the accuracies from previous models (over 80%) as well as they proved how temporal context is essential for the modelling of human activities in buildings. TIme started to appear as a significant feature for the success of modelling such occupancy states.

In [38], an HMM based algorithm was designed to map occupancy patterns for the control of lighting systems in an office building. This model could also be used to generate data and patterns instead of making state prediction. This information was later introduced into building energy simulation software such as ESP-r or Energy+ to predict energy consumption under certain conditions. Models progressively were increasing their complexity and frequently their performance. However, accuracy levels were far from perfect and many times the robustness of the models was limited tp the use of specific sensors and layouts.

The work in [39], presented an agent-based system based on synthetic data to estimate energy consumption. This study was more focused on appliances and equipment behaviour rather than actual occupant behaviour. However, in this occasion they had to back their predictions in smart meter data ad an input, making this approach not suitable for buildings lacking this type of devices.

In [40], they proposed another multi-agent model (the learner agent being based upon Markov Decision Problems), arguing that an energy reduction of the 20% could be achieved by introducing simulated occupant comfort patterns. In this case, the data was collected through temperature sensors in a university building recreating offices conditions. This work was one of the few that actually related their model to an actual predicted value of energy savings. It also showed how recent modelling approaches used multi-agents to the creation of complex systems in which the machine learning classifiers were only a part of the system.

More recently, in [41] another multi-agent based approach was proposed. In this work, they presented a Particle Swarm Optimisation (PSO) algorithm as the central agent for the model. This model attempted the regulation and control of temperature, illumination and CO_2 concentrations in an office building scenario. They sought to create a system which could maintain indoor comfort parameters in adverse situations such as energy shortage, though evaluated with synthetic data. Even though multi-agent models were claimed to be ale to improve accuracy and robustness, sensors were still showing limitations and data nature still represented one of the main pitfalls to overcome for these models.

2.2.4 Online Modelling Approaches Of OM

Alongside the accessibility of internet, faster networks and data ubiquity, researchers have developed models for detecting real-time occupant estimation. These models are based on making predictions in online scenarios when not all the data is available (only current time data), meaning that data is feed progressively into the model as the time advances.

One of these novel approaches is the work in [14], where they presented a HSMM based algorithm, arguing they could improve HMM based ones, since HSMMs include the advantages of Markov models, but enhanced with a predetermined duration time for different states. This allegedly added more flexibility when modelling transitions between states (HMMs transition time distributions are inherent exponential, while HSMMs transitions can be modelled as for instance normal distributions). In [20], the authors extended their experiments including novel techniques such as SVM and ANN. The aim of this work was to establish baselines to compare algorithm performances and which were the advantages and drawbacks of each method. From this work, they concluded suggesting that other considerations related to the nature and pre-processing of the data could have a significant impact on the final performance of the model. These works also used a combination of sensors, claiming getting better results than using just one type of sensor (potential savings of over 30% energy if implementing the right occupancy profiles).

Later, in the work in [42], the authors presented a model based on a combination of wireless sensor devices including PIR and contact door sensors. In this work, they identified some of the technical issues present when working with Wireless Sensor Networks (WSN), such as battery lifetime or the need for looking into more permanent solutions; with the objective of preventing these issues to tamper with the models' potential results. Here, the potential savings were estimated aroung 15% but compared to other modelling approaches and using low-cost off the shelf sensors.

Although many researchers were claiming that a combination of various sensors may allow the development of more accurate models [43][44][45], the authors in [46] proposed an algorithm for presence detection that, after being tested with multiple sensors combined, it showed better results when only PIR data was used. This suggests that the use of one single type of sensor might be advisable when facing deployment constraints,

as well as it points towards the idea that some sensors encode more quality information than others in this field. This work claimed to get high levels of accuracy over 90%, but it was limited to the presence detection of a single occupant only when sitting at his/her desk.

One of the few occupancy models addressing a domestic scenario was the work in [44], where they used motion and temperature sensors to detect presence and regulate heating valves for each room. However, their algorithm estimations were based mainly on RFID data and the ambient sensor data was used to contrast and relate occupation to temperature values. The research in [47] also used RFID sensors in combination with a k-nearest neighbour (kNN) algorithm for occupancy estimation and tracking. However, as discussed above, these sensors are not usually used in OM for intrusiveness and ubiquity issues.

In the work in [45], the authors proposed a model in which indoor parameters and energy consumption data was introduced as inputs into a model based on Neural Networks and Fuzzy Logic for number of occupants estimation. Data from temperature sensors was used to study appliances usage, and server watchdogs were used to monitor PC connections. Coupled with outdoors data, this work claimed to improve energy efficiency and occupant comfort, though little information on the actual performance of the model was reported.

In [43], a rich-sensor device was also introduced including temperature, humidity, motion, sound, light, CO_2 and door state sensor. The model was able to establish the number of occupants in communal rooms. This work identified CO_2 levels as the main indicator among the rest of the inputs. They compared different machine learning models (linear regression, perceptron and SVM) to perform occupant detection; claiming performances over 90% for some classifiers, with a sampling time resolution of 15 minutes.

Lately, other works were proposed to perform similar detections such as in [48] and [49], both performing occupant number estimation up to 8 occupants. These contributions represent some of the latest achievements in occupancy models for Online detection and have similar characteristics: both built their algorithms upon ANN models (in [48] the model was built upon Time Delayed (TDNN), while in [49] opted for radial basis function neural networks (RBFNN)), and also identifying specific sensor inputs as having more

impact than others for the occupancy estimation. Another interesting contribution from these works was the study of the increasing error rates when attempting to forecast much time in the future. The main limitation is that models based only on one type of sensor, in this case CO_2 are prone to errors specific to those sensors and their use in OBPM models [50].

TABLE 2.1: OM approaches. For multi-agent models, we state the algorithms used by the learning agents. Sensors legend: *Mo: Motion, Co: Contact, C2: CO_2, Te: Temperature, Hu: Humidity, Li: Light, So: Sound, RF: RFID, MeT: Meters*

Models		Task	Sensors	Research Active Period	Refs.
ANN	FuzzyLogicNN, RBFNN, TDNN	Presence Detection, Number Estimation	Mo, Te, Li, C2,Hu, Li, So	2010-2013	[20][51][48] [49][52][53]
Markov Models	HMM, HSMM	Energy Performance, Number Estimation, Presence Detection, Tracking	Mo, Te, Li, C2, Hu, Li, So	2009-2012	[40][14][20] [10] [36][38] [54]
Stochastic	Pareto, Belief Network	Presence Detection	Mo, Co	2005-2006	[33],[34]
Multi Agent	PSO Learner, Markov-Based Learners	Energy Performance	MeT, Te, Li, C2	2012-2013	[40],[41]
SVM	Multi-class, RFB	Number Estimation	Mo, Te, Li, C2,Hu, Li, So	2010-2012	[20],[43][12] [43][53]
Heuristic and GP	Knowledge-Driven Classification	Presence Detection	Mo, Co	2010	[42][32][37] [7][46]
SVM	Multi-Class, RBF	Number Estimation, Presence Detection	Mo, C2, Te, Hu, Li, So, Co	2012-2013	[43][53]
Regression	ARMA Linear, Logistic	Number Estimation, Presence Detection	Mo, C2, Te, Hu, Li, So, Co	2012-2013	[43][53]
Perceptron	Multi-Layer	Number Estimation	Mo, C2, Te, Hu, Li, So, Co	2012	[43][53]
Multi Agent	SVM, Regression, Perceptron	Number Estimation, Presence Detection	Mo, C2, Te, Hu, Li, So, Co	2012	[43][39]

Other works attempted to make predictions of occupancy states at some point in the future. The work in [54], sought prediction in energy consumption with 24 hours in advance, showing the unacceptable rate of error (model accuracy lower than 50%) when forecasting with such large period of time in advance. This suggested more reasonable time spans were advisable. The data collected from the energy meters was used to extract occupancy patterns in order to create usage profiles.

The previously mentioned work in [43], also was deigned to perform future estimation

of numbers of occupants. Results for forecasting were obviously significantly less accurate compared to those for normal Online detection, and showed how difficult this task becomes when expanding the prediction line into the future. In this work, the data was discretised in timeslices of 15 minutes, but performance error was only manageable within the following timestep (15 minutes in advance), achieving accuracies between 67% and 75%.

Lately, the research conducted in [53] used a Bayesian combined forecasting approach evaluating several different models performance such as TDNN, SVM and Regression. Using a setting of 10 minutes time model resolution, they tried to predict presence between 10 minutes and 2 hours into the future. However, experimental results suggested predictions over 6 timesteps in advance (60 minutes) yielded unacceptable error rates. As occurred with other similar contributions, these proposals showed potential for occupancy forecasting, but always within certain time limits due to the inherent error introduced when making future state predictions.

Occupancy models are summarised in Table 2.1.

2.2.5 OM Considerations

There has been a clear timeline of progression in which static models combined with 'live' sensor techniques enabled real-time detection and further occupancy forecasting. Many works suggested that data from simple binary sensors such as motion have the potential to be used to estimate occupancy by itself under certain circumstances [10][34][46][32]. However, recent work showed that by using combinations of low-cost sensors, occupancy states prediction can be further boosted and used to help in the control of building systems [20][43][55][56].

In spite of recent advances in the Internet Of Things (IoT) and data fusion [57], these approaches are not always capable of modelling the complexities of the relationships with the occupancy states and the correspondent sensor signals. While sensor signals can carry important information by themselves, this sensor information suffer from limitations in terms of latencies, false positives/negatives, noise, physical layout limitations and so on [50][58][59]. To improve the predictions that can be made from sensor data,

it is necessary to develop models that can incorporate the data and overcome the mentioned issues. This together with the potential for different applications in the smart buildings context, are the main motivations for the creation of OM models and why this is an important research topic nowadays [13].

Recent occupancy models are pushing to improve accuracy, mostly in online scenarios where the streams of data are constantly changing and in predictive tasks where the error rates increase as the window for prediction becomes longer. Due to this, these models cannot give accurate predictions when attempting to do so using a long time window. Also, in order to understand the real differences between model results, more benchmarks and comparable frameworks are needed where the impact of different datasets, sensors and deployments can be assessed alongside the actual model performance. Additionally, more datasets need to be made publicly available in order to use them for model comparison and also to prevent the lack of data being a constraint for the emergence of new proposals. Despite some data has been made available and the existence of sensor test-beds, little current public datasets include relevant occupancy data including typical occupancy sensors such as temperature or CO_2 labelled with number of occupants present together in one space.

2.3 Activity Modelling Approaches

2.3.1 Activity Modelling Definition

Activity recognition models, are primarily focused on domestic scenarios and their main task is usually ADL classification. This section covers existing approaches related to modelling high-level activities [60] (i.e. sleeping or eating) based on non-intrusive sensor data. User activities will be estimated from sequences of sensor data and the models will attempt to identify patterns in order to infer which activity is being conducted. Normally for these activity models intrusiveness does not pose such a problem as happened with public buildings, since homes are inherently private spaces and the information would not be usually disclosed to third parties except in cases where there is an explicit agreement by the user in doing so (healthcare monitoring, security, etc.).

The main applications for activity models are digital houses (automatic media and smart appliances) designed to improve the user experience, healthcare monitoring systems, security and finally energy management and reduction. Due to this, we can expect for these scenarios to incorporate a higher sensor density and diversity (the latter depending of the complexity of the activities intended to model). Beyond ambient sensors, AM models require sensors attached to TV and media devices, appliances, communications or to the users themselves like RFID or smart phones data.

2.3.2 Offline Modelling Approaches Of AM

One of the pioneering works was proposed in [17] where the authors used data from RFID and wireless sensors (temperature, light and PIR) and contacts to detect phone calls. By means of fuzzy sets to integrate them, they used HMM to run the model limited to 3 scenarios in which lights, music and TV were regulated.

In [61], they also used Markov models but with datasets from Georgia Aware Project [62] and MIT House [26], which consisted of data from pressure mats made of piezo sensors to detect movement patterns at home (user tracking). These pressure mats provided the paths users were following, and the Markov models were used to model the likelihoods of any of those possible paths.

More recently, the authors in [63] tried to overcome one of the major challenges in AM modelling: multi-occupancy. They intended to tackle this problem by describing collaborative activities, which were as the typical ADL but assuming two different users. Activities were defined respectively for each occupant, 4 of them being cooperative leaving 7 activities as individual ADLs. They used the WSU CASAS Datasets [64], which included motion, temperature and analogue sensors for water and stove usage.

The works in [65][66][21] made a significant contribution to the field since they made public their datasets and presented benchmarks for different model performances including HMM, HSMM or CRF. These datasets included sensor readings from PIR for motion, contact for doors and drawers and a float sensor for the flush. A number of other researchers used these datasets as in [12][2][67][51] or [68]. Based on the combined experimental findings, it was noted how contextual factors can have a huge impact on

the model performance, which is crucial to understand which model is better suited for determined scenarios and datasets.

The work in [2], presented a model based on active learning techniques. Their objective was to find patterns within the ADL sequence in order to reduce the data necessary to define an activity; hence a reduction in labelling efforts. This is a crucial aspect to address since one of the main constraints for occupancy and activities is the scarcity of labelled data.

Active Learning techniques were also proposed in [67], where they used clustering techniques and unsupervised learning claiming obtaining as much accuracy as those ones using other traditional approaches; thus avoiding the need for large amounts of labelled data. Moreover, their model was able to address overlapped activities. Overlapping and concurrency are among the major challenges for activity models due to the difficulty to distinguish between two activities happening at the same place and time, or when activities are momentarily abandoned to perform any other task in the meantime.

In [57], they also tried to address these issues by means of an algorithm based on web ontology language (WOL). They claimed that, since it is particularly difficult to mathematically define a differentiation of chain sequences when occurring at the same time, WOL and knowledge-driven techniques are particularly useful to overcome this limitation. Another promising work addressing multi-occupants in activity models was the one in [68] where ontological knowledge-driven methods were used too, although limited to two occupants.

Lately, authors have been designing models with more recent machine leaning approaches found in activity modelling literature. While in [69] they proposed a model based on back propagation ANNs, in [70] they opted for the inclusion of GP algorithms in order to find the optimum weight value for an ensemble model including various ML techniques; both using the datasets from WSU CASAS [64].

It is interesting to note that, despite the increase in popularity of novel approaches such as ANN or SVM models, traditional approaches such as HMM or HSMM have been among the most popular approaches found in existing contributions: e.g the works in [17][61][63][65][54][67] used Markov models due to their ability to model state and sensor sequences.

2.3.3 Online Modelling Approaches Of AM

As occurred with occupancy models, online activity models can also be found in current literature. The approach is also similar and involves the detection of activities using streaming data and in occasions incremental model update.

In the work in [71], the authors introduced an ANN-based for abnormal behaviour detection for healthcare monitoring. However, their online approach was the development of a more effective way of training a neural network in order to allow re-training between inferences. This approach did not however attempted to update parameters for incremental learning and just introduced a more efficient way of model retraining.

A similar approach was followed in [72], where a model to fast and effectively retrain support vector machines was proposed. It consisted in using the first model training parametrisation to find the support vector and use only them for model retraining. Although this paper showed good learning rates and inferences, model parameters were never incrementally updated.

In [51], a finite state automata was used to remove sensor redundancies and therefore reduce the amount of data to train and test the model. The authors claimed this was an efficient way to allow online detection but experiments validating this approach were not conducted.

Another recent activity model is the one in [73], where they used a learning automata and fuzzy temporal windows. Although this method was only intended to detect abnormal behaviour vs. normal behaviour (not ALDs) they were able to perform incremental online learning for the learning of future patterns in the data.

One interesting approach using window segmentations was presented in [74] where they shown methods to create variable windows with sensor events that could be used to improve activity model performance. This approach was based on the idea of incorporating mutual sensor information, time inputs and previous window information to maximise the amount of information needed to make a reliable activity prediction.

The time window approach was also used in the work [75], where they presented a knowledge-driven approach based on ontologies claiming that this model can be adaptive

to future changes in data and data patterns obtaining high levels of accuracy using synthetic generated data.

Although activity models literature is extensive, it is not easy to find incremental online learning models within this particular field. Some methods use the idea of improving the efficiency of the training phase (as described above) in order to make the model able to be retrained in the time between the incorporation of a new sample. Models that pursue the incremental updating of parameters to improve efficiency and prevent gradual model complexity increase and unlimited data storage are still scarce for this particular modelling approach.

A summary of existing AM models is shown in table 2.2.

TABLE 2.2: Activity modelling. The ADL number in brackets represents the number of different model states or activities performed. Sensors legend: *Mo: Motion, Co: Contact, Te: Temperature, Li: Light, RF: RFID*

Models		Activities	Sensors	Research Active Period	Refs.
Markov Models	HMM HSMM, CRF SCRF	ADL(16act), Multimedia & Luminance Control	RF, Te, Li, Mo, Co	2014-2016	[17][61][63] [65][67][66] [21][2][12] [72][74]
ANN	ANN	ADL(8act)	Simulated (Co, Mo)	2008	[71][69]
Knowledge Driven	Ontologic Knowledge-Driven	ADL(8act) Concurrent	Mo, Co	2014	[75][57][68]
Finite, (fuzzy)Learning Automata	Finite State Machine	ADL(16act)	Mo + Multi	2013	[51][73][70]

2.3.4 AM Considerations

Despite recent advances, activity models need to overcome specific issues. Multi-occupant scenarios where concurrency and interleaving of activities happen together with the possibility of having more than one user performing ADLs at a time, seem to be the major challenges these models faced based on the woks surveyed. Additionally, the datasets and sensors used, the way the data is pre-processed and the methodologies chosen for model validations, have a large impact on the final results and performance reported.

2.4 Discussion

After thoroughly reviewing the existing approaches for OBPM, we identified a series of critical aspects that need to be investigated in more depth.

2.4.1 Main Achievements

Machine learning approaches combined with sensor data have shown potential for modelling occupant behaviour patterns both for occupancy and activity models. Table 2.3 shows popular techniques that appear in literature frequently. For example, Markov-based models including HMM or HSMM are the most used so far, however other methods such as ANN and SVM have recently shown increasing popularity. These approaches provide better ways to potentially manage energy consumption in buildings by incorporating occupant comfort and behaviour parameters into the final decision making process. This means that these models can use the sensor information to adaptively manage building systems in order to improve efficiency while maintaining user comfort. Moreover, the rapid development of computing technology and its ubiquity also contributed to the quick growth of new techniques in the field.

TABLE 2.3: Existing machine learning modelling approaches for OBPM.

Modelling Approach	Task	Works
Markov-based Models	Presence Detection Number Estimation Activity Recognition	[10],[36],[38],[40], [14],[20],[54],[17], [61],[63],[66],[67], [21],[2].
ANN	Presence Detection Number Estimation Activity Recognition	[20],[51],[48],[49], [45],[47],[69].
SVM	Presence Detection Number Estimation Activity Recognition	[20],[43],[53].
Other Machine Learning	Presence Detection Tracking Number Estimation Activity Recognition	[47],[43],[53],[67], [46],[66].
Multi-Agents	Energy Performance Presence Detection Number Estimation	[39],[40],[48],[43]
Genetic Programming	Presence Detection Activity Recognition	[37],[70]
Knowledge-driven	Presence Detection Activity Recognition	[42],[44],[57],[68]

Access to publicly available datasets and ubiquity of sensor have largely contributed to the success of these models and their increasing popularity among many research groups. However, there is still the need for more comprehensive and bigger datasets, specially

including occupancy data from diverse scenarios or using multiple different sensors. In fact, the number of works using public data is over 60% in activity models, while drops to around 15% of the works surveyed related to occupancy models.

Furthermore, although there is no restrictions on this regard, we have noted that models tend to fall naturally within either OM or AM types. Every OBPM previous work follows one of these approaches following the selection criteria for scenario, sensor and classification tasks.

2.4.2 Main Limitations

In spite of encouraging work, current state-of-the-art approaches suffer from limitations including:

Data Nature: Data quality is one of the main issues that affects model performance of occupant behaviour modelling, and it is determined by the quality of the sensing devices, the topology adopted (e.g. wireless, cable, sensor networks), transmission rate (resolution), noise, sensors layout or the physical characteristics of the building among others. More work is needed to study which specific sensors provide better occupant information in buildings, and which are the best practices for sensor selection and physical network deployment.

Despite encouraging work [13][76], it is not an easy task to say which sensors are better suited depending on the modelling approach. For example, PIR sensors are the most common in occupancy and activity models. However, as some research pointed the convenience of using sensor combinations and other sensors started to be more ubiquitous, different sensing options become available (i.e. temperature, humidity, sound and air quality). Due to all these considerations, sensors will have to be carefully considered to ensure they will be able to capture the relevant real-world information to fit the needs of the model to be designed. Some works have addressed this issue [50][58][59], but these are scarce and more reliable comparisons need to be made. For example, in the work in [50] they identified specific limitations in the current literature and the use of diverse sensors (e.g. PIR have many false positives while CO_2 has better accuracy but low responsive times) or on the work in [58] where they claim sensors need to be calibrated for each potential use.

Sensor selection is one of the most important challenge in the field as many models used different approaches and some of them seem to have its own importance depending the application the model is built for. This challenge increases with the fact that in our context we only want to consider non-intrusive sensors, which naturally offer 'less' richer information compared with a video-camera or some data collected from smartphones or wearables (for example, it is challenging to track a person's movements using PIR, temperature or CO_2 sensors while it is easier using the GPS coordinates from her phone).

Data Pre-Processing: Based on the works surveyed, many authors noted that the way data is pre-processed can also have a large impact on results [65][21][64][77]. Sensors' signals require a transformation to make them readable for the model through pre-processing stages. In these steps, data is converted into variables after being re-arranged and discretised. These are critical steps as pointed by previous works that showed how different system granularities can have an impact over the model performance [77]: e.g. models making estimations with a frequency of 60 seconds, will be in much more disadvantage that models performing a 30 minutes window prediction, since obviously the information collected from sensors in 30 minutes will have much more data than information collected each minute and more time to process it. In addition, model resolution will by constrained by the specific sensor/network transmission rate, data nature and the model operational time.

OnLine settings: Online models need to be able to handle streaming data. This represents a two-fold challenge:

- Models need to make predictions without a batch of data, therefore not knowing how data is going to look like posteriorly, and
- It is not unusual in online settings that models will be required to update their parameters incrementally as new data is continuously fed.

With the appropriate design, complex algorithms can also be used for online learning such as SVM [78][43], ANN [79]. However, when the time models have to perform certain operations is limited, complex models can be too slow to be used 'on the fly'. Furthermore, since these models parametrisations are commonly based on gradient descent algorithms and complex optimisation functions it has been noted that their incremental

update can be even more complex and slow than retraining the model using all the data again. Contrarily, Markov models have nice properties for parameter update as well as a more clear representation of the data relationships due to their generative model properties. However, these updates are limited when models need to process partial streamed data.

Finally, from our literature review analysis we can extract some interesting ideas:

- Many techniques are available and some existing works claim being highly accurate, but more effort to assess and compare those results is needed in order to evaluate what are the best methods.
- OBPM recent datasets are increasingly heterogeneous and variate, based on multiple sensors, deployments and data acquisition and pre-processing techniques. Therefore, there is a need to investigate how models can handle different data and what role data nature holds in the final model performance.
- Models that address online scenarios are not perfect and in some cases they need to be more accurate and efficient. There is the need therefore of developing approaches that take into account the diverse challenges and limitations OBPM models face in order to find solutions more robust and adaptable.

Chapter 3

Benchmark Experiments Based On Existing Approaches

This chapter presents a benchmark of experiments where we have conducted a thorough comparison and evaluation using well-known machine learning techniques for OBPM based on publicly available real datasets using metrics of performance evaluation such as accuracy or precision. Additionally, in order to better understand which are the factors that might have an impact over the overall model performance, we have conducted supplementary experiments where we have investigated and identified other aspects and limitations that could affect the accuracy of the models as well as tried different methods in relation to data preprocessing approaches.

The following sections provide more information about the experimental settings in terms of the used datasets, methodologies and ways of conducting performance evaluation. Regarding the datasets needed to train and validate the models, we firstly introduce different publicly available datasets, from which we have selected two of the most relevant to conduct our experiments. We define the methodologies we have selected for our experimental evaluations and we discuss the results obtained.

3.1 Motivation

One of the critical conclusions drawn from our literature review, is that models are not perfect and more work needs to be done to understand how these occupant patterns in

buildings can be more accurately modelled [80]. Analysis of current works suggested that it is crucial to investigate not only model characteristics, but also data nature aspects, and their impact over the model performance [38][9]. To investigate how models behave under similar conditions, we have conducted a benchmark evaluation using some of the most prominent machine learning techniques found in literature. Additionally, in order to find the more robust approaches for OBPM modelling, we have evaluated model performance with different datasets containing data from a variety of sensors and collected by different research groups. To further prove model robustness and generalisation properties, we have also conducted supplementary experiments using different data pre-processing techniques consisting of the timeslice approach using different time resolutions and a comparison with a chunk data approach [77]. The decision of choosing these datasets has been made under the following considerations:

- These two datasets have been previously used successfully in other researches, therefore are representative and relevant datasets in our context[21][81].
- Previous works [65][66][21][64][74], using these datasets have provided with thorough model evaluations using different ML and preprocessing techniques. This enables us to have a framework of evaluation where we can compare the results obtained through our own experiments.
- These datasets include a variety of sensors and pre-processing approaches, therefore allowing us to additionally study how contextual factors have an impact over the model performance.

3.2 Publicly Available Datasets

Although this trend is changing, there are not many publicly available datasets for OPBM. However, having access to this data is critical for two main reasons: a) It enables other researchers to design and evaluable their proposals using data from real world scenarios without having to set their own collection system up; b) Having data from different groups and scenarios is critical to evaluate model generalisation and to create benchmarks of performance for existing approaches.

Some examples of publicly available datasets include:

- UCI Machine learning repository (http://archive.ics.uci.edu/ml/) is one of the most popular sources for data that can be applied to machine learning related project including sensor data.
- CRAWDAD (http://crawdad.org/purpose-Human-Behavior-Modeling.html) Which stands for A Community Resource for Archiving Wireless Data, has an specific *Human Behaviour Modelling* section, which includes several datasets including mobility data, data from smart phones or location and proximity data.
- Finally, other important source is the Boxlab (https://boxlab.wikispaces.com) repository, which contains a large number of home datasets. This depository is only focused on "instrumented living environments ". Projects that have uploaded their data here are: Aware Home at Georgia Tech [62], CARE [82], PlaceLab [83] among others.

Apart from those larger datasets, there are single datasets collected for groups for example: CASAS [64], DomusLab [84], Aras [1] or Tim Van Kasteren's [21]. These datasets were collected from different ambient sensors, scenarios and teams, so they are specially useful to test models' adaptiveness, and can be used as benchmarks to compare the different modelling approaches.

In order to perform our benchmark experiments, we selected WSU CASAS [64] and Tim Van Kasteren's [21]. These datasets contain sensor inputs and human activities labels, so they can be used to create OBPM approaches. These two datasets contain different sensors and also have already been used for model performance comparison, which make the ideal for the purposes of our experiments. More information about these datasets is given in the following sections.

3.2.1 Experimental Datasets

For our experiments, two datasets have been selected (D1 and D2) from previous human behaviour detection studies: D1 was published by [21] and was originally used for ADL recognition purposes, and D2 was made publicly available by WSU CASAS [64] and also used for ADL recognition. These two datasets have been selected for our experiments in this work because, in addition to making public their datasets, both projects provided their own baseline recognition performance experiments. These are extremely helpful

since, by using the same data, the performance of the models proposed in our project can be compared and evaluated under similar conditions. Other important consideration for the choice of these datasets was their substantial length (D1 contains over 42k sensor readings and more that 800 activity occurrences and D2 has over 600k readings including 600 activity annotations). Finally, the variety in sensor nature found in the two datasets (D1 is based on 5 various types of sensors while D2 is mainly based on just motion sensors) will also be useful to understand variations in performance when models are fed with data from different sensor nature.

3.2.1.1 Dataset 1 (D1)

D1 data was collected from three scenarios: House A, House B and House C. D1 data collected uninterruptedly and divided in days. The types of sensors found here are motion passive infra-red (PIR), reed switches, pressure mats, mercury contact sensors and float sensors. For example, House A has a total of 14 sensors and the labels were annotated by the occupants using a Bluetooth voice detector device rather than other scenarios in which the activities were just written down on paper. Further information can also be found in: https://sites.google.com/site/tim0306/datasets.

Data annotation was made using two sets of annotations. The first set refers to sensor events which were recorded using 4 columns stating the start time, the ending time, the sensor ID and the value of the sensor. Note that, as all the sensors included in this dataset are binary, the state is always 1 (ON) since it only reflects the time the sensor has been active. The second set also indicates starting and ending time but for the activity performed in that interval, therefore indicating the ID for the activity being performed. The numbers of activities go from 10 in House A, 13 in House B and 16 in House C. Fig. 3.1 contains an example showing how this dataset is annotated. In Fig 3.2 (Source: https://sites.google.com/site/tim0306/datasets) we can see a detail of the floor distribution of the scenarios and in Tables 3.1-3.4 there are the general details of this dataset.

Start time	End time	ID	Val
19-Nov-2008 22:47:46	19-Nov-2008 22:49:17	28	1
19-Nov-2008 22:49:21	19-Nov-2008 22:49:22	2	1
19-Nov-2008 22:49:24	19-Nov-2008 22:50:14	6	1
19-Nov-2008 22:50:19	20-Nov-2008 11:14:11	28	1

and activities:

Start time	End time	ID
19-Nov-2008 22:49:00	19-Nov-2008 22:50:00	1
19-Nov-2008 22:50:40	19-Nov-2008 22:51:45	4
19-Nov-2008 22:59:25	19-Nov-2008 23:00:00	17
19-Nov-2008 23:00:50	20-Nov-2008 01:25:00	28

FIGURE 3.1: Example of annotation for Dataset 1. The first rows are the sensor annotations and the latter ones are the activities labelling.

TABLE 3.1: D1 General Information.

	HOUSE A	**HOUSE B**	**HOUSE C**
Age	26	28	57
Gender	Male	Male	Male
Setting	Apartment	Apartment	House
Rooms	3	2	6
Duration	25 days	14 days	19 days
Sensors	14	23	21
Activities (Including Idle)	10	13	16
Annotation	Bluetooth	Diary	Bluetooth

3.2.1.2 Dataset 2 (D2)

This dataset was collected using 27 motion sensors deployed in a domestic scenario through 56 days, though this dataset was not labelled continuously. Unlike D1, this dataset contains only motion sensors events. Nonetheless, as we will discuss in future sections and proven before by many researchers [10][33][34][85] and many other models reviewed [57], motion sensor data proves to have the potential to provide information enough to accurately perform activity recognition under certain considerations.

This dataset has only one set on annotations (including sensor readings and labels), which contains 7 columns stating the time, the sensor ID and the value of the sensor as in D1, which in this case can be either 'ON' or 'OFF'. In addition to the sensor event, in the same line we can also find when an activity starts (denoted by the activity label plus 'begin') or ends (denoted by 'end'). This dataset features a total of 10 activities. For

(a) House A

(b) House B

(c) House C, First floor

(d) House C, Second floor

FIGURE 3.2: Floorplan D1, House A, House B and House C. The red rectangles represent the sensor nodes.

further information visit: http://ailab.wsu.edu/casas/datasets/. An example showing the annotations in this dataset can be seen in Fig. 3.3. In Fig 3.4 there is a floor map for this dataset and other details can be seen in Table 3.1 and 3.6.

3.3 Machine Learning Models

Here, we briefly discuss the models that we have used in this chapter experiments. We have chosen some of the most popular approaches found in previous literature combining generative and discriminative models, and probabilistic and non-probabilistic techniques. The models proposed are:

Start time	Sensor ID	Value	Activity	Status
2009-06-10 03:49:24.007006	M009	ON		
2009-06-10 03:49:25.013601	M005	ON		
2009-06-10 03:49:28.063972	M009	OFF		
2009-06-10 03:49:29.073763	M005	OFF	Bed to toilet	end
2009-06-10 03:54:23.058206	M002	ON	Night wandering	begin
2009-06-10 03:54:26.044383	M006	ON		
2009-06-10 03:54:28.002777	M002	OFF		

FIGURE 3.3: Example of annotation for Dataset 2.

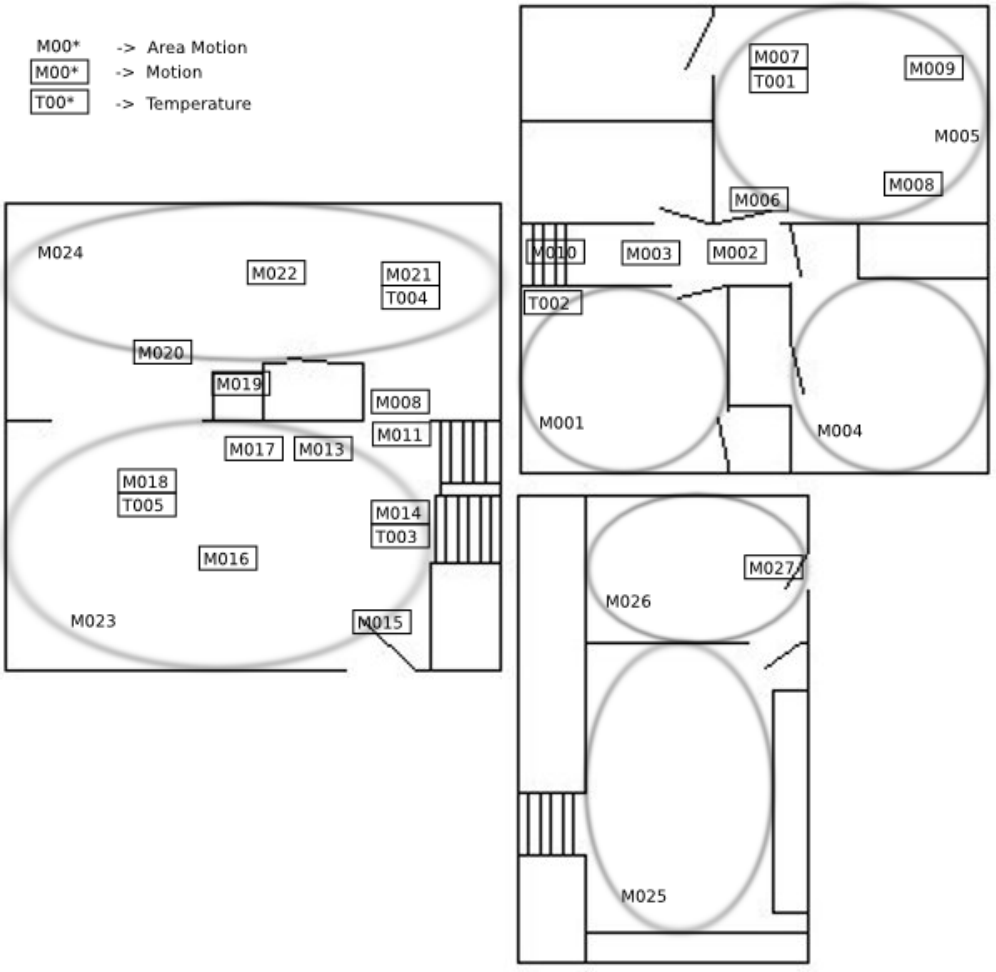


FIGURE 3.4: Floorplan for D2 divided in three areas.

TABLE 3.2: Features and labels for *D1, House A*.

Sensors:			
	NUMBER	ID	NAME
	1	1	'Microwave'
	2	5	'Hall-Toilet door'
	3	6	'Hall-Bathroom door'
	4	7	'Cups cupboard'
	5	8	'Fridge'
	6	9	'Plates cupboard'
	7	12	'Frontdoor'
	8	13	'Dishwasher'
	9	14	'ToiletFlush'
	10	17	'Freezer'
	11	18	'Pans Cupboard'
	12	20	'Washingmachine'
	13	23	'Groceries Cupboard'
	14	24	'Hall-Bedroom door'
Activities:			
	1	-	'idle'
	2	1	'leave house'
	3	4	'use toilet'
	4	5	'take shower'
	5	6	'brush teeth'
	6	10	'go to bed'
	7	13	'prepare Breakfast'
	8	15	'prepare Dinner'
	9	16	'get snack'
	10	17	'get drink'

- HMM
- HSMM
- SVM (linear and non-linear)
- kNN

These models have been selected because: HMM and HSMM are arguably the most popular approaches for OBPM models and have been extensively used for pattern recognition systems specially when using temporal data sequences; SVM are one of the most representative approaches of a new generation of models that use the capabilities of modern machines and recent developments in gradient functions or kernel functions to achieve highest levels of performance; and finally kNN models are here included to show how

Table 3.3: Features and labels for *D1, House B.*

Sensors:			
	NUMBER	ID	NAME
	1	1	'toilet door'
	2	3	'fridge'
	3	5	'cupboard groceries'
	4	6	'toilet flush'
	5	7	'frontdoor'
	6	9	'cupboard plates'
	7	10	'Bedroom door'
	8	12	'pressure mat bed right'
	9	13	'pressure mat bed left'
	10	14	'mercury switch cutlary drawer'
	11	15	'mercurary switch stove lid'
	12	16	'PIR bedroom'
	13	18	'mercury switch dresser door'
	14	19	'PIR bathroom'
	15	20	'pressure mat piano stool'
	16	21	'gootsteen float'
	17	22	'pressure mat chair study'
	18	24	'balcony door'
	19	25	'window'
	20	26	'toaster'
	21	27	'microwave'
	22	28	'PIR kitchen'
	23*	N/A	N/A
Activities:			
	1	-	'idle'
	2	1	'Leaving the house'
	3	4	'Use toilet'
	4	5	'Take shower'
	5	6	'Brush teeth'
	6	10	'Go to bed'
	7	11	'Get dressed'
	8	13	'Prepare brunch'
	9	15	'Prepare dinner'
	10	17	'Get a drink'
	11	24	'Wash dishes'
	12	31	'Eat dinner'
	13	32	'Eat brunch'

sometimes simpler approaches can be used for OBPM modelling with a good compromise between model performance versus complexity.

TABLE 3.4: Features and labels for *D1, House C*.

Sensors:			
	NUMBER	ID	NAME
	1	5	'bed right, pressure mat'
	2	6	'couch, pressure mat'
	3	7	'freezer, reed'
	4	8	'toilet flush upstairs, flush'
	5	10	'toilet flush downstairs. flush'
	6	15	'drawer with keys to backdoor'
	7	16	'bathroom swingdoor left'
	8	18	'cutlery drawer, mercury switch'
	9	20	'cupboard pots and pans, reed'
	10	21	'microwave, reed'
	11	22	'cupboard storage bins, reed'
	12	23	'cupboard herbs and plates,reed'
	13	25	'toilet door downstairs'
	14	27	'cupboard bowl and cups'
	15	28	'frontdoor, reed'
	16	29	'bedroom door'
	17	30	'fridge, reed'
	18	35	'bathtub, pir'
	19	36	'dresser, pir'
	20	38	'sink upstairs, flush'
	21	39	'pressure mat bed left'
Activities:			
	1	-	'idle'
	2	1	'leave house'
	3	3	'Eating'
	4	4	'use toilet downstairs'
	5	5	'take shower'
	6	6	'brush teeth'
	7	7	'use toilet upstairs'
	8	9	'shave'
	9	10	'go to bed'
	10	11	'get dressed'
	11	12	'take medication'
	12	13	'prepare Breakfast'
	13	14	'prepare Lunch'
	14	15	'prepare Dinner'
	15	16	'get snack'
	16	17	'get drink'

3.3.1 Hidden Markov Models

These models are based on Markov sequences, which assume interdependence between previous occurrences of classes (one previous step). This algorithm calculates the probability of a hidden state for consecutive timeslices (first order Markov chain) combined

TABLE 3.5: D2 General Information.

	HOUSE A
Residents	2+pet
Gender	Male and Female
Setting	House (3 areas)
Rooms	4
Duration	56 days (intermittent)
Sensors	27
Activities (Not Including Idle)	10
Annotation	Questionnaire

with the likelihood of a state for a given sensor observation (see Fig. 3.5). The observable states are the sensor readings. The hidden states that trigger those sensors are the activities. The probability of each state depends on the observations X and the transition probability. The observations (sensor readings) depend on the hidden variable, and the hidden variable or state depends on the previous hidden state (first Markov assumption). These models are hugely popular in literature due to their ability to model occupancy states as sequences of observations (sensors) and hidden variables (states). Formally factorises as

$$P(Y, X) = \prod_{t=1}^{N} p(x_t|y_t)p(y_t|y_{t-1}) \tag{3.1}$$

Where the joint probability $P(Y, X)$ can be obtained by multiplying $p(x_t|y_t)$ (observation probability from sensor readings) to the state transition probability $p(y_t|y_{t-1})$ for each timeslice t. Transition probability means how likely is an activity to occur by knowing the previous one. Note that to determine the transition probability in time $t = 1$ instead of the transition (we assume there is no $t = 0$) we use the prior probability $p(y_t)$ [15].

3.3.2 Hidden Semi-Markov Models

When dealing with activities or states that relate directly with a determined duration (e.g. we can safely assume that a *shower* activity takes around 5 minutes while a night sleep would be between 6-8 hours), it is important that the model can also describe that kind of duration behaviour. HSMM also utilise the first Markov assumption to calculate the current state based on the previous state as HMM, but also including a duration parameter which model state duration using any probability distribution. Modelling duration represents that when entering into a certain state, the model will compute a

TABLE 3.6: Features and labels for *D2*.

Sensors:		Activities:	
NUMBER	ID	NUMBER	ID
1	M001	1	Bed_to_toilet
2	M002	2	Breakfast
3	M003	3	Bed
4	M004	4	C_work
5	M005	5	Dinner
6	M006	6	Laundry
7	M007	7	Leave_home
8	M008	8	Lunch
9	M009	9	Night_wandering
10	M010	10	R_medicine
11	M011		
12	M012		
13	M013		
14	M014		
15	M015		
16	M016		
17	M017		
18	M018		
19	M019		
20	M020		
21	M021		
22	M022		
23	M023		
24	M024		
25	M025		
26	M026		
27	M027		

likely duration in which the state will not change to next one. The equation

$$P(Y, X) = \tag{3.2}$$

$$\prod_{t=1}^{N} p(x_t|y_t)p(y_t|y_{t-1}, d_{t-1})p(d_t|yt, d_{t-1}) \tag{3.3}$$

shows that in this occasion, we can use the HMM approach as an starting point but including state duration $p(d_t)$. State duration probability is the time the current activity will take and it is determined by the actual current state and the prior state duration [16].

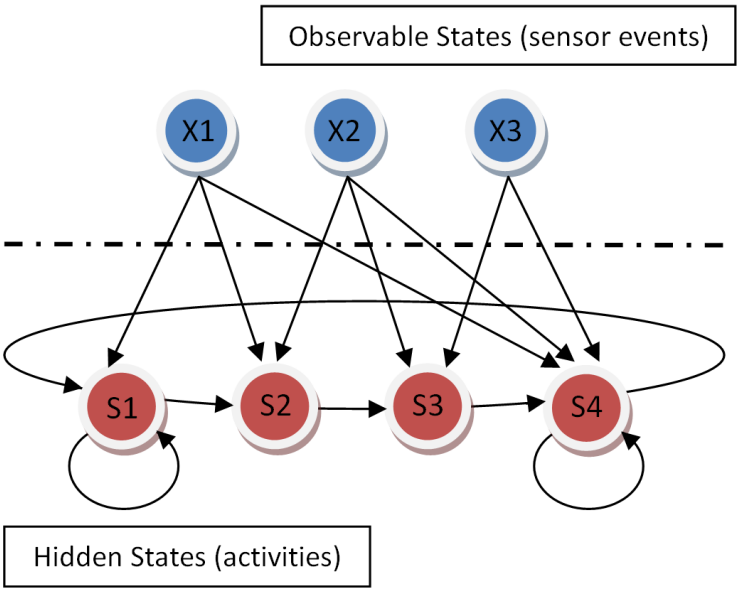
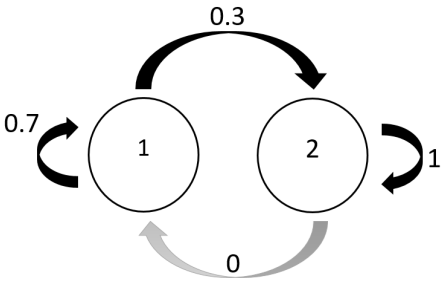


FIGURE 3.5: HMM basic structure.

3.3.3 Markov Models Zero Emission And Transition

There are cases, especially when the datasets contain a high number of features and/or classes, where not all the potential occurrences happen empirically in the training set. For example, if we have 10 different classes maybe transition from class 1 to class 7 never happened in the training set. It is also possible that a sensors has never been triggered during a determined class occurrence. In these cases, transition and emission probabilities will be set to zero when training the model parameters using expectation maximisation (EM) algorithms. However, there will be cases where, even though these examples never happened in the training data, they can potentially happen in the future. Therefore, the algorithms needs to be ready for this to happen.

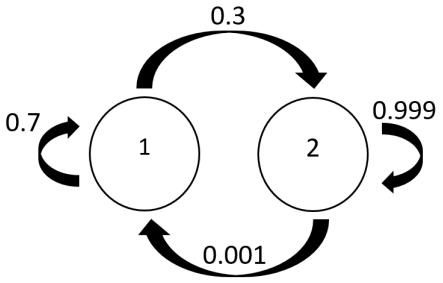
This problem can be solved by applying a small probability (typically 0.001) to each occurrence in the model observation or to each transition in the transition model, allowing a small probability to these unlikely events to be at least possible. The procedure is simply adding the small quantity to each zero value after applying the EM parametrisation and normalise the results accordingly (transitions and emissions are probability matrices therefore need to add to 1 row-wise).



Transition Probability

$A_{I,J}$	1	2
1	0.7	0.3
2	0	1

FIGURE 3.6: Basic HMM transition example where transition from state 2 to state 1 never happened in the training set ($a_{2,1} = 0$).



Transition Probability

$A_{I,J}$	1	2
1	0.7	0.3
2	0.001	0.999

FIGURE 3.7: Basic HMM transition example with transition smoothing.

3.3.4 Support Vector Machines

SVM models learns by representing feature points in space by creating a hyperplane, which separates the points of different classes. The most representative points in the boundaries are called support vectors and their position determines the final hyperplane shape, which will maximise the distance between support vectors from different classes. Points will then be separated into regions by the hyperplane and new classification points will be labelled based on their region location. In order to go beyond linear classification, SVM uses the kernel trick method, which consists of modelling feature relations (from input space to feature space) instead of the features themselves, allowing the use of multidimensional hyperplanes. With these hyperplanes, SVM learners can handle multivariate data (such as from occupancy typical datasets with many sensors), which can be thus processed and used for classification tasks.

We call our data $(\mathbf{x_i}, y_i)$ for $i = 1, ...m$, where m is the total number of samples, $\mathbf{x_i} \in \chi \subseteq \mathbb{R}^d$ is the input of sensor readings and $y_i \in \{+1,\text{-}1\}$ represent each of the labels (multiclass SVM also keeps as will be explained below). This model learns from data to

find a hyperplane function such as:

$$f(x) = \mathbf{w^T x} + b. \tag{3.4}$$

where $f(x)$ represents the plane as a function of the training samples $\mathbf{x}$, b is the bias term and $\mathbf{w^T}$ is the *weightvector*. In order to learn the model parameters b and $\mathbf{w}$, we want to optimise the function:

$$\frac{1}{2}||w||^2, \quad s.t. \;\; y_i(w^T \mathbf{x_i} + b) \geq 1, \tag{3.5}$$

which is solved by means of a Lagrangian multiplier, yielding the optimisation expression:

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y_i y_j \alpha_i \alpha_j \mathbf{x_i^T x_j} - \sum_{i=1}^{m} \alpha_i. \tag{3.6}$$

When (3.3) is solved using quadratic programming, returns a matrix of α coefficients for each $\mathbf{x_i}$. Substituting the values in α into one of the Lagrangian constraints we can solve the value for $\mathbf{w}$ and for the bias term b.

Also, the coefficients in α can be separated into zero and non-zero values. All the non-zero values in the α matrix will correspond to what we call the support vectors, which are the values in the dataset that determine the final shape of the hyperplane we are learning. For more information about SVM models, we refer the reader to the work in [30].

This kernel method enables the SVM to separate non-linear data as a linear model would do. Although this technique significantly increases the flexibility and the potential to successfully generalise with more complex data, it also increases the complexity of the quadratic programming optimisation, which in turn results in a much more computational intensive task.

We will be using here both linear and non-linear multiclass SVM models:

3.3.4.1 Linear SVM

The technicalities of this model were covered in previous sections. This linear SVM (LSVM) is a simpler approach than the non-linear SVM as this does not apply the

input into feature space mapping. This model has shown great potential for classification (unless the data is highly non-linear), and yields results that can challenge those obtained by other more complex models such as MCSVM, but faster and resource efficient. This algorithm is specially indicated for large quantities of sparse data with high number of features and samples. To apply this approach we used the liblinear MATLAB libraries [86].

3.3.4.2 Multi-Class SVM

The SVM is inherently a model to perform binary classification by nature. However, many of the real-world applications need to perform multi-class classification such as the present one. The SVM function used for our work performs the *one vs. one* [87] multiclass inference approach, which has been noted as the most convenient technique for multi classification tasks. The *one vs. one* involves creating a different classifier for each pair of classes, evaluating and finally selecting the most recurrent. Ties are also possible, this being solved most of the time by selecting a random class from the tied labels.

As shown in Fig. 3.8, we see an example of the *one vs. one* classification method for three classes (A, B and C), which also needs to learn three SVM classifiers. For a new point d, the pair A *vs.* B yields A, the pair B *vs* C gives C and the pair A *vs* C will predict class A. From the results sequence ABA, we predict the new inferred label would be A.

3.3.5 k-Nearest Neighbours

In the case of kNN, we can observe a very traditional yet simple classification method extensively used for pattern recognition. This model is one of the most simplistic ML approaches, as it just computes the distance between a new point and its closer neighbours to infer the label of the most repeated label amongst the neighbouring points. As seen in Fig. 3.9, in this case a 3-NN neighbour shows that the class 'triangle' is the most popular among the 3 neighbours chosen. The most recurring label will be the chosen for the new point's classification. For this experiments we used our own kNN function

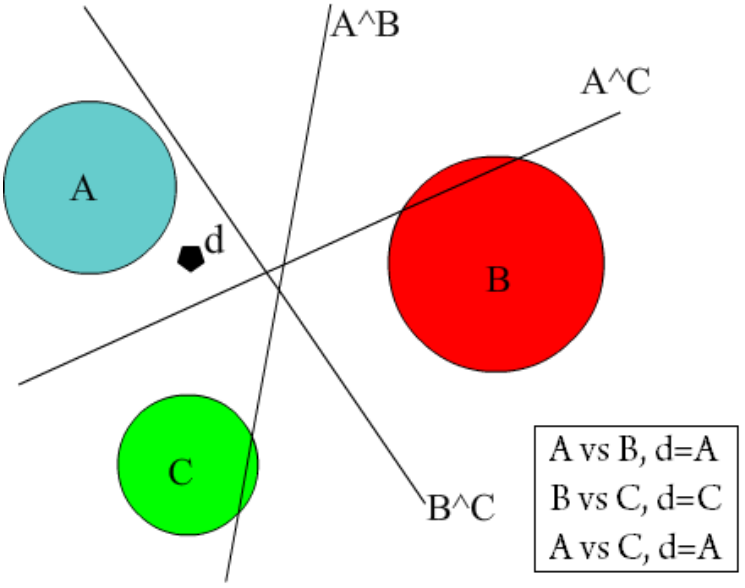


FIGURE 3.8: One vs one multiclass SVM approach.

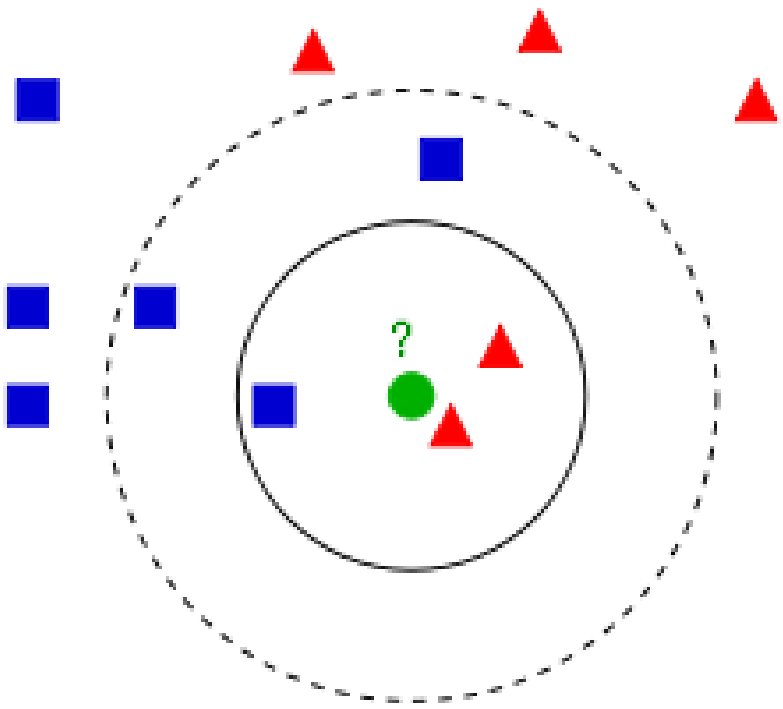


FIGURE 3.9: kNN models: The proximity of other data points will indicate the chosen label.

based on the euclidean distance, selecting the number of neighbours which reported better accuracies. A more detailed description of this model can be found in [88].

3.4 Experimental Evaluation

In this section we provide details on the experiments we have conducted to evaluate the performance of several state-of-the-art machine learning techniques.

3.4.1 Experimental Setup

Let $S = f(O)$ be our model and T a numerical set $T = 1, ..., N$ equal to the number of training samples; the training data extracted from our datasets is $\{S_T, O_T\} = \{(s_1, o_1), (s_2, o_2), ..., (s_N, o_N)\}$, where each s is the training label and each o is the sensor reading sample. If we consider that (s_1, o_1) occurred at time $t = 1$, and (s_N, o_N) at time $t = N$, which is when the last annotation in the dataset, we need to specify the time in between each time step $t = 1, 2, 3, \ldots, N$. This is what we call timeslices. When preparing the data for analysis, we need to specify what is going to be the resolution of the system, that is, what is the time between samples. As the dataset has a resolution of milliseconds, we could assign timeslice parameter as small as that value. However, if we opt to do that we would be creating $1000ms \cdot 60s \cdot 60m \cdot 24h = 86.4 \cdot 10^6$ samples per day, which is computationally unmanageable. Timeslices with a duration of a second long or more might be considered, but previous works as well as our own experiments suggest that this length is too costly in terms of computation times and do not really give a real improvement in terms of model accuracy. Based on previous literature and experiments including these datasets, we set an initial timeslice duration of 60 seconds [66][21] as these previous work suggested as the optimal value. As in both datasets the data was separated in days, we used a leave-one-out cross validation approach for separating training and testing data. This approach consists in leaving one day for testing evaluations while using the rest of the days for training. This is repeated for each day all over the dataset, and the resulting accuracies are expressed in terms of the mean of the results obtained for each day.

3.4.2 Unlabelled Data ('Idle Class')

Not all the data available in D1 and D2 is fully labelled. This means that not every o_t has a s_t label associated. This issue can be addressed in two different ways. The first solution would be to create an 'idle' activity and consider it as another different class,

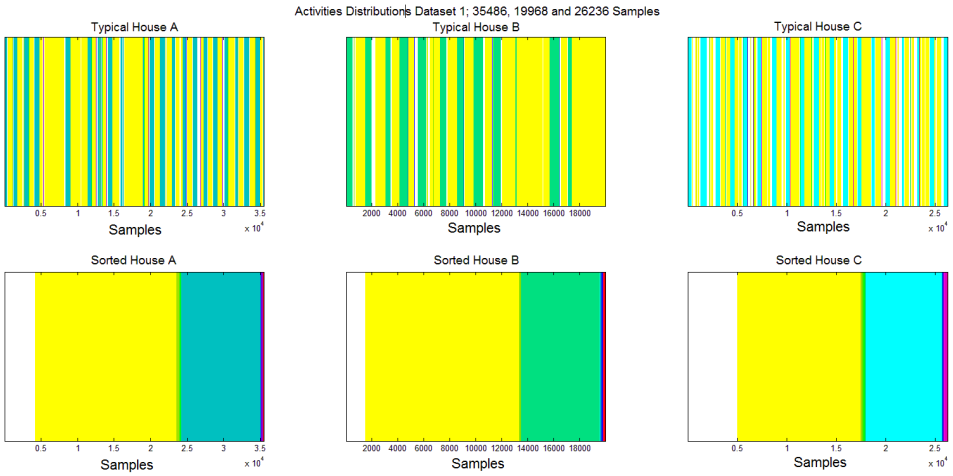


FIGURE 3.10: D1 samples. The unlabelled data is the white region, not as big as in Dataset 2 (between 7% and 12% of the total).

assigning every empty s_t to that label. The other option would be simply removing those samples from the training data.

Previous researches using D1 kept using the unlabelled data and using the 'idle' activity so we decided to keep this samples in order to maximize the use of the readings in D1. However, for D2 this approach was not advisable based on the periods between the first date and the last date that was not actually labelled. This is probably due to the fact that D1 was meant to contain continuous 24 hour data. The amount of unlabelled data for D1 was just of 12%, 7%, and 19% for House A, B and C respectively. Nevertheless, for D2, the amount of information from sensors not associated to any activity (the frequency of empty S_t) accounted for more than the 80% of the total samples. Due to this, the classifiers trained with D2 data predicted just class 'idle' for all the test points due to the massive imbalance of this new class 'idle' in addition to the fact that any sensor firing combination could have an 'idle' label associated since we do not know what activities were actually occurring during those blank timesteps.

To prevent this, all unlabelled datapoints had to be removed from the feature array, thus the label 'idle' was not considered. Ultimately, using a discretisation of 60 seconds between samples, D1 contained 35860 samples over 25 days and 10 activities and D2 contained a total of 9606 samples over the 56 days instead of the initial 80600 (See Fig. 3.10 and Fig. 3.11) and 9 different activities or classes.

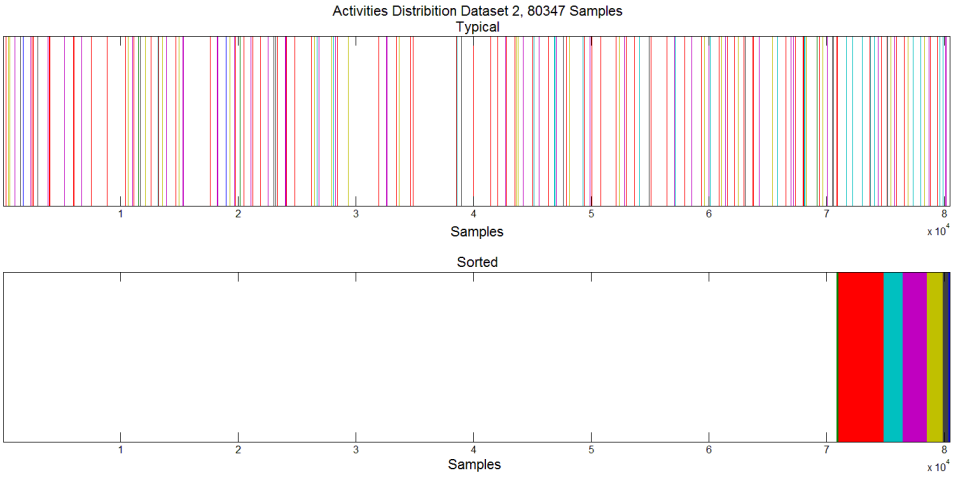


FIGURE 3.11: D2 activity occurrence. In the over 80000 samples, more than 70000 (white area) are unlabelled or 'idle'.

3.4.3 Model Performance Comparison

3.4.3.1 Dataset 1 Results

We have evaluated our models in terms of classification accuracy and we expressed the results in percentages ($True\_Positives/Total\_Predictions$ in %). After the experimental evaluation, we can conclude that HSMM clearly outperforms the accuracies achieved by HMM and KNN, and it gives similar results as to those seen for kNN and LSV. MCSVM performed better than the other methods evaluated in our experiment. For example, in the *House A*, MCSVM had 84% of accuracy with much more consistency with some days achieving more than 90%. Also in *House B*, this model clearly outperforms all the methods proposed in earlier works. In table 3.7 we can see the accuracies per model and scenario for D1. In the latter scenario *House C* the values reached only a 44.57% accuracy, however this is still the best overall option amongst our three methods.

TABLE 3.7: Accuracy (in percentages) results from the Performance Evaluation Experiment Dataset 1.

	HMM	**KNN**	**HSMM**	**LSVM**	**MCSVM**
HOUSE A	68.49%	73.06%	70.80%	80.63%	85.60%
HOUSE B	63.51%	65.12%	82.00%	73.53%	84.74%
HOUSE C	36.54%	40.23%	44.30%	43.21%	44.57%

Regarding the kNN, results were a 73% average accuracy for *House A* and slightly lower in the other two, which is not a bad performance for such a simple classifier. We conducted the kNN inference by modifying the neighbour parameter until finding

the peak performance when 87-91 neighbours were considered. Apart from the typical euclidean, other preliminary measurements for distance were evaluated (i.e. mahalanobis and correlation), yet the variations were too small to be significant and never above the values mentioned. We kept the euclidean approach.

In the case of the HSMM model, we can see a significant improvement over its HMM counterpart. HSMM rivals the best accuracies with the SVM approaches, getting similar results to LSVM and only surpassed by the MCSVM.

3.4.3.2 Dataset 2 Results

In spite of the good results showed in Dataset 1, in Dataset 2 results were generally poor (Table 3.8). All models achieved accuracies around 50%-60%. Because all models failed to capture the patterns on the data coupled with the fact that the same techniques performed significantly better using other dataset, we can assume that it was a problem either with the data itself or with the way the data was fed into the models.

TABLE 3.8: Classification accuracy percentage results from the Performance Evaluation Experiment Dataset 2.

	HMM	**KNN**	**HSMM**	**LSVM**	**MCSVM**
Dataset2	59.85%	48.32%	50.32%	52.66%	51.60%

In spite being the model with highest accuracy, MCSVM is by far the model that took more time to perform the calculations. While Markov models took lees than a second, MCSVM needed several minutes in 2 of the three scenarios. Based on these times, it can be argued that the MCSVM may be too slow to perform real-time predictions[1].

TABLE 3.9: Time from the performance evaluation experiments. The times are expressed in seconds.

	NB	**HMM**	**HSMM**	**KNN**	**LSVM**	**MCSVM**
HOUSE A	0.84	0.91	1.41	36.76	20.58	420.85
HOUSE B	0.51	0.55	0.89	18.55	10.84	47.31
HOUSE C	0.66	0.744	1.32	27.79	31.34	829.21

[1]Note that in this work we have discretised our data in timeslices of 60 seconds. Therefore, we need to develop an on-line approach fast enough to be able to incorporate a new point and give an estimated class prediction within 1 minute.

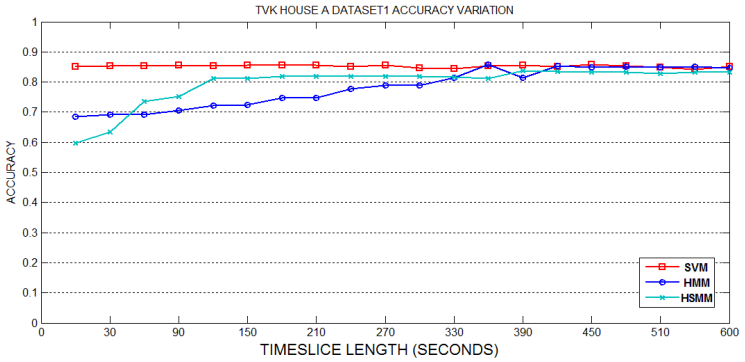


FIGURE 3.12: TimeSlice Approach House A variation results.

3.4.4 Supplementary Experimental Evaluation

The previous experiments were conducted using time steps of 60 seconds. However, after reviewing the previous works that used the D2 dataset, we noticed that they were proposing a different approach to pre-processing the data. In order to evaluate our model in similar conditions and make real comparisons of our experimental results compared to them, we proposed an alternative method for our experimental evaluation. We called this method the chunk data approach (CDA) and it consisted in grouping together all the samples occurring while the same state is happening. To illustrate this, we define the differences between the timeslice approach (TSA) variation and the chunk data (CDA) pre-processing and we evaluated these approaches using Dataset 1 and Dataset 2.

3.4.4.1 Timeslice Variation (TSA)

We evaluated the models to each scenario and observed how the accuracy changed for different timeslices. Despite the loss of information due to the increase of the TSA duration, models were able to show steady performances while varying the timslice parameter. The other models also maintained the values within certain levels, yet showing some variations. However, none of them indicate any significant change except in the case of House C, in which SVM boosted its classification accuracy from around 40% for 30sec timeslices up to over 60% for the bigger timeslices of 10 minutes. This is also an indication of the adaptability of the support vectors to different input spaces.

Fig. 3.12, Fig. 3.13 and Fig. 3.14 show these variations for each of the scenarios in D1; results are shown the more relevant approaches: HMM, HSMM and MCSVM.

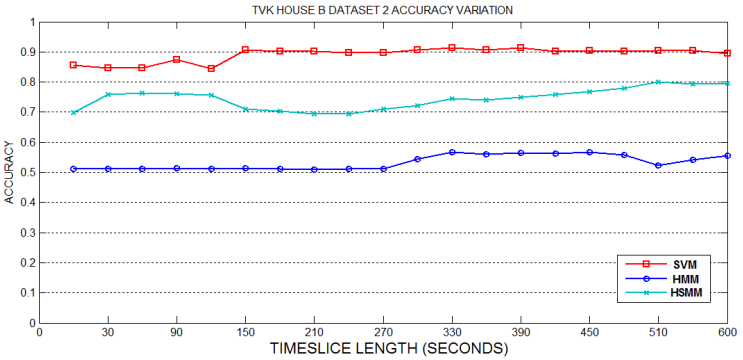


FIGURE 3.13: TimeSlice Approach House B variation results.

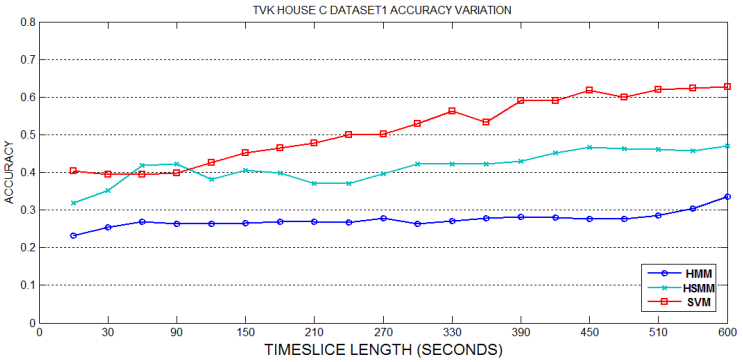


FIGURE 3.14: TimeSlice Approach House C variation results.

This might seem contradictory with the fact that the longer the timeslice, the more information about real sensors readings is lost. However, as we are working with unbalanced datasets, there is a chance that the samples which are going to be lost earlier (as we increment timeslice duration), are those labelled as the less frequent activities. Consequently, fewer classes are to be classified and the models perform better with less infrequent states. As discussed in previous sections, D1 dataset is divided into three different scenarios named House A, House B and House C; comprising 25, 14 and 19 days of data respectively. We have evaluated the models using different sample lengths. We evaluated the three models for timeslices ranging from 30 seconds per sample to 10 minutes. We evaluated each scenario independently performing a cross validation dividing the data into days, testing one of them while leaving the rest for training (leave one out), and then calculating the average over the obtained results.

TABLE 3.10: Comparison of Timeslice and Chunk approaches

Time (mins)	t=1	t=2	t=3	t=4	t=5	t=6	t=7	t=8	t=9	t=10	...
Labels			Act1					Act2		Act3	...
Timeslices (60s)	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	...
Chunks			Sample 1					S 2		S 3	...

3.4.4.2 Chunk Data Approach (CDA)

All the results obtained when training the models using the D2 and the TA approach, were really bad in terms of accuracy levels (around 50% of accuracy). Comparing our results with the ones reported in publications associated with the dataset [64], we noted they achieved much higher accuracies even when using similar methodologies. That pointed to the pre-processing technique of the data as the reason of the poor performance rather than in the model selection. Under this scope, the data was processed in chunks instead of timeslices. Each chunk of data contained all the sensors events happened while an activity was active. The CDA approach also implied removing all the unlabelled data from the D2 and the final number of samples was a total of 600, which is the number of any activity occurrence through the whole dataset. See table 3.10 for an example comparing timeslice and chunk methods. In this figure, 3 classes (activities) occur during 10 minutes. In the 60 sec timeslice approach, a sample is generated each minute. So, for example, we have 6 samples labelled with Activity 1, 3 with Activity 2 and 1 more with Activity 3. For chunk approach, we simply have one occurrence of Activity 1, Activity 2 and Activity 3.

For D2 we we saw that TA approach were really poor. However, applying the CDA pre-processing approach to the D2 made the accuracies rose to values around 80%-90% (see Fig. 3.15). This clearly points towards the hypothesis that pre-processing techniques also play a critical role in model performance. Also, this means that the CDA can only be used for offline purposes as we need all the data events occurring to define an state event, while the TSA is the suitable pre-processing technique for online purposes as we can define an state as a succession of sequential timestamps labelled with the same class.

Results for DATASET 2, Timeslice Approach (60 secs.) & Chunk Data Approach for pre-processing.

CASAS RESULTS	NB	HMM	CRF	—	EXP 1	SVM	HMM	HSMM
TA	N/A	N/A	N/A	—	TA	51.60%	59.85%	48.32%
CDA	80.33%	75.50%	90.77%	—	CDA	**91.67%**	75.33%	84.16%

FIGURE 3.15: Experimental results for D2 compared with previous researches using this data and our 60 second TSA.

3.5 Discussion

From the initial classification accuracies, the SVM approaches obtain the best results in the majority of the experiments conducted. However, this does not make them the best option for all cases. After studying the time the models need to process the data, the support vector models need much more time (and therefore memory) to perform the operations. In fact, Markov models performed all the operations in less than a second, while SVM reached values over 10 minutes in some cases. For this reason, for applications where the time is critical these approaches cannot be used to create the models even though showing high levels of accuracy.

On the other hand, we know that the existing HMM approaches model state dwelling time by allowing the system to self-transition from state Si to the same Si calculating the probability of remaining stationary each time step, which means multiplying a probability each time step, and hence state duration is inherently exponentially distributed. HSMM-based approaches, on the other hand, allow to calculate the most probable dwelling time d based on explicitly modelled duration distributions, during which the state will remain unchanged. However, in real scenarios, the state duration should be better captured by using different temporal distributions other than exponential distribution.

Moreover, both approaches present limitations when attempting to make activity predictions in an on-line fashion. Markov models traditionally infer probable states based on whole sequences of observable data. In fact, some of the inference algorithms usually associated with these models including the Forward-Backward or the Viterbi algorithms need a whole batch observable sequences to make their estimations. This poses a real challenge for state prediction of streaming data as future observable data is not available when a new state is reached. Therefore, state prediction has to be decided with the uncertainty of what observable sequence follows. In addition, for the state prediction

of occupant presence or absence, the existing HMM or HSMM based models simply treat all sensor events presenting occupancy state, which affects the overall accuracy performance of prediction. In fact, in real scenarios, individual sensor's contribution to the state prediction is different and should consider weighting factor for each individual sensor event.

Compared to HMM which is one of the most popular choices in current literature, HSMM has clearly outperformed this model for the datasets and techniques used in these experiments. Although other approaches such as the MCSVM also showed good levels of classification accuracy, they suffer other limitations when taking into account model efficiency and complexity. Time is crucial in OBPM, therefore this prevents these methods from being used in this context. Conversely, HSMM models present a unique trade-off between performance and speed/complexity showing good accuracy results while using a very small percentage of resources to process the data compared to other methods. Therefore, if we are able to improve the HSMM performance and address the specific limitations of this model, we could take advantage of its consistency and speed.

Since OBPM models often are used in online settings, based on our analysis, we can conclude that novel approaches need to be able to perform state prediction using just initially partial data and achieving performance results even beyond than traditional HMM and HSMM approaches. In addition, new approaches need to be able to process stream data and dynamically adapt the model predictions to the new included data. All these operations will ideally be done while preserving high levels of classification accuracy.

Chapter 4

A Novel Online Dynamic Hidden Semi-Markov Model For OBPM

To address the limitations mentioned in previous chapters, our objective is to build a model that can perform online occupant behaviour pattern modelling via streamed sensor data. In order to be able to detect occupancy or activities in real-time, we need that our model can process streamed data and that its parameters can be incrementally updated with new data while ensuring high levels of classification accuracy. We separate this into two aspects: 1) Online detection and 2) Incremental online detection. In this chapter, we will mainly focus on the first aspect/development. Here we present a novel approach that can perform online occupancy detection, while in Chapter 5 a novel incremental online learning approach will be presented on order to address the incremental aspect of the OPBM modelling.

As our online DHSMM is specially designed to overcome some limitations when using HSMM models for online OBPM modelling, in the following sections we present the limitations of traditional HSMM/HMM's and our proposed approach to address them.

4.1 HSMM Background

As discussed in previous chapters, HSMM were introduced to allow modelling state duration explicitly [15][89][90]. Therefore, in addition to HMM parameters (namely prior, transition and observation models), HSMM also makes use of a duration parameter. This

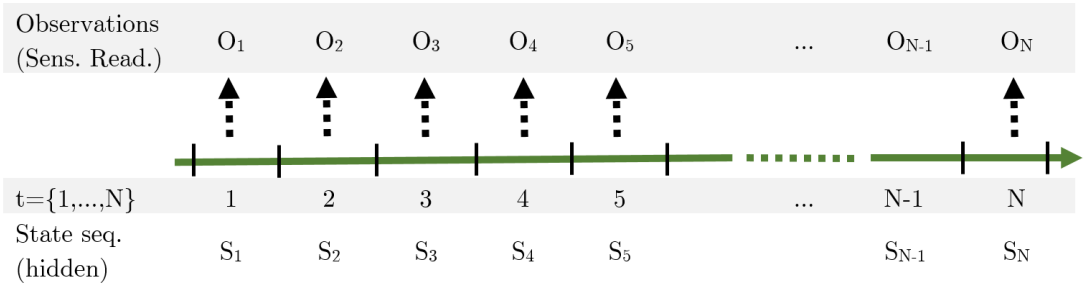


FIGURE 4.1: HMM Structure.

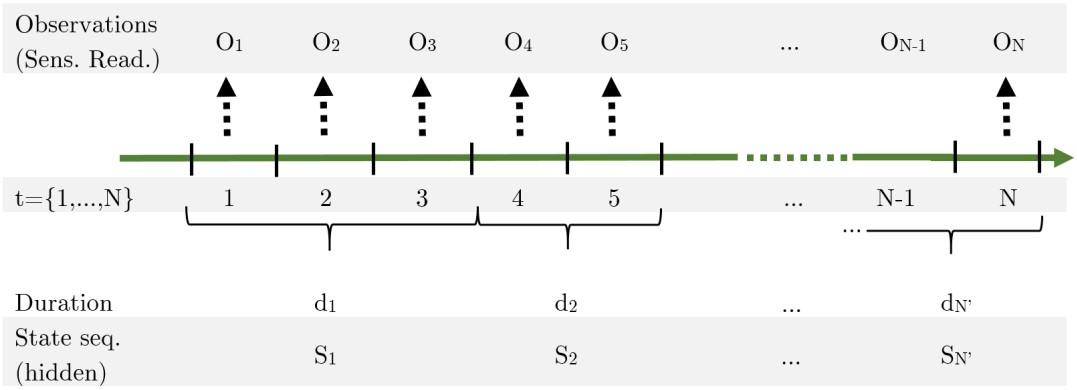


FIGURE 4.2: HSMM Structure.

means that transitions do not occur at each time step, but states remain unchanged for a determined duration (See Fig. 4.1 and Fig. 4.2). In the HMM approach, if we have N samples, we also have N number of states associated to a N number of observations. Whereas in the HSMM approach, we have multiple observations happening during the same state. Therefore, the states sequence will always be equal or less than the number of observations $N <= N'$. In typical OPBM datasets, N' will be commonly significantly smaller than N. For example, in a model where the sampling time is one minute and states or classes have an averaged duration of 10 minutes, then N' will be ten times smaller than N.

4.1.1 HSMM Parameters

Let S_T be a sequence of states, divided in N samples where each S_t, $t \in T = \{1, 2, ..., N\}$ represent each possible state at time t and M would be the number of different states.

As discussed in previous sections, HMM can only model state duration exponentially. To overcome this, the hidden semi-Markov model was introduced as an extension of

the original HMM, where additional parameters were included to model state durations explicitly. HSMM parameters can be described as follows:

$$\lambda = (Q, O, A, B, \delta, \pi). \tag{4.1}$$

Where, $O = \{o_t | t \in T\}$ and $Q = \{q_t | t \in T\}$ are observations and states respectively. A represents MxM dimensional state transition matrix, which includes the probability of transitioning from state i to j, $a_{i,j} = P([q_t = s_j | q_{t-1} = i])$ with the form:

$$A = \{a_{i,j}\} = \begin{pmatrix} a_{1,1} & a_{1,2} & ... & a_{1,M} \\ a_{2,1} & a_{2,2} & ... & a_{2,M} \\ ... & ... & ... & ... \\ a_{M,1} & a_{M,2} & ... & a_{M,M} \end{pmatrix}$$

The emission probability (observation model) B is a MxN matrix containing the probability $b_i(v_i) = P([o_t = v_i | q_t = s_i])$ for each state to trigger each of the observations v_i, $i = \{1, 2, ..., k\}$ where k is the number of features. The parameter π expresses the prior probability of each state $\pi_i = p([q_1 = s_i], i = \{1, 2, ..., M\})$.

Finally, duration parameter δ is the (MxD) matrix where D_i expresses maximum state (max_dur) duration and $\delta_i(d) = P(\delta = d | s_i)$ is the probability of state s_i lasting for d time steps.

As parameters Q and O are basically data, we will subsequently drop them from the parameter set, which will be defined by: $\lambda = A, B, d, \pi$. Given the above parameters, we can express the HSMM joint probability of observations and hidden states using:

$$P(O, S) = \prod_{t=1}^{N} P(O_t | S_t) P(S_t | S_{t-1}, d_{t-1}) P(d_t | S_t) \tag{4.2}$$

For the maximisation of this joint probability, several dynamic programming algorithms have been proposed which allow to calculate the most likely state sequence for a given $\{O_T, \lambda\}$, the most likely observation sequence for λ, or the most probable parameters that maximise $\{S_T\}$. See Rabiner et al. HMM tutorial for more information about these techniques [15] and Yu's [91] work for the HSMM case.

4.2 Main Current Limitations

Attending to the given definitions, we can reason some direct limitations:

1) Observations contain just one value per sample, which means that there is no way to give more relevance to certain sensors (inputs) in scenarios were the physical deployment of the sensors favours the interaction of some of them over the rest. For example, a sensor in a central hallway will be more likely to be triggered that one located in a small corner of a less used room.

2) If a state is defined throughout all its duration, we need all the observations at the same time in order to estimate what state and for how long is going to occur (similar to the chunk approach studied in Chapter 3).

4.3 The Proposed Online Dynamic Hidden Semi-Markov Model (DHSMM)

Having identified the existing limitations, we need a solution to improve the current approaches performances.

Firstly, we need to define how HSMM model parameters correspond to our occupancy detection context. We consider our approach within the definition of a discrete Markov process [15], where sensor signals have discrete values and conform the input features which are used to create the observation model. The transition matrix defines the likelihood of moving from one state to another, and the duration model can be built upon the temporal behaviour each state showed in the training set.

Our new dynamic hidden semi-Markov model (DHSMM), extends HSMM traditional models to overcome the previously discussed issues. We based our approach on explicit HMM (HSMM), which assume that transitions are independent of the previous state duration, self-transitions are not allowed ($a_{ii} = 0$) and state duration is only dependent on the current state and independent on the previous one [37].

- **Novel Weighted Parameter Model**: In order to be able to give more significance to determined sensors based on their actual relevance, we included the

features separately and we have assigned weights associated to each different feature to make accurate initial state predictions. This will allow us maximise each one to the observations from data and help us to predict states using just one time step input (every time a new state is reached). This could be optional for batch inference, but it becomes crucial for 'live' purposes where only current time data is available.

- **Novel Dynamic Duration Model**: To make our model more flexible and also to alleviate potential issues when our predictions make a transition prediction misclassifying the transitioned state, we have extended our model to calculate the probability of remaining in one state dynamically throughout all the time a state is occurring. When performing state inference in an online setting, previous works based on semi-Markov approaches calculated this by sampling a random duration τ (τ being the number of steps the state must remain) from the distribution model each time a new state is entered [16][92]. Therefore, the state remains stationary (fixed static duration) for τ number of steps. Our DHSMM is able to model duration as the probability of remaining in a state by using a complementary cumulative distribution function (1-CDF). After the initial state prediction, our model combines the probability of remaining with new observations so the next state transition can be found dynamically.

4.3.1 Weighted Observation Model

The physical sensor deployment can modify the importance each of the signals within the model. For example, in an activity dataset including motion sensors, a sensor located in a hallway (more transited area) will be fired probably more times compared to one located in a corner of the kitchen. Therefore, when the sensor of the hallway is fired, it does not give much certainty about the occurring class. However, if the one in the kitchen is triggered, the potential activities are reduced to the ones related to *eating* or *cooking*. In fact, if we know that the corner is often used only when using the microwave, a trigger in that sensor will give a high likelihood that an activity related to cooking (using the microwave) is occurring.

In the original HSMMs and HMMs with multiple inputs, the observation model is estimated by considering all sensor inputs at a time t as a unique observation o_t feature,

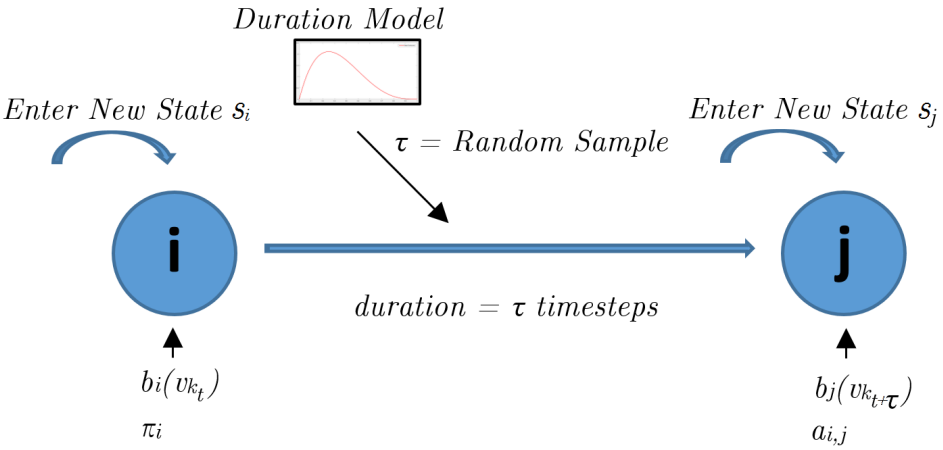


FIGURE 4.3: Online state inference with initial τ random duration sample.

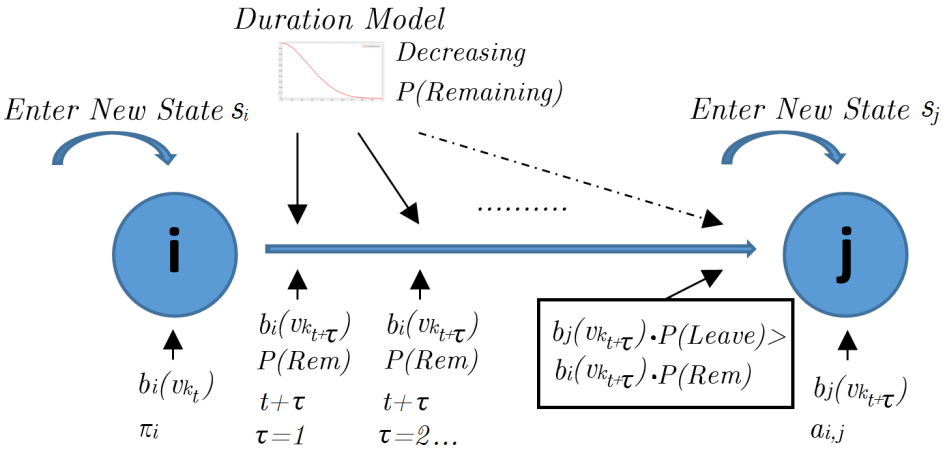


FIGURE 4.4: Online state inference with DHSMM duration model, dynamically detecting duration.

consisting of an array of sensor signals at a determined time. An alternative way of doing this, can be treating each sensor signal as a different observation feature happening at the same time. Therefore, to compute the emission probability in the first case we have the probability of occurring a combination of sensor signals whereas in the latter we need to calculate the aggregated probability of each of the sensors for a given state. To illustrate this, let's suppose we have 3 binary sensors signals $z1 = 1$, $z2 = 1$ and $z3 = 0$ at time t. Given a system with two possible states s_i, i={1,2}, the two possible observations would be:

1) Probability of one unique sequence [z1 z2 z3] to be [1 1 0]

$$b_i(z1, z2, z3) = \tag{4.3}$$

$$P(z1, z2, z3|S_i) = \frac{P(sequence = [1, 1, 0]|s_i)}{\sum_i P(s_i)}; \quad or \tag{4.4}$$

2) A different probability for each sensor

$$b_i(z1, z2, z3) = P(z1, z2, z3|s_i) = \tag{4.5}$$

$$\frac{P(z1 = 1|s_i) + P(z2 = 1|s_i) + P(z3 = 0|s_i)}{R_t} \tag{4.6}$$

where R_t is the normalising factor, subject to $\sum_i b_i = 1$.

The first approach is not the best option for occupancy datasets as many sensors can be involved. Therefore, a lot of potential combination might not occur in the training set but be present in the new incorporated data. The second approach overcomes this by aggregating the likelihood of each sensor separately. However, this approach evens the significance of each sensor, yet in many applications has been noted that this is not the right modelling approach as some sensors contribute more to the model outputs [14][56]. To improve the observation model further, we propose including a weighted factor cR that is applied to each sensor signal as follows:

$$b_i(z1, z2, z3) = P(z1, z2, z3|s_i) = \tag{4.7}$$

$$\frac{cR_1 \cdot P(z1 = 1|s_i) + cR_2 \cdot P(z2 = 1|s_i) + cR_3 \cdot P(z3 = 0|s_i)}{R_t}. \tag{4.8}$$

4.3.1.1 Correlation Function

In order to calculate the weight parameters, we use a correlation function applied to each sensor signals during the training phase to find the more significant sensors. We then use the correlation values to assign the weights. This ensures the emission modelling will adapt more realistically to different scenarios that include various sensor topologies and sensors of diverse nature. The equation of the correlation function between two variables (A, B) for N number of samples is as follows:

$$corr(A,B) = \frac{1}{N-1}\sum_{i=1}^{N}\frac{A_i - \mu_A}{\sigma_A}\frac{B_i - \mu_B}{\sigma_B} \tag{4.9}$$

This can also be computed as a function of the covariances of the variables:

$$corr(A,B) = \frac{cov(A,B)}{\sigma_A \sigma_B} \tag{4.10}$$

In this approach, we calculate cR by comparing each sensor signal with the rest of the sensors, and obtaining the correlation between them for each of the states that appear in the dataset. Then the absolutes of this values are aggregated and normalised for each cR_i.

4.3.2 Dynamic Duration Prediction Approach

Following the idea of creating a non stationary time dependant duration model [22], when state duration inference is done using DHSMM, a transition boundary is not agreed initially by sampling a likely duration τ when a new state is reached. Instead, we use a Complementary CDF function (1-CDF) fitted from the training data set to calculate the probability of remaining $P(Rem)$ in the same state each time step. Initially, $P(Rem)$ will be forced to 1 for the number of timesteps specified in the minimum duration parameter *min_dur*. During that initial period of time, the state will remain unchanged regardless what observations occur. Once *min_dur* has ended, the system will decide whether to remain or leave the current state based on:

Algorithm 1 Dynamic transition detection.

1: **if** $b_i \cdot P(Rem) > b_j \cdot (1 - P(Rem))$
   then $S_t \leftarrow s_i$ ▷ No transition. State remains
2: **else** $S_t \leftarrow s_j$ ▷ Transition occurs. State changes

If no transition eventually occurs, $P(Rem)$ will decrease according to the CCDF until reaching the value of maximum duration *max_dur*, where $P(Rem)$ will be set to 0 thus forcing the system to transition to other state.

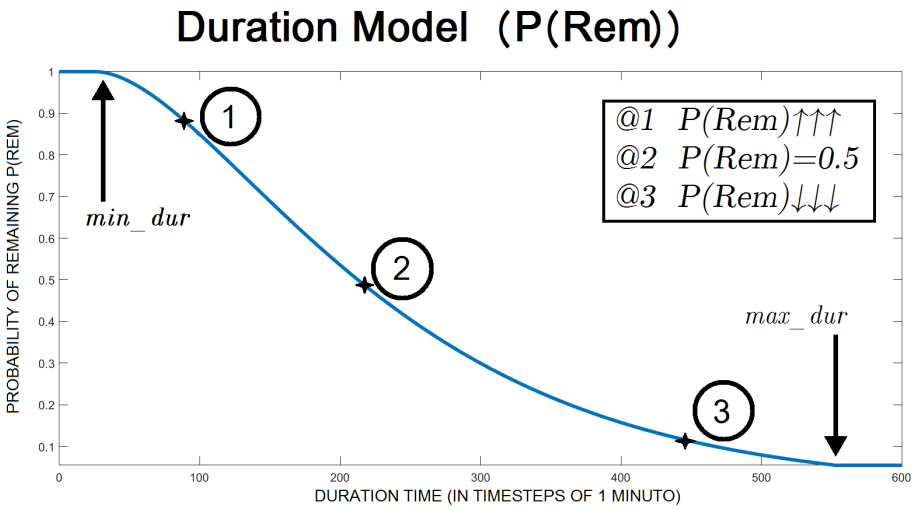


FIGURE 4.5: DHSMM duration model. In this example, P(Rem) @1 will be close to 1 (unlikely transition); @2 P(Rem)=0.5, so transition will be determined by the observations only; @3 as P(Rem) reaches 0, the system will be pushed to enter a new state.

As can be seen in Fig. 4.3, the traditional approach for predicting a state duration is based on sampling a duration time from the previously trained duration model. When entering a new State S_i, the τ time is calculated and the estate remains until this time finishes. Then, based on the observation (b_j) at time $t + \tau$ and transition probability $(a_i j)$, a new state is chosen and a transition occurs. The dynamic duration model however, as can be seen in Fig. 4.4, does not calculate a fixed τ. Instead, from the 1-CDF of the duration model (duration model specifies a probabilistic distribution of state duration for each state), a probability of remaining P_Rem is calculated. Each timestep, the probability of remaining combined with the observation at that time is compared with the probability of leaving. If the current stae is i and we are monitoring the potential transition to a following state j, this can be expressed as:

$$b_j(vk_{t+\tau}) \cdot P(Leave) > b_j(vk_{t+\tau}) \cdot P(Rem), S_{t+\tau} = j \tag{4.11}$$

$$b_j(vk_{t+\tau}) \cdot P(Leave) < b_j(vk_{t+\tau}) \cdot P(Rem), S_{t+\tau} = i \tag{4.12}$$

This means that while the probability of leaving is not higher that the one of remaining, the state i is not leaved.

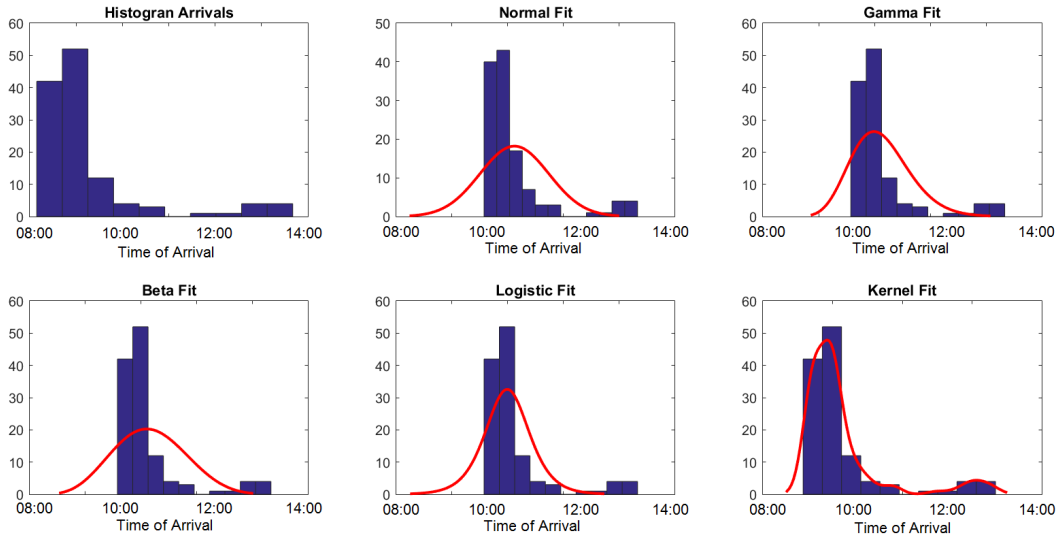


FIGURE 4.6: Example of different statistical functions fitted for arrival times.

To illustrate this, the example in Fig. 4.5 shows a CCDF generated from fitting the data with a gamma distribution. $P(Rem)$ is set to 1 until min_dur. After that, each time step DHSMM will calculate the joint probability of remaining and the observations $P(Rem, o_t|s_i)$ for all possible states. $P(Rem)$ will decrease over time and the transition will happen when the joint probability of leaving will be higher than to remain ($P(1 - Rem, b_j|Leave) > P(Rem, b_i|Remain)$); setting $P(Rem) = 0$ when max_dur is reached. In our experiments, we have used a mixture of histogram visual inspection and a MSE error function to choose the best probabilistic function fit for each class (see Fig. 4.6), from which we will construct our CCDF. Our approach also allows to use several different statistical functions for each state duration model.

4.4 Experimental Evaluation

In the experiments conducted in Chapter 3, we used activity datasets. With the objective of giving our experiments a more comprehensive approach to general OBPM models, we wanted to evaluate our approach using an occupancy dataset. In spite of a recent number of datasets being made publicly available recently, it is difficult to find quality occupancy datasets in terms of the number of features and samples, the reliability of the source, or their impact in previous works. We therefore decided to modify one of the activity sensor dataset [21] and adapt its contents to simulate an occupancy dataset. Since an occupancy dataset could be based on presence and absence states, we modified the

dataset so all the 'quiet' activities (including $leave_house$ or *sleeping*) where labelled as 'absence', and the rest of the activities where labelled as 'presence'. This is consistent with the fact that we labelled 'presence' all activities that required active occupant presence and 'absence' the ones that not.

We have compared our approach with existing HMM and HSMM-based approaches for real time occupancy detection. We have fed streamed data into the three models to compare the classification accuracy when performing online occupancy detection. First, the models have been trained offline and then, in each timestep, a likely state has been predicted and compared with the labels (ground truth) in the test set.

4.4.1 Data Description

The data as explained above, was composed of 2 classes representing occupant presence and absence. The observation data consisted of 14 different binary sensors as shown in the description of D1. After pre-processing the data, there were a total of approximately 12,000 labelled samples, each representing an observation sequence of 14 features for each time step, which were discretised into slices of 60 seconds.

4.4.2 Evaluation Metrics

To asses model performance, we have compared our DHSMM against traditional HMM and HSMM approaches. We have performed n-fold cross validation and used various standard metrics in [93] including: **Accuracy** $= (TP + TN)/(P + N)$, **Precision** $= (TP/TP + FP)$, and **Recall** – $TP/(TP + FN)$. where P represents positive samples; N represents negative samples; TP represents true positive; TN represents true negative; FP represents false positive; FN represents false negative. Each new data point has been processed by our system in a streaming fashion.

4.4.3 Results

In Fig. 4.7 we can see a comparison of the outputs of an HMM, HSMM, DHSMM and the ground truth respectively. For each timestep, a probable class (absence or presence) is generated. The more these pictures resemble the ground truth distribution,

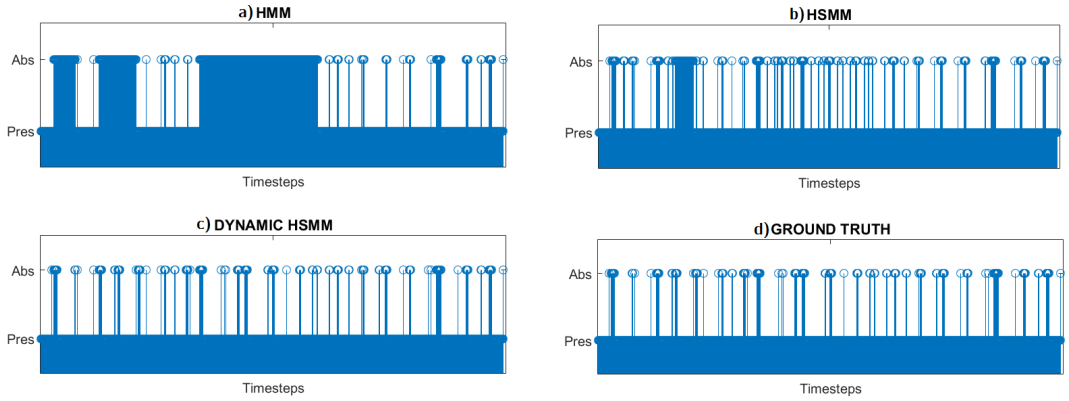


FIGURE 4.7: Predicted states from a) HMM, b) HSMM, c) DHSMM against the ground truth in d).

the more accurate the classifier is. Compared to the ground truth in Fig. 4.7 (d) Ground Truth), the HMM-based approach fails to reproduce the dynamics of the fast paced short absence periods in many cases. The HSMM-based approach significantly improves the representation of short periods compared to the HMM-based approach. However, it fails to reproduce the sequences accurately.

The DHSMM increases the classification performance of Markov models as can be seen in the graphs. The weighted inputs can be seen in Fig. 4.9, where we can see how sensor 9 has the largest weight associated, and other such as 4, 8 or 12 have very little contribution.

Fig. 4.7 and Fig. 4.8 show that our proposed DHSMM clearly outperforms the HMM and HSMM-based approaches.

In Fig. 4.8, the accuracy, precision and recall are shown. Our proposed DHSMM has a high accuracy of over 98% outperforming the HMM based approach 65.6% accuracy and the accuracy of the HSMM 91.7%.

For precision and recall, our DHSMM achieves high levels of recall (99.28% and 91.92%) and precision (99.16% and 92.99%). The results show that DHSMM significantly outperforms both HMM and HSMM and is able to process streaming data with high degrees of accuracy for this dataset.

HMM

Accuracy: 65.6%		Ground Truth		
		Presence	Absence	Precision
Predicted	Presence	6795	4034	62.75%
	Absence	77	1036	93.08%
	Recall	98.88%	20.43%	

HSMM

Accuracy: 91.7%		Ground Truth		
		Presence	Absence	Precision
Predicted	Presence	9975	854	92.11%
	Absence	139	974	87.51%
	Recall	98.63%	53.28%	

D-HSMM

Accuracy: 98.6%		Ground Truth		
		Presence	Absence	Precision
Predicted	Presence	10738	91	99.16%
	Absence	78	1035	92.99%
	Recall	99.28%	91.92%	

FIGURE 4.8: Confusion matrices, accuracy, precision and recall.

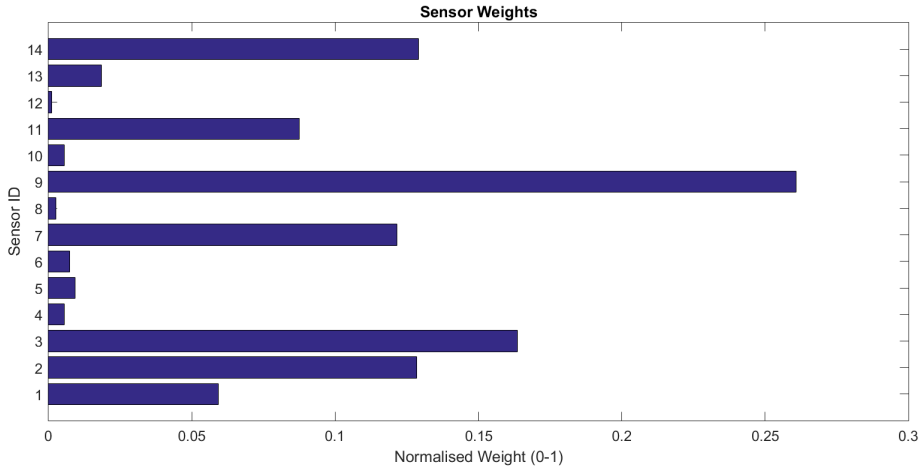


FIGURE 4.9: cR_i Normalised weights corresponding to each sensor.

4.5 Discussion

Based on the findings in literature and the findings in our own experiments, we already identified some of the challenges for OBPM. On order to create a solution that could

take advantage of the Markov approaches that could perform online operations, we have proposed the new DHSMM model. Comparing the HMM and HSMM model, we can say that the semi-Markov property improves the classification accuracy. This is probably due to the fact that OBPM datasets are related to time (both in terms of when the states happen and how long do they last). Thanks the the dynamic duration model and the weighted function we have proposed here, we have demonstrated that semi-Markov models can indeed be used to perform online detection and that our DHSMM addresses successfully some of the limitations noted in HMM and HSMM models when comparing the classification accuracy.

The accuracy of DHSMM achieved 98% while the HMM-based approach shown 65.6% accuracy and the HSMM model 91.7%. This means an increment of 50% and 7.5% respectively. Although the HSMM model hugely improved the initial HMM values, our DHSMM outperformed both approaches. Based on these results, we can conclude that our DHSMM algorithm achieves the highest levels of classification performance while handling online streamed data. Our approach improves the HMM and HSMM results using the same data and under the same conditions for all the metrics evaluated. Our model can accurately capture and predict occupancy periods with partially available data.

However, our model still suffers from an important limitation. This comes from the fact that in the context of OBPM, it is not unusual that the patterns in the data change with time. Since the model parameters are the responsible for capturing those patterns, when data patterns change, parameters should be able to change accordingly.

In the next Chapter, we have extended our Online DHSMM to perform incremental learning. This will allow our model to be able to adapt to changes in data while improving its computation performance while maintaining almost the same good result in classification accuracy.

Chapter 5

Online Incremental Learning Of Dynamic Hidden Semi-Markov Model (DHSMM)

This chapter will focus on developing a new approach to incrementally learn over data stream for occupancy detection, based on the online DHSMM developed in Chapter 4.

We describe incremental learning (differentiating it from online learning [94]) as the updating of model parameters without storing any of the previous samples, which are simply discarded. This is done by re-estimating parameters iteratively with new data point, that are processed by the model but not kept in memory afterwards. As long as new sensor data is observed, new values are given to the parameters in order to incorporate the new information received.

The advantages of this approach are significant. Firstly, we can incorporate new knowledge that perhaps it was not initially present in the training set. Therefore, enabling our model to dynamically adapt to potential changes in the patterns underlying the data. For example, an occupant might alter his/her sleeping patterns after getting a new job or occupants starting a diet can change their eating/cooking patterns as well. For example, if the activity *sleeping* would usually be modelled as starting at 8pm and lasting for 7 hours, maybe after the new job the occupant has to go to bed not earlier that 11pm and for a maximum period of 6 hours (due to shifts). In such an scenario, we want to allow our model to potentially change its parameters to adapt to the new

patterns in the data. Furthermore, performing incremental updating means that the model do not need to be retrained from scratch, therefore improving the efficiency of the model. In our context, fast real time updates are crucial as explained in Fig. 5.1. Here, we see how in an online setting, after the initial training model, a new point 1 is included and a certain amount of time is needed to make a prediction and posterior update of the model before new point 2 arrives. All operations must be finished by the time the next point arrives in order to prevent lagging. After completing this, we can discard all previous data and keep only the relevant information for us, which is 'stored' in the actual model parameters. Finally, when scarce data is available at an initial time, a potentially inaccurate model can be incrementally updated to eventually match the parameter estimation of the same model trained batch learning using the whole dataset.

All these concerns are particularly relevant in our context, as OBPM models are subject to potential changes in occupant patterns. In addition, when the information has to be used online to regulate BEMS or other smart home applications speed becomes crucial. Therefore, our proposed model has to be able to perform incremental learning and do it fast while maintaining high levels of accuracy performance.

Although some authors have devoted time and effort to the study of Markov and Bayesian models in the context of the online learning and the incremental learning (i.e. [95][96][97]), little work has been conducted to use semi-Markov approaches in OBPM combination with online and incremental learning techniques. The current HMM incremental parameter updating approaches, have significant limitations when these models are extended to include hidden semi-Markov properties. Furthermore, in order to perform updates for our specific DHSMM model, further considerations need to be done due to the special nature of the dynamic duration model and the weighted input observation model.

In order to describe how our approach can perform incremental learning from new data, we start by introducing the three inference algorithms traditionally used for Markov models as a starting point to describe how our approach overcomes its specific challenges.

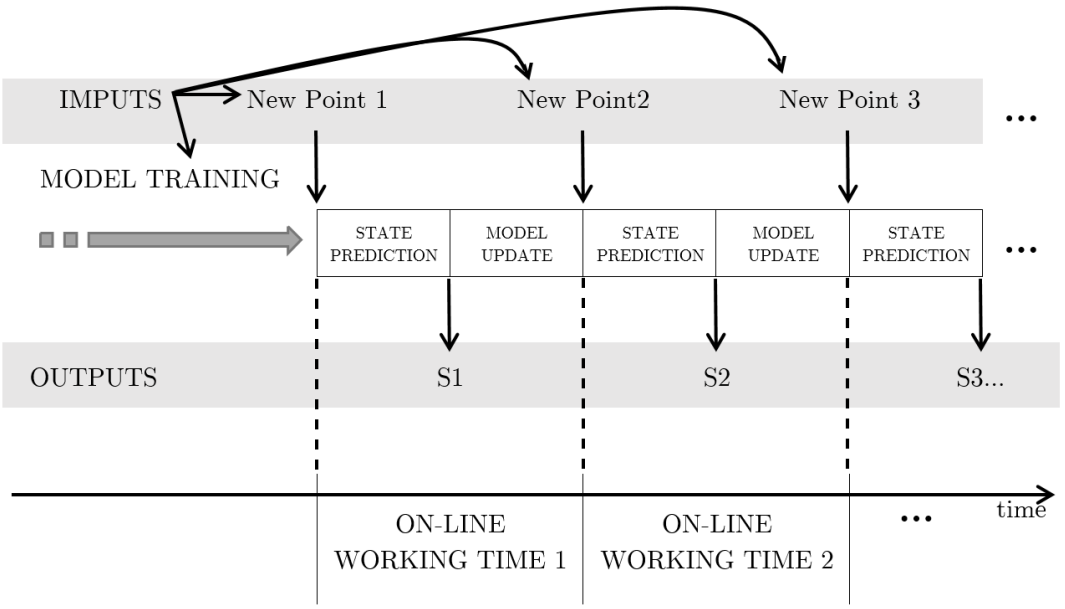


FIGURE 5.1: Online setting processing data.

5.1 Markov Models 3 Problems

Once a Markov model is built and the parameters $\lambda = A, B, \pi$ are already learnt, they can be used for more than calculating the joint probability and junt predict the current state. In fact, there are three specific tasks that these models can perform by using recurrent algorithms to alleviate the complexity of the operations to be performed. These are commonly referred to as the three problems of interest [15].

For a previously trained model λ, the so called three problems of interest are:

- The Decoding Problem: For a given observation sequence $O = o_1, o_2, ..., o_N$, we want to calculate the most likely state sequence $Q = q_1, q_2, ..., q_N$ associated to those observations. This is done by using the Viterbi algorithm.
- The Evaluation Problem: We want to calculate the probability of a new observation sequence $O = o_1, o_2, ..., o_N$. This can be expressed as the conditional probability of $p(O|\lambda)$ and can be done using the Forward-Backward algorithm.
- The Learning Problem: We focus on updating model parameters $\lambda = A, B, \pi$ when using a new set of observations $O = o_1, o_2, ..., o_N$. To complete this we use the Baum-Welch algorithm.

5.1.1 Viterbi Algorithm

The Viterbi algorithm can be used to calculate the most likely sequence of states given a sequence of observations iteratively. However, this algorithm need to be used alongside a batch learning approach to make the sequences predictions.

The need for these algorithms, comes from the idea that the best immediate state (the state that maximises the joint probability at some specific time) might not be the best option if we also observe what is the next most probable states. As HMM models work by using a transition probability, sometimes it happens that an particular lower probability at a determined time, maximises the whole sequence once all the data has been observed. The Viterbi algorithm uses the whole state sequence to find the most likely sequence, by studying a whole observed sequence of observations. In order to facilitate computations, we define an auxiliary variable

$$\phi_t(i) = \max_{q_1,q_2,...,q_{t-1}} p(q_1, q_2, ..., q_{t-1}, q_t = i, o_1, o_2, ..., o_t|\lambda) \tag{5.1}$$

as the highest joint likelihood that observations sequence and states up to time $t = t$ can have when the current state is i. Therefore, we can extend this by

$$\phi_{t+1}(j) = b_j(o_{t+1})[\max_{1\leq i\leq N} \phi_t(i)a_{ij}], 1 \leq i \leq N, 1 \leq t \leq T-1 \tag{5.2}$$

where the initial calculation,

$$\phi_1(j) = \pi_j b_j(o_1), 1 \leq i \leq N. \tag{5.3}$$

We can observe that the procedure to calculate ϕ_t is by recurring its value while choosing the maximum probability at each time step. Once this is completed, state $j*$ can be calculated using

$$j* = arg \max_{1\leq i\leq N} \phi_t(j), \tag{5.4}$$

and back tracking the most likely sequence.

5.1.2 Forward-Backward Algorithm

The Forward-Backward (FB) algorithm is used to calculate the probability of being in a determined state from time $t = 1$ to $t = T$. This is achieved by recursively calculating the parameters known as α and β.

The forward α variable represents the probability of an observed sequence $O = o_1, o_2, ..., o_N$, ending at state i. It is expressed by

$$\alpha_t(i) = p(o_1, o_2, ..., o_t, q_t = i|\lambda) \tag{5.5}$$

which is calculated

$$\alpha_{t+1}(i) = b_j(o_{t+1}) \sum_{i=1}^{N} \alpha_t(i) a_{ij}, 1 \leq i \leq N, 1 \leq t \leq T-1 \tag{5.6}$$

and the initial calculation

$$\alpha_{t+1}(i) = \pi_j b_j(o_1), 1 \leq i \leq N \tag{5.7}$$

All this recursions are intended to calculate the final α variable:

$$\alpha_T(i), 1 \leq i \leq N \tag{5.8}$$

which can be used to calculate the probability of an observed sequence given the model $p(O|\lambda)$ by

$$p(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i). \tag{5.9}$$

Similarly, the backward variable $\beta_t(i)$ is also the probability of an observed sequence $O = o_1, o_2, ..., o_N$. But this time, instead of calculating from time $t = 1$ to $t = t$, we calculate from $t = t$ to the final observed sequence data point at $t = T$. This can be expressed as:

$$\beta_t(i) = p(o_t + 1, o_t + 2, ..., o_T|q_t = i, \lambda) \tag{5.10}$$

Again, recursions allow us to do these calculations easily

$$\beta_t(i) = \sum_{i=1}^{N} \beta_{t+1}(i) a_{ij} b_j(o_{t+1}), 1 \leq i \leq N, 1 \leq t \leq T-1 \tag{5.11}$$

where the initial calculation at time $t = T$

$$\beta_T(i) = 1, 1 \leq i \leq N \tag{5.12}$$

We note that

$$\alpha_t(i)\beta_t(i) = p(O, q_t = 1|\lambda), 1 \leq i \leq N, 1 \leq t \leq T-1 \tag{5.13}$$

As with the forward example, $p(O|\lambda)$ can also be calculated using the β variable:

$$p(O|\lambda) = p(O, q_t = 1|\lambda) \sum_{i=1}^{N} \alpha_t(i)\beta_t(i) \tag{5.14}$$

5.1.3 Baum-Welch Algorithm

The Baum-Welch (BM) algorithm is used to update the model parameters based on the forward α and backward β variables. We introduce two new variables ξ and γ, which will be calculated using the already calculated FB variables. The first BW variable ξ defines the probability of being in a state i at time t and moving to state j at $t = t + 1$. The equations

$$\xi_{i,j} = p(q_t = i, q_{t+1} = j, O|\lambda) \tag{5.15}$$

which equals to

$$\xi_{i,j} = \frac{p(q_t = i, q_{t+1} = j|O, \lambda)}{p(O|\lambda)} \tag{5.16}$$

This can expressed in terms of α and β,

$$\xi_{i,j} = \frac{\alpha_t(i) a_{ij} \beta_{t+1}(o_{t+1})}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} \beta_{t+1}(o_{t+1})}. \tag{5.17}$$

On the other hand, the variable γ is used to calculate:

$$\gamma_{i,j} = p(q_t = i|O, \lambda) \tag{5.18}$$

This second variable can also be calculated using the FB,

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{5.19}$$

Finally, the relationship between γ and ξ is:

$$\gamma_t(i) = \sum_{i=1}^{N} \xi_{i,j}. \tag{5.20}$$

5.1.4 Traditional Parameter Update

Given the previous algorithms and auxiliary variables, the traditional method for parameter update would be as follows:

Priors Π_i:

$$\pi_i' = \gamma_1(i); \tag{5.21}$$

Transition a_{ij}:

$$a_{ij}' = \frac{\sum_{i=1}^{T-1} \xi_{i,j}}{\sum_{i=1}^{T-1} \gamma_{i,j}}; and \tag{5.22}$$

Emissions $b(v_k)j$:

$$b_j(k)' = \frac{\sum\limits_{o_t=V_k}^{T-1} \gamma_{i,j}}{\sum_{i=1}^{T-1} \gamma_{i,j}} \tag{5.23}$$

5.2 The Proposed Online Incremental Learning DHSMM Model

Built upon our Online DHSMM model, we have developed a novel incremental learning approach by performing the following operations:

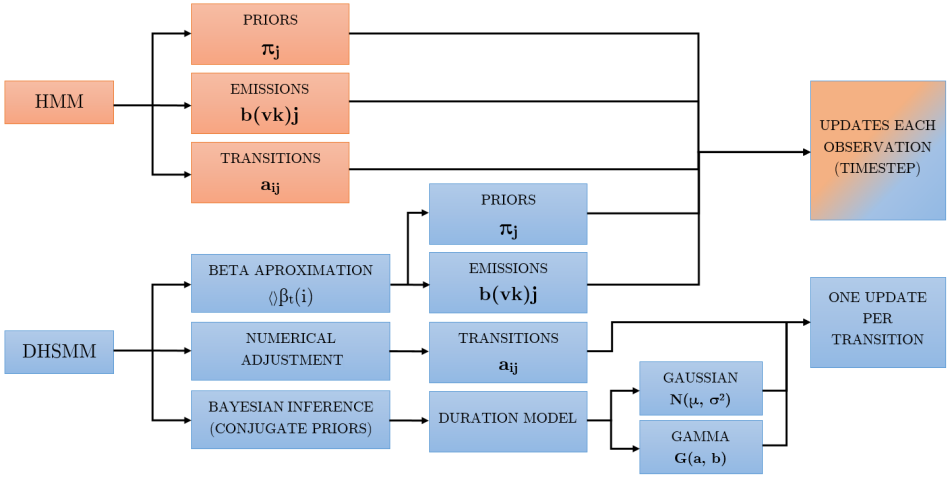


FIGURE 5.2: Model update HMM vs HSMM.

1. We calculate the β using an approximation technique based on a slidHMM approach [98], which allows us to calculate its value by using only the current single observation o_t instead of the usual o_{t-1}.

2. Using the approximated β, we can calculate ξ and γ variables using the traditional BW algorithm which allow us to update the priors and the emissions.

3. Due to the semi-Markov properties of our approach, to update the transition parameter, we use a numerical adjustment based on a learning rate parameter.

4. Finally, we update the duration model by adjusting the parameters of the probabilistic functions used to fit the state durations (Gamma or Gaussian) using Bayesian inference methods based on conjugate priors.

A graph to visualise how parameter updating is performed for HMM and for DHSMM can be seen in Fig. 5.2. To update traditional HMM models, we can use the BW algorithm to update priors, emissions and transitions. For the incremental online DHSMM, we need to use β approximation value in order to perform the BW algorithm operations, which will be used to update emissions and priors. For the transitions we use a numerical approach and for the durations we rely on Bayesian inference techniques. While HMM is updated each timestep, due to the semi-Markov properties of our approach, transitions and durations are updated once per state transition.

5.2.1 β Parameter Approximation

The main problem when doing these updates in an online setting is that all of them require information from one step ahead in time. Particularly, the variable β, which uses the observation o_{t+1} to calculate its value in time t. To address this issue, we have followed the idea in [98] so we can obtain an approximation of the β parameter without future data. This approach is called slidHMM and so far it is only been evaluated used for HMM models. Therefore, we have adapted this idea to work in an online setting for DHSMM approach, which only uses a single new observation at time t.

The approximation is based on the premise that each $\beta_t(i) \approx \beta_t(j)$. Therefore:

$$\beta_t(i) = \sum_{i=1}^{N} \beta_{t+1}(i) a_{ij} b_j(o_{t+1}), \tag{5.24}$$

would approximately equal to:

$$\beta_{t+1}(i) = \frac{\beta_t(i)}{\sum_{i=1}^{N} a_{ij} b_j(o_{t+1})}. \tag{5.25}$$

As can be seen, approximating the β values for all states allow us to approximately calculate this variable using only previous data. Having solved this issue, we can use this updated β to recalculate ξ and γ and perform the updates according to the traditional methods discussed early. The two parameter updates that can benefit from this approach are the priors and the emissions which are conducted as follows:

Using this approach we can update the priors and emissions as follows:

5.2.1.1 Priors Model Updating

Priors π can be updated following the traditional HMM approach by plugging the β approximation value into the formula to obtain the BW variables ξ and γ.

5.2.1.2 Observation Model Updating

In the case of the B observation parameter, we follow the same rationale as with the priors, but we also need to adapt the update to our DHSMM approach. Specifically,

DHSMM works by including different inputs each one corresponding to a sensor signal. However, the traditional HMM updating gives you an update for the whole observation probability (which includes all sensor signals). To address this, we calculate the difference between the emissions probability $b_j(k)$ and $b_j(k)'$ and we distribute that difference between the sensors that have actually been triggered at that time (see Fig. 5.3).

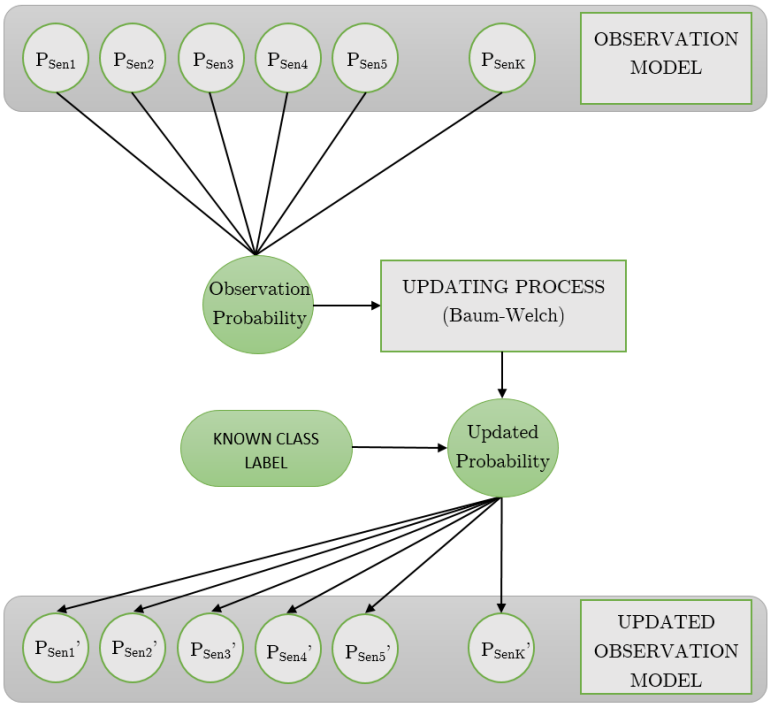


FIGURE 5.3: Observation update.

5.2.2 Transition Model Updating

There is a limitation when incrementally updating the transition model a_{ij}. While is fine to use this technique when a new batch sequence is feed, this parameter cannot be efficiently updated using the ξ and γ variables when we only include one new sample at a time. This is due to two reasons:

1. The fact that we receive a new sample each time does not mean we receive a new transition (in a HSMM framework, transitions happen after a number of samples) so we do not update transition each time step.

2. The fact that transitions are less frequent that samples make not efficient the frequentist calculations in which the BW algorithm are based.

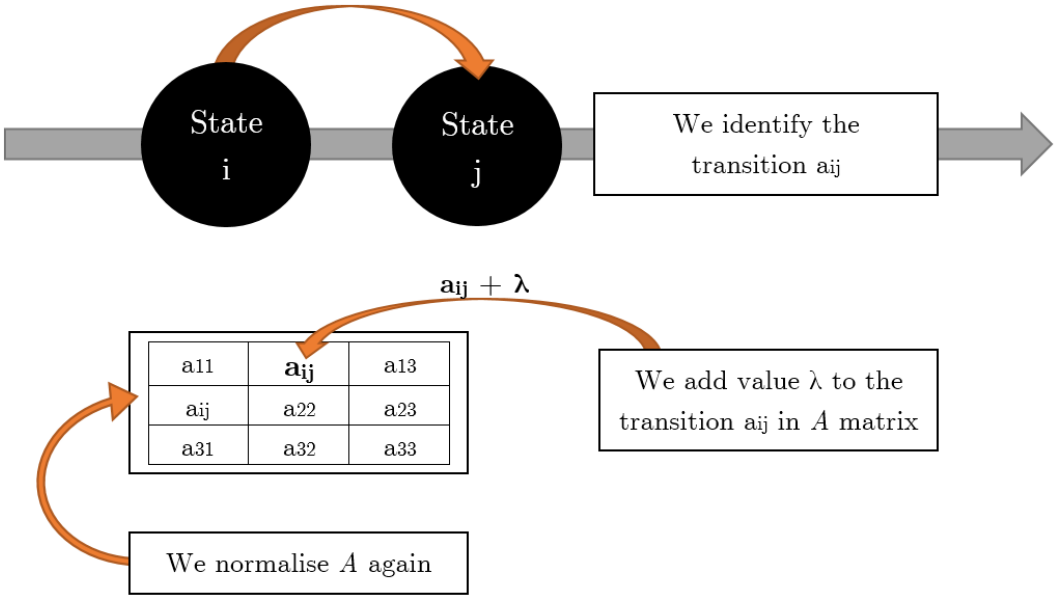


FIGURE 5.4: A transition model updating process.

Therefore, to update this parameter we apply a numerical approach using a learning rate parameter λ, which is applied to the transitions that has been newly sampled from the streamed data. As can be seen in Fig. 5.4, this process consists in adding the value of learning rate λ parameter to the transition parameter and re-normalise the matrix again. The steps are: 1) We identify the transition a_{ij}, 2) We add the value of λ to that particular transition in the transition matrix; 3) We normalise A.

There are different criteria to determine how to calculate λ [99][100].The learning rate parameter is calculated defining how many updates we need to change completely the values of this parameter. For example, if the probability of going from i to j is 1, and we want to reduce it to the opposite value of 0, we need to define how many steps we want this to take. If we choose a λ parameter of 0.001, we would need 1000 steps to move from 0 to 1 while if the parameter is set to 0.01, 100 step would be necessary to do this change.

In our experiments, we have chosen a parameter $\lambda = 0.01$ as we want to to observe actual changes in the model and also this is small enough to require a good number of updates to make significant changes. It is important to note that, although the patterns might change in the data, not necessarily are always towards the same values. In fact, in practice, these new values tend to hoover around some determined values instead of showing significant shifts towards values significantly distant from the original ones.

5.2.3 Duration Model Updating

The duration model also cannot be updated every time a new point is processed. Instead, this update is made effective once the current state ends (so a new whole duration can be added). The algorithm keeps track of the number of steps the state was occurring and uses this duration value for the updating process.

The duration updating has to be performed over the parameters that define the probabilistic function the model each state duration. For example, if the new duration value belongs to an state which duration we modelled using a Gaussian distribution, the update will be performed over the two parameters namely mean and standard deviation.

Here, we have developed an approach to update two of the most common and suitable distribution functions that we can use to model our states durations: Gaussian and Gamma distributions.

5.2.3.1 Gaussian Distribution

We use this distribution as is one of the most common probabilistic functions and it can be fitted easily even when there is not much data available. The Gaussian pdf is represented by:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{5.26}$$

where x is one sample and its probability is parametrised by the mean (μ) and standard deviation (σ).

In order to update these parameters, we apply the conjugate Bayesian analysis approach for these type of distributions [101]. Using the conjugate priors, we can estimate that the parameters posteriors after having included a single new point can be calculated as follows:

If we have an initial Gaussian distribution of the form

$$X \sim \mathcal{N}(\mu_0, \sigma_0) \tag{5.27}$$

where X refers to the expected duration of a occupancy state or activity duration. We can express the updated parameters after including a new value x by

$$\mu|x \sim \mathcal{N}\left(\frac{\sigma_0^2}{\sigma^2+\sigma_0^2}x + \frac{\sigma^2}{\sigma^2+\sigma_0^2}\mu_0, \left(\frac{1}{\sigma_0^2}+\frac{1}{\sigma^2}\right)^{-1}\right) \tag{5.28}$$

where this x is a new value in the dataset and we want to update the parameters to accommodate this new value.

5.2.3.2 Gamma Distribution

The gamma distribution is also one of the most well known probability distributions. We include this function because from previous analysis we could check that it was one of the most successful fitted functions when fitting the data from states durations. The Gamma pdf is represented by:

$$\frac{\beta^\alpha}{\Gamma(\alpha)}x^{\alpha-1}\mathrm{e}^{-\beta x}\mathrm{I}_x(0,\infty), \quad \text{where } \alpha>0 \text{ and } \beta>0 \tag{5.29}$$

In this occasion, the parameters to model are α and β parameters. To update these parameters, we apply a conjugate approach as well, but including the following considerations:

If we have a Gamma distribution such as that

$$G \sim \text{Gamma}(\alpha_0, \beta_0) \tag{5.30}$$

We can left the β parameter fixed, and we can update the α parameter from β and the new x point

$$\alpha' = \alpha_0 + (\bar{x_0} - x) \tag{5.31}$$

$$\beta' = \beta_0 \tag{5.32}$$

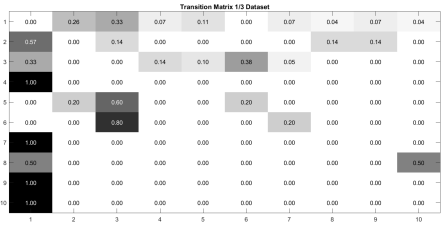
Where $\bar{x_0}$ is the median of the previous training data.

5.3 Update Analysis

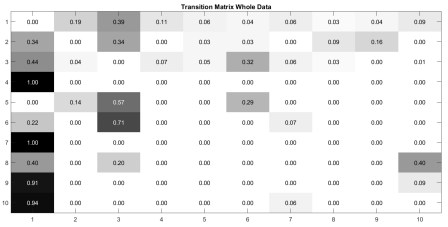
In order to evaluate the updating methods, we run a series of experiments focusing on the evolution of the parameters while are being updated. We have based these experiments on the study of the transition model and the duration models.

5.3.1 Transition Updating Analysis

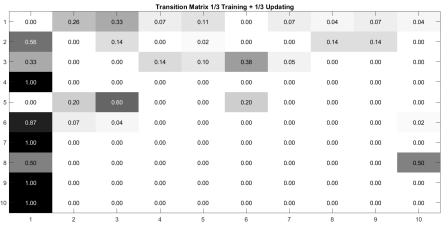
A transition matrix a_{ij} shows the probability of transitioning from state i (rows) to state j (columns). We have used the House A from the Dataset 1 used in the experiments in Chapter 3 to see how the transition parameter accommodates new data. We firstly trained a transition model using a third of the data (see Fig 5.5, picture a)) and we updated the parameter point by point to compared it with a transition model obtained using all the available data (picture b). It can be noted that a) is indeed different from b) (used offline 1/3 and 3/3 of the data respectively). Using a) as an starting point, we update the model and we see in c) how with an extra 1/3 of the data there is an improvement but not enough. Finally, d) shows the final transition matrix after having been updated with the 2/3 of the data. In an ideal situation, the matrices in b) and d) should be almost identical which they are not. However, we can note that the probabilities starting in a) move towards the figures in b) in most of the cases, which indicates that the values are moving on the right direction. Ideally, the best case scenario would give similar results for both batching and incremental learning. Although not identical, the results obtained here are likely to improve should the available data was larger (only over 400 transitions are available here). However, these are good enough to consider them promising, since most of the values improve the resemblance between the batch training and the one accommodating the updated values (similarity between Fig 5.5 b) and d)).



(A) Batch: one third.



(B) Batch: all data.



(C) Batch: one third. Increm: one third.

(D) Batch: one third. Increm: two thirds.

FIGURE 5.5: a) 1/3 Batch data. b) All Batch data. c) 1/3 Batch + 1/3 Incremental d) 1/3 Batch + 2/3 Incremental.

5.3.2 Duration Updating Analysis

In order to visualise how parameter would behave when including new data points we have used only the absence durations to fit a Gamma function with a fraction of the data while including the rest of the data sequentially and updating the model parameters iteratively. Using the dataset from Chapter 4, we have fitted a Gamma function with occupant daily arrival times using a third of the data and plotted the PDF. We then incrementally update the parameters using the rest of the data. Fig. 5.6 shows how the distribution shape changes slightly as the new data is also belonging to the initial distribution. The green line shows the last updated distribution and the black one shows the distribution obtained using all the data from the beginning. We can see that the updated distribution clearly adapts towards the one obtained with the full data.

5.4 Experimental Evaluation

In pursuance of an evaluation of the benefits of our incremental learning approach, we have conducted a series of experiments to compare how the different parameters change using our "incremental" vs a "non-incremental" approach (non-incremental refers to retraining or rebuilding the model keeping all previous values). To this purpose, we have used the same dataset used in the online model evaluation presented in Chapter 4.

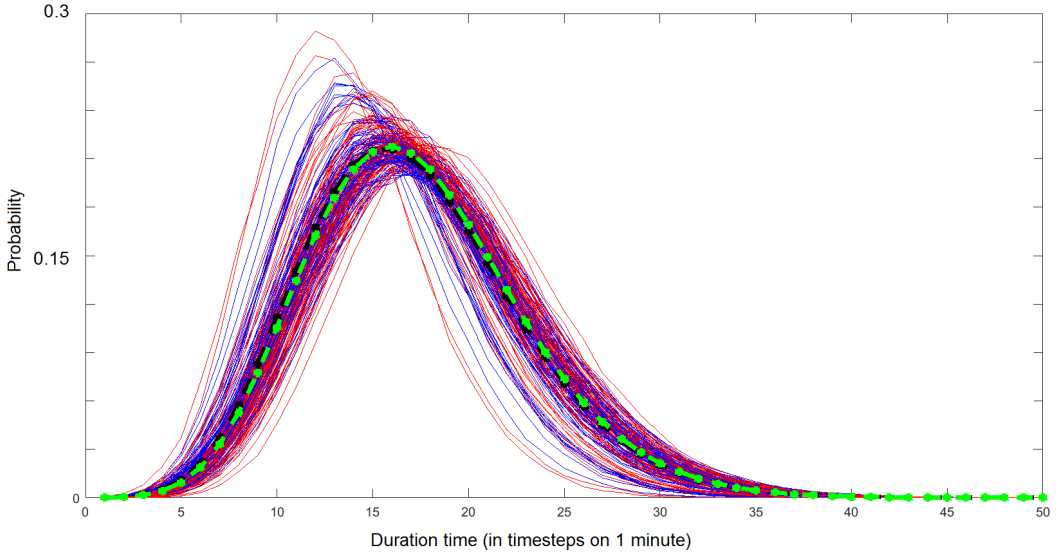


FIGURE 5.6: Gamma parameters updating.

5.4.1 Experimental Setup

We have used the D1 used for Chapter 4 experiments. As this dataset contains only two occupancy states as labels (absence and presence), the transition matrix does not need to be updated (always 0 for self transitions and 1 for absence to presence and vice-versa). Similarly, as we only use the prior model at the start we do not have to update that parameter either. We thus present an evaluation of the observation model, the duration model and the results achieved. We have also evaluated the computational cost for both approaches so we can have an insight of the efficiency of the model in terms of speed and scalability.

We have conducted the experiments as follows:

- **Retraining Approach**: We initially trained the model using a third of the data. Later, we processed the second third of the data adding one new sample at a time and retraining the model each time using both the old data and the new datapoint. We finally using the last third of the data to test the model with the current parameters.
- **Updating Approach**: We also initially trained the model using a third of the data. We then processed one point at a time but updating the model incrementally (we only keep parameters, all previous datapoints are discarded). Finally, again we use the last third of the data for testing the updated model.

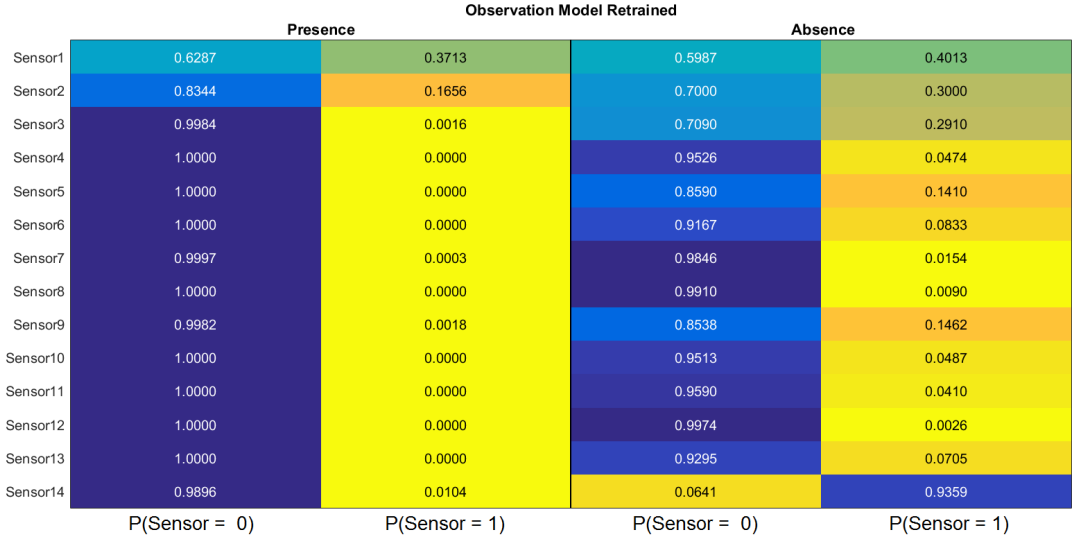


FIGURE 5.7: Final observation model after using the retraining approach. Each pair of columns on the left give the probability of Sensor1-Sensor14 being OFF(0) or ON(1) when each state occurs.

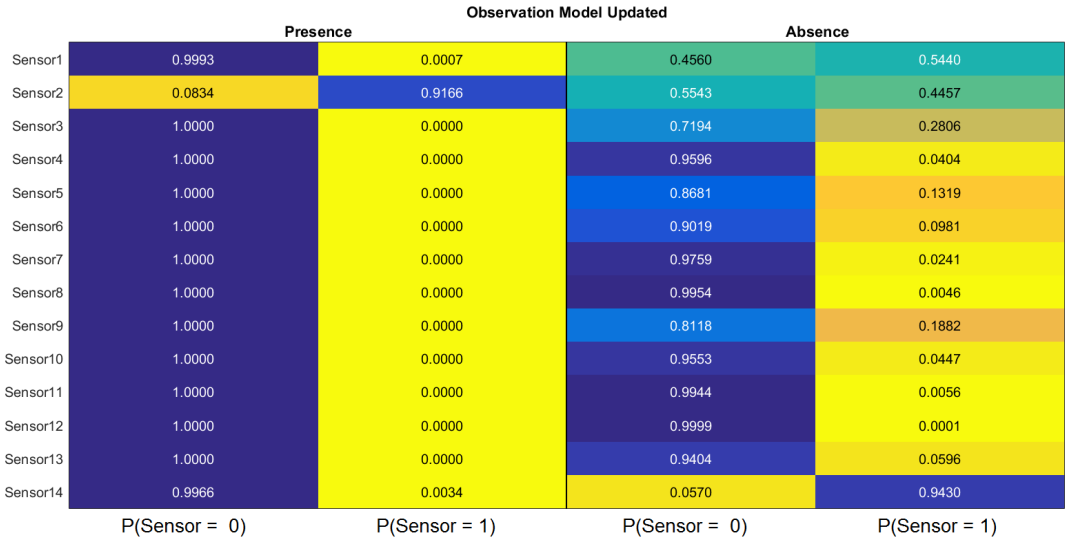


FIGURE 5.8: Final observation model after using the updating approach. Values are similar to the one obtained with the retrainings.

5.4.2 Observation Model

In this dataset we have 14 sensors and 2 possible states. The observation model shows the probability of each sensor being triggered by a determined state. In Fig. 5.7, we see the observation model obtained by retraining the model, whereas in Fig. 5.8 shows the observation model after updating the model. Although slightly different, both seem to have captured which are the most significant sensors for each state. The high probabilities for 0 values are common for sparse datasets.

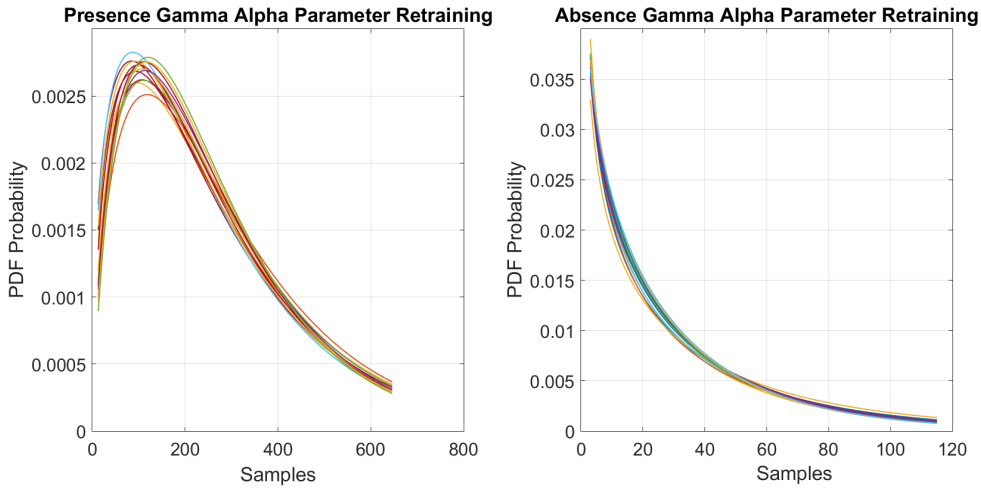


FIGURE 5.9: The curves on the left represent the evolution of the pdf for the presence state while retraining the model. On the right, the evolution for the absence state.

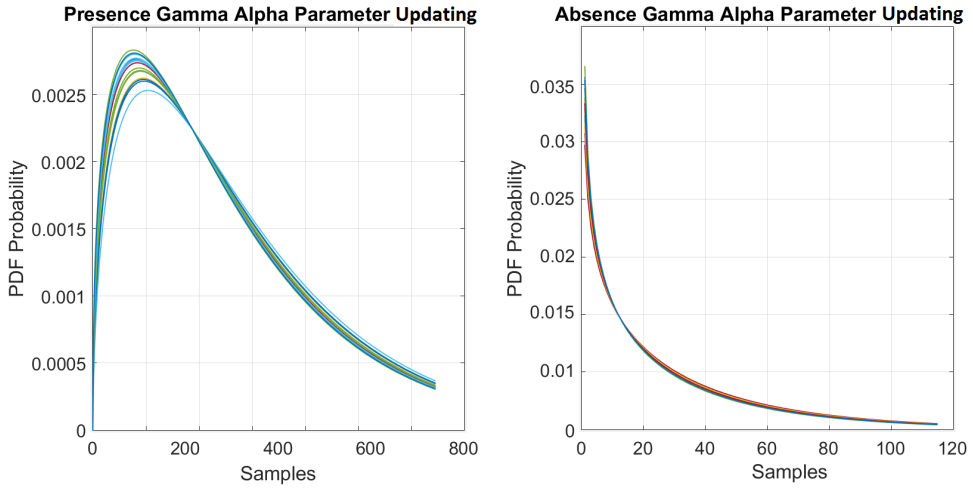


FIGURE 5.10: Similar to Fig. 5.9, here we can see how the pdf changes while updating the model.

5.4.3 Duration Model

Both states (presence and absence) durations were modelled using Gamma functions. Therefore we use the Gamma updating method as discussed in previous sections. In Fig 5.9 we can see how the pdf function evolves when the model is retrained. On the other hand, Fig 5.10 shows the evolution of the function by using our updating method instead of the retraining. As was the case with the observation model, although the final values are slightly different, they are similar enough to produce similar results.

In Fig 5.11 we can additionally monitor how the alpha parameter for both states evolved with each retraining/updating depending on the approach. It can be noted different

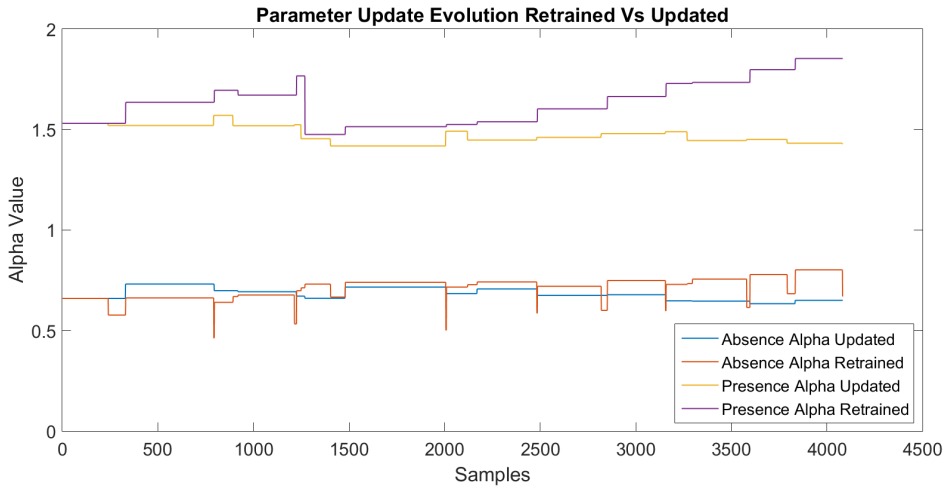


FIGURE 5.11: The two curves above show how the presence α parameter evolved. Final values are not different as expected. The two curves below belong to the absence α, again similar values for updating and retraining.

paths but again we can see a similar final behaviour, specially with state absence where the value alpha for both approaches is almost identical.

5.5 Results

So far, we have been able to verify that either retraining the model using all the data each timestep or updating the parameters using only a new point to learn incrementally, the models produce are initially fairly similar. The experimental results also confirm that the models are similar and accurate enough as the retraining approach achieved a 97% of accuracy (similar to the results in Chapter 4 experimental evaluation) and the updating approach obtained an almost identical 96% of accuracy. This similarity can be visually validated when plotting both states predictions against the ground truth (see Fig. 5.12).

This slightly lower accuracy results are due to the fact that the incremental approaches are learning the parameters 'slower', therefore they perform slightly poorer compared to batch learning. However, the differences are minimal and the results are still significant.

In addition to accuracy, it is important to attend to operational time. As it can be seen in Fig. 5.13, not only the operation time for the updating is substantially lower (a cent of a second versus roughly a tenth of a second). Moreover, as the updating is always done by including a new point, the time it gets to perform the operations do not

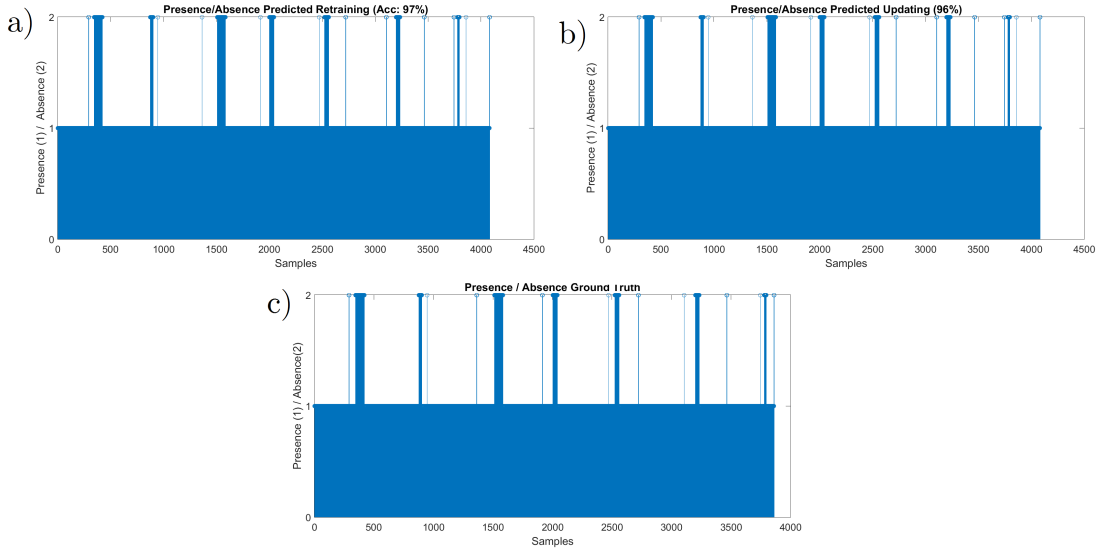


FIGURE 5.12: Both approaches achieve high standards of accuracy.

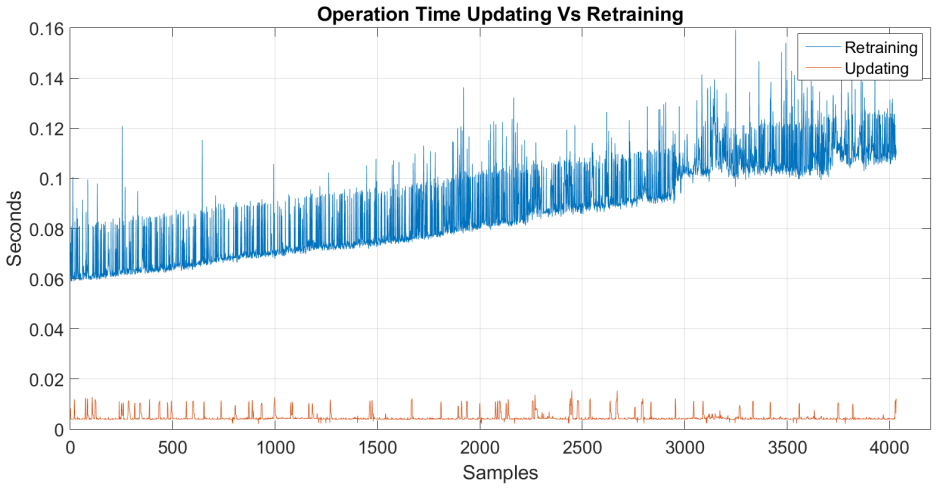


FIGURE 5.13: The time needed to retrain increases and is significantly higher that the updating approach. Updating approach is scalable over streaming data.

vary. However, when retraining the model the number of data used increases (unless a memory size or number of samples constraint is imposed) and therefore the time to perform the retraining increases accordingly. Therefore, although the accuracy results are similar, the computational time is much more favourable to the incremental learning approach.

5.6 Discussion

Based on the experimental results, we can see that there is a small decrease in overall classification accuracy when comparing the incremental learning with the batch learning. This is expected due to the fact that in this setup where the data is similar at all times, the batch learning parameters get the best possible parametrisation straight forward, while the incremental learning options seem to learn the parameters accurately enough but slower. That would explain the slightly differences reported.

In terms of scalability, even though our datasets were not very big, an important increase in computational cost can be noted from the reported results when we incorporate new data by retraining the model. Due to our dataset not being too big, it only took a tenth of a second to retrain the model. As we are using a 60 second time granularity, we have more than enough time to perform the model retraining operations between each time step. However, for datasets with a lager number of samples/features, retraining time would increase and could become a potential issue. This issue becomes critical when we have limited time to perform all operations. Due to all these reasons, we can argue that our model overcomes this limitation as the processing time remain constant. Therefore, we can conclude that our model is scalable to large datasets.

Summarising, in contrast to non-incremental learning/traditional batch learning approaches, our approach significantly improves the efficiency of the model by reducing the time and the amount of memory necessary to store old data points as those are discarded. The incremental learning DHSMM approach not only improves model efficiency but also almost matches the high levels of accuracy while making the model significantly faster and scalable.

Chapter 6

A Novel Framework For Multi-Occupant Modelling Using A Dynamic Hidden Semi-Markov Model Approach

Most of existing OBPM approaches focus on single occupant only. There is little work done on multiple occupants. In this chapter, we present a novel model framework based on DHSMM for multi-occupant behaviour pattern detection, consists of single-layer and multi-layer approaches.

The following sections will detail the existing works, challenges on multiple-occupant behaviour pattern modelling and our proposed approach and experimental evaluation.

6.1 Current Work

Multi-occupant pattern detection has been recognised as one of the main challenges for OBPM models [23]. For example, the work in [102] aims to detect occupancy to regulate HVAC systems in buildings. Using a variety of techniques such as regression or stochastic models, they attempt to establish occupancy profiles of multi-occupant based upon sensor data and time+date information to asses the impact of each of these more

realistic profiles on the building energy performance. A similar approach is taken in the work in [103], where they used a multi-agent system to simulate occupancy interactions based on thermal condition variations to study the effects of occupancy presence over the building energy performance. In spite these efforts, it can be concluded from their results due to the lack of real data, which limits these works to the use of synthetic data with the subsequent issues to validate and escalate the results reported.

Some researchers have attempted to model multi-occupant activities based on two datasets: the ARAS dataset [2] and the CASAS dataset [64], limited to a maximum of 2 occupants. These two dataset have enabled series of works that have attempted to model multi-occupant activities although both limited to a maximum of 2 occupants. Both datasets have several different sensors deployed in smart homes environments and are used to infer up to 27 activities performed by 2 different persons at the same time sharing the same space. These works include the works presented in [104][105][106] or [107], where they used ARAS and CASAS datasets to infer occupant activities based on data-driven algorithms with no intrusive sensors.

Although addressing multi-occupant models additional work hs to be done to improve the current limitations. For example in the work in [104], they addressed milti-occupant models but they only used Naive Bayes and HMM models, which are limited for OBPM. Similarly, in [105] the author claimed having improved CASAS results but they limited their experiments to CRF and HMM models. Another model based in CRF was proposed in the work in [106], where they used different pre-processing techniques and data association techniques, athough this work was more focused on presenting a bechmark of results that introducing a novel technique. Finally, the work in [107] similarly used adapted CRF and HMM models claiming multi-occupant accuracies between 87% and 96%.

All these works used well-know traditional methods to improve the accuracy of previous reported methodologies. However, none of them attempted to study the implications of working in an online setting would have to be taken into account. Moreover, they never tried to perform any sort of incremental learning and the results reported never gave indications of the model efficiency, computational times or scalability of their models.

In spite these recent efforts, there are really few works that have attempted to model multi-occupant scenarios successfully in the context defined in this thesis. Moreover, so

far as we are aware there has not been any approaches that attempted to model multi-occupant patterns using explicit duration Markov models and processing the datasets in an online fashion with incremental learning parameter updating.

6.2 Main Challenges

We have identified three main factors that have a significant impact on the design and performance of multi-occupant models: association, interaction and data scarcity.

6.2.1 Sensor Readings Association

As discussed in the previous chapters, the sensors involved in this work are non-intrusive, which is an advantage in terms of intrusiveness and privacy issues. However, due to not including any specific personal data, detecting multiple occupants in the same space becomes a real challenge and it is difficult to know which sensor readings have been triggered by each one of the occupants. Ambient sensors such as the ones we ae considering (e.g. temperature, motion or CO_2) do not make any differentiations in their readings to be related to occupants unlike video, RFID or information from smart homes. For these models to be successful on their predictions, they have to be able to extract underlying separated patterns by processing the inputs as a whole. All these considerations are the reason why detecting multiple occupants remains one of the main challenges in spite of recent efforts.

In Fig. 6.1, we see that when there is only one occupant (left) it is relatively easy to assign each of the firings to a determined occupant (as there is only one). However, when more than one occupant is present in the same place at the same time (right), it becomes significantly harder to ascertain which signal corresponds to each occupant.

6.2.2 Occupant Interaction

There are cases where occupants can be performing the same activity at the same time or performing an action collaboratively. In these occasions, it also becomes really hard for OBPM models to make a successful differentiation from the states sequences belonging to each occupant as well. This becomes increasingly challenging due to the

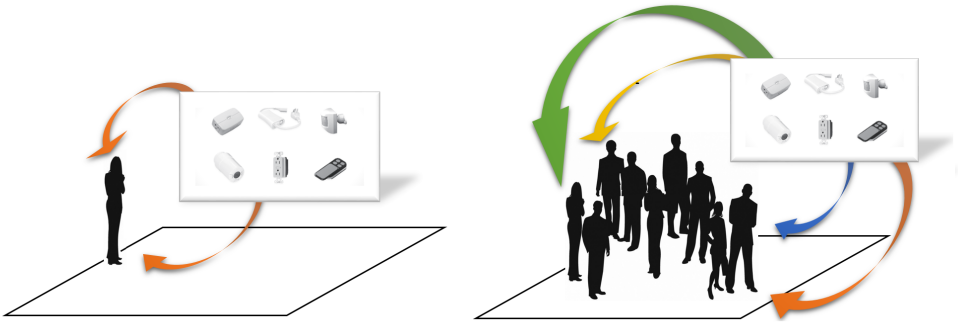


FIGURE 6.1: Association

fact that the triggering of physically close sensors do not necessarily imply there has been a real interaction between the occupants. Similarly to sensor readings association, the uncertainty about occupant interactions also contributed to harden the prospect of performing successful classification tasks in these scenarios.

In Fig. 6.2, we see that when occupants are performing well defined separated activities (left), algorithms have less problems to model each state accurately. When activities are being performed concurrently or collaboratively (right) it becomes much harder to model them properly and assign the according labels to each occupant.

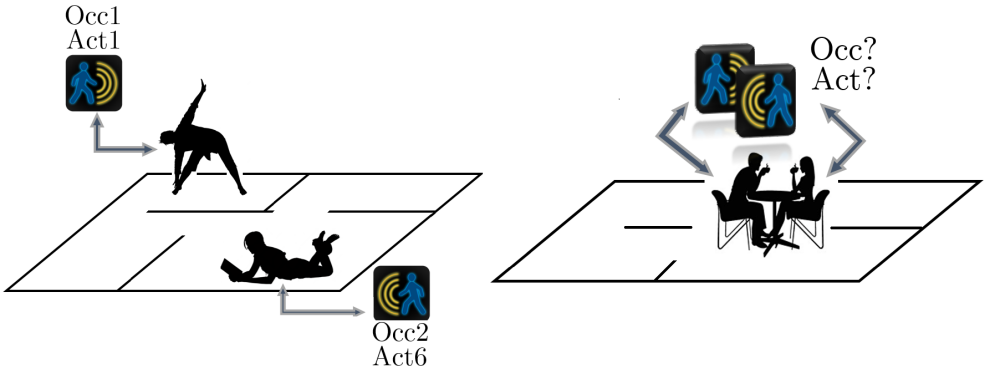


FIGURE 6.2: Interaction

6.2.3 Limited Publicly Available Data

The datasets available for multi-occupant modelling are really scarce. Due to this, most of the experiments conducted in existing works have been conducted using two of the most representative datasets: i.e. CASAS dataset [64] and ARAS dataset [2]. Due to this important limitation in available data, it becomes even more difficult to develop models with good robustness and scalability.

6.3 Framework For Multiple Occupant Pattern Modelling And Detection

Here we propose an online incremental learning framework to detect multiple occupants based on our DHSMM approach. We have conducted several experiments using the popular multi-occupant ARAS dataset [2].

6.3.1 Dataset Description

ARAS dataset contains data collected in two different houses (HouseA and HouseB) and include a total of 20 sensors and up to 27 different activities to be modelled (see Table 6.1).

These activities correspond to all the possibilities each occupant have. There are no activities that are performed by a collaboration of occupants, however any of the activities can be performed by any occupant at any time. This means that activities can be concurrent since can be performed by each activity at all times. Interleaved or single occupant concurrent activities are not present in this dataset.

The data is provided as a list which shows the date and time a sensors has been triggered, and the label and the occupant associated to that sensor firing as can be seen in Fig. 6.3. The sensors associated to each activity can be seen in Table 6.2. This dataset is divided into 30 days where data has been collected at different times. For our experiments we use a leave-one-out cross validation approach for the validation of results, which means that we train the model using one day of the data and we update the model incrementally with the rest 29 days of data.

6.3.2 The Proposed Methodology

Built upon our DHSMM approach, we have proposed two methods to handle the multi-occupant data, so our model captures the separated occupant patterns:

1. Single-Layered approach: a single-layer model is used to detect all occupants states producing only one output. Therefore, the activities of all occupants need to be encoded in the same label. For example, if we consider two possible states and

TABLE 6.1: ARAS dataset consists of up to 27 different activities or labels (left) and a total number of 20 sensors or features of diverse nature.[2]

Activity Explanations		Sensor Explanations			
ID	ACTIVITY	Column	Sens ID	Sens Type	Place
1	Other	1	Ph1	Photocell	Wardrobe
2	Going Out	2	Ph2	Photocell	Couch
3	Preparing Breakfast	3	Ir1	IR	TV receiver
4	Having Breakfast	4	Fo1	Force Sens	Couch
5	Preparing Lunch	5	Fo2	Force Sens	Couch
6	Having Lunch	6	Di3	Distance	Chair
7	Preparing Dinner	7	Di4	Distance	Chair
8	Having Dinner	8	Ph3	Photocell	Fridge
9	Washing Dishes	9	Ph4	Photocell	Kitchen Drawer
10	Having Snack	10	Ph5	Photocell	Wardrobe
11	Sleeping	11	Ph6	Photocell	Bathroom Cabinet
12	Watching TV	12	Co1	Contact Sens	House Door
13	Studying	13	Co2	Contact Sens	Bathroom Door
14	Having Shower	14	Co3	Contact Sens	Shower Cab. Door
15	Toileting	15	So1	Sonar Distance	Hall
16	Napping	16	So2	Sonar Distance	Kitchen
17	Using Internet	17	Di1	Distance	Tap
18	Reading Book	18	Di2	Distance	Water Closet
19	Laundry	19	Te1	Temperature	Kitchen
20	Shaving	20	Fo3	Force Sens	Bed
21	Brushing Teeth				
22	Talking on the Phone				
23	Listening to Music				
24	Cleaning				
25	Having Conversation				
26	Having Guest				
27	Changing Clothes				

```
2008-11-10      15:00:26.69822  M13     ON 2 15 1 14
2008-11-10      15:00:28.7381   M14     OFF 2 15 1 14
2008-11-10      15:00:30.352499 M14     ON 2 15 1 14
2008-11-10      15:00:30.76444  M13     OFF 2 15 1 14
2008-11-10      15:00:32.78852  M13     ON 2 15 1 14
2008-11-10      15:00:35.80133  M13     OFF 2 15 1 14
2008-11-10      15:00:36.63942  M14     ON 2 15 1 14
2008-11-10      15:00:38.418989 D12     CLOSE 2 15 1 14
2008-11-10      15:00:38.99253  D12     OPEN 2 15 1 14
2008-11-10      15:00:41.96863  M14     OFF 2 15 1 14
2008-11-10      15:00:43.215759 M14     ON 2 15 1 14
2008-11-10      15:00:45.04572  M21     ON 2 15 1 14
2008-11-10      15:00:45.995589 M21     OFF 2 15 1 14
```

FIGURE 6.3: ARAS Dataset, House A.

TABLE 6.2: Sensors, activity associated and location

Ambient Sensors	Actions / Location
Force Sensor or Pressure Mat	Sleeping, sitting, napping / Under the beds and the couches
Photocell	Opening the drawers and the wardrobes / In the drawers, the wardrobes and the refrigerator
Contact Sensors	Opening and closing of the doors, cupboards/ On door frames, shower cabin, cupboards
Proximity Sensors	Detecting close distance objects / On the chairs, on the closets and on the taps
Sonar Distance Sensors	Detecting presence / On the walls, door frames
Temperature Sensors	Cooking / Near the oven in the kitchen
Infrared Receiver	Watching TV / Near the TV

two occupants, the number of possible outputs will be $2^2 = 4$. (See Fig. 6.4). The single layer approach models each output as a combination of of activities. The output class will always be a combination of the activity each occupant is performing.

2. Multi-Layered approach: is based on the idea of using a multi-layered model. A number of models equal to the number of occupants is trained, so each model incorporates the labels of one of the occupants (See Fig. 6.5). In this case, we have separated and independent models one for each occupant, and they give separated outputs. However, both outputs are combined to evaluate them against the ground truth where we have also a separated class for each occupant.

6.4 Experimental Evaluation

For our experiments, we use the HouseA of the ARAS dataset, and we apply both the single-layered approach and the multi-layered approach. This dataset is actually made of 27 different activities, however we combine them into 6 larger groups of states per occupants. As explained in [1], the most frequent activities namely sleeping, eating, personal hygiene, going out and relaxing have been grouped together; the rest have been grouped into 'other'. As we have 2 occupants performing 6 different activities each, we

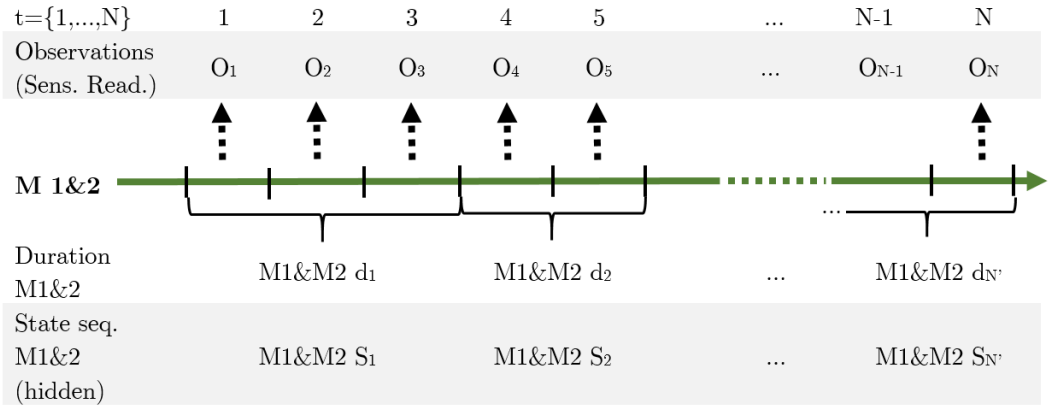


FIGURE 6.4: One model is used to perform two occupant activity detection. The labels are used in a combined way, and the output of the model is a class which encodes a combination of the activities each occupant is performing.

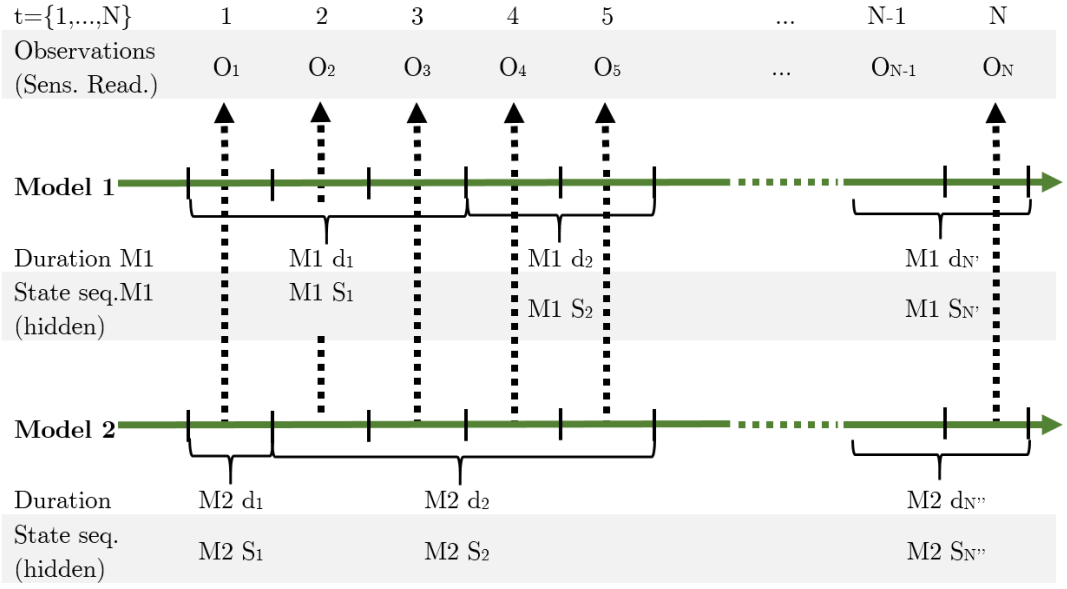


FIGURE 6.5: One layer of models is used for each occupant. The features are shared between all the models. However, each of the models are trained using the labels of one different occupant. Therefore, each of them detects the activity performed by their associated occupant.

can have 6^2 total number of combinations possible. We adopt this approach because is the one followed in [1] so we can have a baseline to compare the results they obtained against our own.

6.4.1 Single-Layered Approach With $6x6$ Activities (Combinations)

In this first experiment, we follow the combination of 6 labels per occupants using the single-layered approach. Here, we model each combination as a possible outcome or state of the system. As we have 36 possible combinations, the single-layered model has 36 states or classes. The numbers are from 11 (occupant 1 performing 1, while occupant 2 performing 1 as well) to 66 (both occupants perform activity 6). We use a single-layeres DHSMM to try to classify up to 36 different classes.

In Fig. 6.6 we can see the confusion matrix of all the predicted combined activities vs the ground truth. Each pair of numbers represent a combination of sequences. For example 12, means Occ1 performing activity 1 (Sleeping) and Occ2 performing activity 2 (Eating). In spite of having a large number of possible states, the model is able to predict correctly a big number of occurrences. The overall accuracy of our DHSMM approach is of 68.99%, which is over the results the dataset publishing team reported: accuracy of 61.5% using a HMM model.

6.4.2 Multi-Layered (2-Layers) Approach With $6 + 6$ Activities

In this occasion we have a separated set of outcomes. Fig. 6.7 and Fig. 6.8 show both confusion matrices for Occ1 and Occ2 respectively. We can see that the model performs significant better than other approaches not only in accuracy, but also in recall and precision values as well.

In order to further evaluate the multi-occupant framework, we repeat the process but on this occasion we use a multi-layered approach where two DHSMM models are use in order to model an occupant per model. We obtain an accuracy of 79.64% and 78.30% respectively for Occ1 and Occ2, and a total overall of 78.97%.

6.4.3 Multi-Layered (2 Layer) Approach With $27 + 27$ Labels

In this last experiment, we have used our DHSMM with the multi-layered approach, but on this occasion we have used the whole 27 activities for each occupant instead of combining them into only 6 as in previous experiments. The purpose of this last

TABLE 6.3: Compared with the baseline value of 61.5% of accuracy, our DHSMM approach achieves better results of accuracy, both using 6 and 27 activities. Multi-layered shows highest classification accuracy.

	Multi-Layer (27 Act)	Multi-Layer (6 Act)	Single Layer (6 Act)
Occupant 1	69.45%	79.64%	-
Occupant 2	69.22%	78.30%	-
Combined	69.34%	78.97%	68.99%

experiment was to analyse the robustness and adaptability of our model with a more complex representation of the data.

The accuracies in these experiments are 69.45% and 69.22% respectively for Occ1 and Occ2. Fig. 6.9 and Fig. 6.10 present the confusion matrix and shows how much more complex these models are compared to the ones used with only 6 activities.

6.5 Discussion

Based on the experimental results, we can conclude that the multi-layered approach is able to better capture the patterns in the data when we are modelling one occupant. In Fig. 6.11, we can see how the accuracies of the models detecting the occupants separately are performing better than the single-layer model detecting both occupants.

Furthermore, even if we use all 27 labels instead of the reduced way with only 6, we can see that the multi-layered approach still shows good performances (see Table 6.3).

Generally, our DHSMM improves the results reported in previous works and shows that is able to handle large volumes of activity data in an efficient way even with the challenging setup of working in an scenario including multi-occupant data in an online setting with incremental parameter update.

Using a reduced number of labels, our DHSMM based model shows that can reach values close to 80% of accuracy using a multi-layered approach and close to 70% when using a single-layer model. Both results are significantly higher that the results obtained by the dataset publishing team in their experiments [1], where they obtained an accuracy of 61.5% using a similar single-layered HMM approach and the same number of reduced labels (from 27 to 6). Furthermore, we have extended our experiment using the whole

set of labels available and showed that our DHSMM model improves that result with a multi-layered approach and 27 activities per occupant.

The experimental evaluations show that our approach can perform activity recognition in an online setting using data from multiple occupants while overcoming the traditional limitations present when using online HSMM-based models.

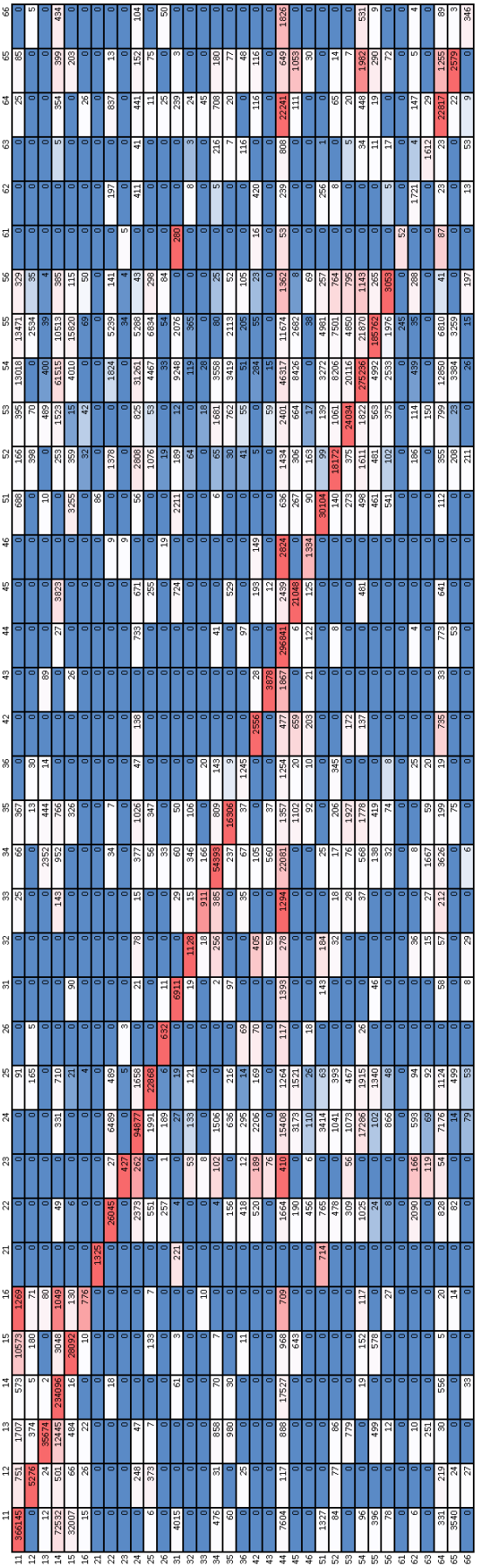


FIGURE 6.6: Confusion matrix of the single-layer approach with 6*6 (36) activities. The results are lower than the ones obtained using the multi-layered model, although slightly better than the ones reported by the dataset publishing team [1].

	Sleeping	Eating	Pers. Hygiene	Going Out	Relaxing	Other	PRECISION
Sleeping	803069	1016	2591	3849	106606	1696	0.87402
Eating	987	176859	2067	7254	74057	3822	0.66728
Pers. Hygiene	9408	2088	89362	3635	20699	1679	0.70435
Going Out	28747	23813	27852	324617	68407	25097	0.65114
Relaxing	8816	20182	4699	1846	637546	3147	0.94279
Other	2329	14051	7163	3808	46335	32801	0.30803
RECALL	0.94107	0.74308	0.66821	0.94089	0.66853	0.48066	

FIGURE 6.7: Confusion matrix of the multi-layered approach for Occ1 and 6 activities.

	Sleeping	Eating	Pers. Hygiene	Going Out	Relaxing	Other	PRECISION
Sleeping	507000	1454	6570	83822	54502	3252	0.77216
Eating	811	71496	3602	74980	24768	1355	0.4039
Pers. Hygiene	3871	404	77033	54675	11939	543	0.51886
Going Out	21856	6532	13661	1049010	47328	4400	0.91794
Relaxing	3822	3658	2108	43812	308812	374	0.85169
Other	2356	4087	4380	60390	17153	16184	0.1548
RECALL	0.93938	0.81588	0.71756	0.76756	0.66482	0.61989	

FIGURE 6.8: Confusion matrix of the multi-layered approach for Occ2 and 6 activities.

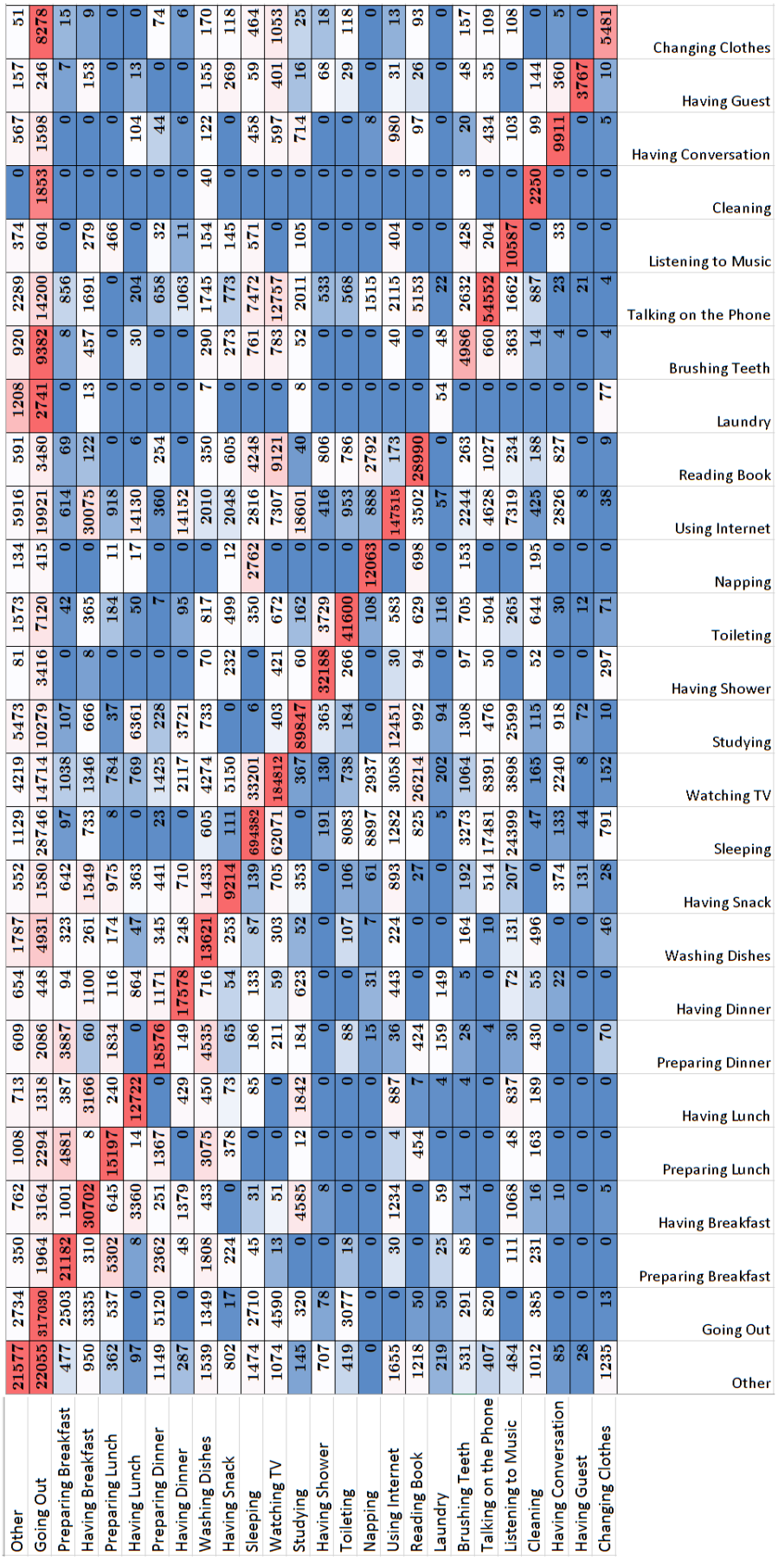


FIGURE 6.9: The confusion matrix shows how the activities are classified by the occupant 1 layer. Most of them are correctly classified (high diagonal values) in particular the most common ones (*going out* and *sleeping*) which means good overall performance for this model.

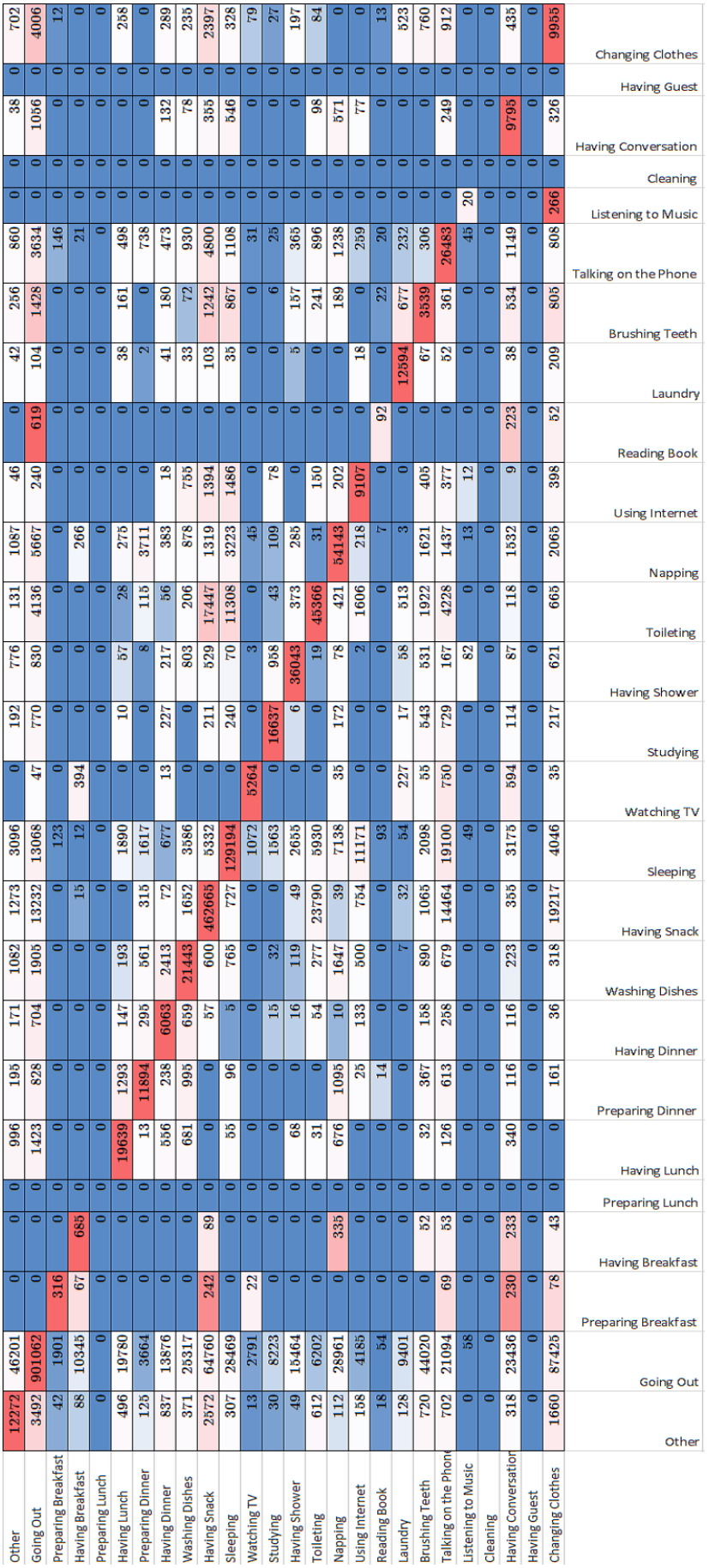


FIGURE 6.10: The confusion matrix for occupant 2 shows similar values compared to occupant 1, showing the good consistency of our approach.

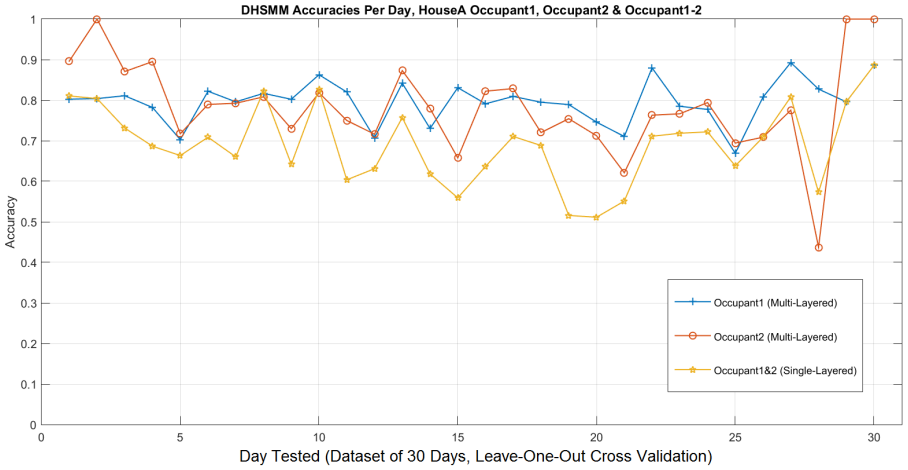


FIGURE 6.11: The multi-layered approach using one model per occupant performs better than the single-layered approach which combines the labels of both occupants.

Chapter 7

Conclusion And Future Work

In this thesis, we have introduced the concept of occupant behaviour pattern modelling (OBPM) and why this is relevant as a potential solution for buildings scenarios. The main objective has been the designing of an algorithm that could process data to extract real occupant information that could be used to regulate building systems according to actual occupant needs. In order to achieve this, we have identified potential limitations and challenges these particular modelling techniques suffer, especially when the systems are intended to perform operations in real-time through streamed sensor data. To overcome these challenges, we have studied potential solutions by evaluating different methods to compare their performance. Based on our findings, we have proposed a novel model that extends traditional approaches to capture the occupancy and activity information in real-time scenarios. Therefore, improving traditional levels of performance while providing a better solution in terms of scalability and model complexity.

7.1 Contributions

In this work, several goals have been successfully achieved towards the development of a novel data-driven approach to fast and accurately model and detect occupancy patterns for both single occupant and multiple occupants, using real time sensor data. The main achievements can be summarised as follows:

1. We have conducted a comprehensive survey on previous research related to Occupant Behaviour Pattern Modelling (OBPM) and different machine learning techniques currently adopted. We have focused our efforts on previous works addressing human behaviour pattern modelling in buildings using only non-intrusive sensors data (e.g. motion, contact, pressure,...). This study allowed us to gain greater understanding about the current state-of-the-art and identify the main methodologies used so far in the existing literature.

2. We have proposed a benchmark of experiments, where we have conducted a thorough comparison and evaluation using existing well-known machine learning techniques for OBPM based on publicly available real datasets. Here we used several metrics of performance evaluation such as accuracy, precision or recall in addition to processing time and model complexity. Additionally, to understand the factors that might have an impact over the overall model performance, we have investigated and identified other aspects and limitations that could affect the accuracy of the models, including different data pre-processing techniques. These experiments included classification of occupant states using several well-known machine learning techniques including k-nearest neighbours (kNN), hidden Markov models (HMM), hidden semi-Markov models (HSMM) and support vector machines (SVM). The objective was to evaluate the performance of different algorithms in a OBPM context to assess which approaches can be better suited for the task at hand. Additionally, we performed a series of supplementary experiments where we introduced different data pre-processing techniques (timeslice and chunk approaches) and variable sampling frequency.

   After analysing the results, we learnt that in HSMM-based models given this scenario, the sequential properties of the Markov models work well to model the sensor sequential data. In fact, HSMM outperformed HMM performance in terms of classification accuracy, while not incurring in prohibitive training times as happened with other models such as SVM. HSMM learned parameters, which include the traditional transition, observation and priors; are extended with a duration model that helps to capture states more accurately. However, there are some problems when modelling real-time scenarios due to HSMM needing whole data batches for modelling a state and the rigid methods when sampling the duration of the

next state. Consequently, the experiments showed that HSMM-based modelling is promising; however, it still suffers limitations as follows:

a) HSMM features observations contain just one value per sample, which means that there is no way to give more relevance to certain sensors (inputs) in scenarios were the physical deployment of the sensors favours the interaction of some of them over the rest. For example, a sensor in a central hallway will be more likely to be triggered that one located in a small corner of a less used room.

b) If a state is defined throughout all its duration, we need all the observations at the same time in order to estimate what state and for how long is going to occur.

3. We have developed a novel dynamic hidden semi-Markov model (DHSMM) to process online streaming data and make occupancy state predictions in real time. This included some of the most attractive properties of the HSMM models for sequential data analysis, but it was specifically designed to address the challenges identified when using HSMM models for OBPM purposes in an online setting. Our DHSMM extended the capabilities of traditional HSMM models in two ways:

   - The weighted observation model: we have fed the sensor observations into the model as separated inputs (one per sensor) and calculated the observation probability by aggregating all the signals together. Additionally, we have included a set of weights associated to each of the sensor signals based on a correlation function (MSE function) to help our model better represent the significance of each of the sensor signals involved.
   - The dynamic duration model: We have predicted each state duration based on an inverse cumulative distribution function, which we used to calculate the probability of remaining versus the probability leaving the state dynamically (each timestep). This was intended to improve HSMM traditional online performance, which bases the duration prediction on a random duration sample calculated at the beginning of each state. To validate our DHSMM approach, we included several experiments using datasets containing presence/absence occupancy data achieving good levels of performance.

- We have developed a novel incremental learning approach for our online DHSMM model, defining incremental learning as updating of model parameters without being allowed to store any of the previous samples (all previous samples are simply discarded). This is achieved by re-estimating parameters iteratively while new data is being processed by our system. As long as new sensor data is observed, new values can be assigned to the model parameters in order to incrementally incorporate the new information received. To do this, we have proposed several different novel approaches to update the various model parameters (i.e. priors, observation, transition and duration models):

  a) Observation and Priors: We have used a novel technique including a β value approximation calculated using the Baum-Welch algorithm using only one new sample of data. Using this approximation, priors and observations can be updated following traditional HMM framework parameter updating approaches.

  b) Transition: Due to the nature of HSMM approaches, we have introduced a numerical approach to the updating of the transition model, consisting in adding a learning rate value (λ) each time a specific transition took place.

  c) Duration: To update the parameters of the functions used to describe the duration model, we have used a combination of Bayesian inference techniques based upon conjugate priors, to stablish an updating methods for the parameters of distributions such as Gamma and Gaussian.

4. Finally, we have proposed a novel multi-occupant framework for OBPM models in order to perform occupancy and activity detection in scenarios where more than one occupant is present. We conducted experiments to validate out proposal by evaluating multi-occupant approaches including a method where each occupant is modelled by an independent model with common features (multi-layered incremental online DHSMM) and a second method where all occupants are modelled together by modelling the states as combinations of states belonging to each occupant (single-layered). The final experimental evaluation shows how our approach is

able to performs fast and accurate occupant behaviour pattern detection in an online setting using with data from various occupants and overcoming the traditional limitations present when using HSMM-based models.

7.2 Challenges

In this project, several challenges have been encountered throughout its development. In the initial stages, this project aimed to create an embedded software tool that could seamlessly be implemented in BEMS. This tool was meant to provide occupant and energy consumption information to study, not only human/building interactions but also the real impact these could have over the energy consumption of the buildings. HAving this data available, it would have enabled us to the development of solutions directly linked to the relationship between energy and occupancy.

To collect this data, our team studied several ways to deploy our own sensor network at the university facilities. The sensor network would collect data from occupants in a controlled space while labelling their occupancy states at all times. This data, would have been used later to model human/building interactions by means of machine learning algorithms. Unfortunately, the resources available were limited and due to time constraints, the sensor network was not deployed and the data was never collected. This fact motivated the decision of researching publicly available quality datasets to train and validate our models. Even though there is an increasing number of datasets available nowadays, specific datasets that could fit in the context of this project were scarce and hard to find. There were also occasions where the datasets were only used for privately funded projects, therefore the data was never made available to other researchers.

After a thorough research in repositories and testbeds, we identified the most suitable datasets (mentioned in the different chapters of this thesis) following some specific criteria: they contained a significant number of samples and they were the datasets used in high quality previous researches related to the modelling of human/building interactions. In spite of finding several high quality datasets, none of them matched completely the characteristics of the datasets our own designed network would have provided. The

main differences were found in the fact that the current public datasets were more activity oriented compared to occupancy, and that none of them linked occupant patterns to energy efficiency and CO_2 emission levels.

Having the dataset limitation, we decided to accordingly modify the scope of our research, leaving aside the particularities of the energy efficiency aspect and focusing on occupant pattern modelling for all types of applications, namely healthcare, security or smart homes. Due to the fact that this contextual side of the project was not relevant any more, we could focus our efforts in the development of a more robust and sophisticated algorithm. By doing this, we could develop a more thorough solution and we were able to devote all our time and efforts to the building our novel DHSMM algorithm.

Additionally, since we never had the option to use our own physical system, it became impractical to embed our algorithm in a real BEMS to evaluate other aspects in the design of these algorithms. However, we had the opportunity of studying in depth a significant number of publicly available datasets and their associated publications. This eventually proved to be an invaluable help to better understand how the techniques of modelling human behaviour work from many new perspectives.

Finally, in spite of all the mentioned limitations, the resulting outputs from this project completely satisfactory and enabled us to explore new routes and lines of investigation that proved equally, if not more, rewarding and fulfilling as the ones initially set.

7.3 Future Work

Future work will include the implementation of our software in real world building management systems (BEMS) applications. The sensor information can be captured in real time and subsequently used to improve the energy efficiency of the buildings by regulating systems 'on the fly'. By making these different technologies work together, we can evaluate the challenges and practical aspects of implementing these technologies.

Moreover, having identified a significant scarcity of occupancy datasets, researchers dedicated to this topic should be encouraged to make available their datasets and findings, so more effort can be devoted to the development of new solutions or the improvement of the existing ones. The same could be said for works that, based on these new datasets,

present frameworks and baselines to perform model comparison. Due to the evolving nature of the development of new algorithms based on multiple different approaches (e.g. machine learning, data fusion and IoT, knowledge-driven models...), baselines and frameworks will be particularly helpful to identify the best solutions for each case or application.

Finally, our DHSMM model could be applied to other field and topics related to modelling human behaviour and patterns discovery such as human gesture detection or low level activity recognition, as it has proven to be a robust and reliable solution in the scenarios where it has been evaluated so far. Up to our knowledge, this is the first time that a model based on a HSMM approach has been able to be used for online and incremental learning in a human pattern modelling scenario; furthermore being evaluated in challenging setups including multi occupant datasets.

Bibliography

[1] Hande Alemdar, Halil Ertan, Ozlern Durmaz Incelt, and Cem Ersoy. ARAS Human Activity Datasets In Multiple Homes with Multiple Residents. *2013 7th International Conference on Pervasive Computing Technologies for Healthcare and Workshops*, pages 232–235, 2013.

[2] Hande Alemdar, Tim van Kasteren, and Cem Ersoy. Using active learning to allow activity recognition on a large scale. *Ambient Intelligence*, pages 105–114, 2011.

[3] Payam Nejat, Fatemeh Jomehzadeh, Mohammad Mahdi Taheri, Mohammad Gohari, and Muhd Zaimi Muhd. A global review of energy consumption, CO2 emissions and policy in the residential sector (with an overview of the top ten CO2 emitting countries). *Renewable and Sustainable Energy Reviews*, 43:843–862, 2015.

[4] Elie Azar and Carol C. Menassa. A comprehensive analysis of the impact of occupancy parameters in energy simulation of office buildings. *Energy and Buildings*, 55:841–853, 2012.

[5] Tae Seung Ha, Ji Hong Jung, and Sung Yong Oh. Method to analyze user behavior in home environment. *Personal and Ubiquitous Computing*, 10(2-3):110–121, 2005.

[6] Luis Pérez-Lombard, José Ortiz, and Christine Pout. A review on buildings energy consumption information. *Energy and buildings*, 40(3):394–398, 2008.

[7] Helton F. Scherer, Manuel Pasamontes, Jose Luis Guzmán, and Julio E. Normey-Rico. Efficient building energy management using distributed model predictive control. *Journal of Process Control*, 24(6):740 – 749, 2013.

[8] Jonathan Brooks, Saket Kumar, Siddharth Goyal, Rahul Subramany, and Prabir Barooah. Energy-efficient control of under-actuated HVAC zones in commercial buildings. *Energy and Buildings*, 93:160–168, 2015.

[9] Richard J de Dear, Takashi Akimoto, Edward A. Arens, Gail Brager, and Christhina Candido. Progress in thermal comfort research over the last twenty years. *Indoor air*, 23(6):442–461, 2013.

[10] Jessen Page, Darren Robinson, Nicolas Morel, and Jean-Louis Scartezzini. A generalised stochastic model for the simulation of occupant presence. *Energy and Buildings*, 40(2):83–98, 2008.

[11] Misha Pavel, Tamara L. Hayes, Andre Adami, Holly Jimison, and Jeffrey Kaye. Unobtrusive assessment of mobility. *Annual International Conference of the IEEE Engineering in Medicine and Biology - Proceedings*, 97239:6277–6280, 2006.

[12] Jose Luis G. Ortega, Liangxiu Han, Nick Whittacker, and Nicholas Bowring. A machine-learning based approach to model user occupancy and activity patterns for energy saving in buildings. *Proceedings of the 2015 Science and Information Conference, SAI 2015*, pages 474–482, 2015.

[13] Thiago Teixeira, Gershon Dublon, and Andreas Savvides. A Survey of Human-Sensing: Methods for Detecting Presence, Count, Location, Track, and Identity. *ACM Computing Surveys*, 5:1–35, 2010.

[14] Bing Dong and Burton Andrews. Sensor-based Occupancy Behavioral Pattern Recognition For Energy And Comfort Management In Intelligent Buildings. *Eleventh International IBPSA Conference*, pages 1444–1451, 2009.

[15] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[16] Shun Zheng Yu. Hidden semi-Markov models. *Artificial Intelligence*, 174(2):215–243, 2010.

[17] Hua Si, Yoshihiro Kawahara, Hiroyuki Morikawa, and Tomonori Aoyama. A stochastic approach for creating context-aware services based on context histories in smart home. *Histories in Smart Home, in: Proceeding of the 3rd International Conference on Pervasive Computing, Exploiting Context Histories in Smart Environments*, pages 97– 100, 2005.

[18] Junichi Yamagishi, Kazunori Ogata, and Yoshiaki Nakano. HSMM-Based Model Adaptation Algorithms for Average-Voice-Based Speech Synthesis. *2006 IEEE International Conference on Acoustics Speed and Signal Processing Proceedings*, 1 (3):77–80, 2006.

[19] Somboon Hongeng, Ram Nevatia, and Francois Bremond. Video-based event recognition: Activity representation and probabilistic recognition methods. *Computer Vision and Image Understanding*, 96(2 SPEC. ISS.):129–162, 2004.

[20] Bing Dong, Burton Andrews, Khee Poh Lam, Michael Höynck, Rui Zhang, Yun-Shang Chiou, and Diego Benitez. An information technology enabled sustainability test-bed (ITEST) for occupancy detection through an environmental sensing network. *Energy and Buildings*, 42(7):1038–1046, 2010.

[21] Tim van Kasteren, Gwenn Englebienne, and Ben Kröse. Human Activity Recognition from Wireless Sensor Network Data : Benchmark and Software. *Activity Recognition in Pervasive Intelligent Environments*, 4:165–186, 2011.

[22] Einat Marhasev, Meirav Hadad, and Gal A. Kaminka. Non-stationary hidden semi markov models in activity recognition. *Signal Processing*, 2006.

[23] Asma Benmansour, Abdelhamid Bouchachia, and Mohammed Feham. Multioccupant Activity Recognition in Pervasive Smart Home Environments. *ACM Computing Surveys*, 48(3):1–36, 2015.

[24] Haowen Chan and Adrian Perrig. Security and privacy in sensor networks. *Computer*, 36(10):103–105, 2003. ISSN 00189162. doi: 10.1109/MC.2003.1236475.

[25] Joshua M. Wiener, Raymond J. Hanley, Robert Clark, and Joan F. Van Nostrand. Measuring the activities of daily living: comparisons across national surveys. *Journal of gerontology*, 45(6):229–237, 1990.

[26] Emmanuel Munguia Tapia, Stephen S Intille, and Kent Larson. Activity Recognition in the Home Using Simple and Ubiquitous Sensors. *Pervasive Computing*, 3001:158–175, 2004.

[27] Anastasios I. Dounis and Christos Caraiscos. Advanced control systems engineering for energy and comfort management in a building environment—A review. *Renewable and Sustainable Energy Reviews*, 13(6-7):1246–1261, 2009.

[28] Irina Rish. An empirical study of the naive Bayes classifier. *IJCAI 2001 workshop on empirical methods in artificial intelligence*, 22230:41–46, 2001.

[29] Sunita Sarawagi and William W Cohen. Semi-Markov conditional random fields for information extraction. *Advances in Neural Information Processing Systems 17 (NIPS 2004)*, pages 1185–1192, 2004.

[30] Christopher J. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[31] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[32] Vishal Garg and Narendra K. Bansal. Smart occupancy sensors to reduce energy consumption. *Energy and Buildings*, 32(1):81–87, 2000.

[33] Thananchai Leephakpreeda. Adaptive occupancy-based lighting control via Grey prediction. *Building and Environment*, 40(7):881–886, 2005.

[34] Robert H. Dodier, Gregor P. Henze, Dale K. Tiller, and Xin Guo. Building occupancy detection through sensor belief networks. *Energy and Buildings*, 38(9): 1033–1043, 2006.

[35] Byung S. Lee, Trevor P. Martin, Nathan P. Clarke, Bivrahg Majeed, and Detlef Nauck. Dynamic daily-living patterns and association analyses in tele-care systems. *Proceedings - Fourth IEEE International Conference on Data Mining, ICDM 2004*, pages 447–450, 2004. doi: 10.1109/ICDM.2004.10023.

[36] Frédéric Haldi and Darren Robinson. Interactions with window openings by office occupants. *Building and Environment*, 44(12):2378–2395, 2009.

[37] Tina Yu. Modeling occupancy behavior for energy efficiency and occupants comfort management in intelligent buildings. *Proceedings - 9th International Conference on Machine Learning and Applications, ICMLA 2010*, pages 726–731, 2010.

[38] João Virote and Rui Neves-Silva. Stochastic models for building energy prediction based on occupant behavior assessment. *Energy and Buildings*, 53:183–193, 2012.

[39] Deniz Kara and Peter Baxendale. An agent based building energy consumption model. *Innovative Smart Grid Technologies - Asia (ISGT Asia), 2012 IEEE*, pages 1–5, 2012.

[40] Laura Klein, Jun Young Kwak, Geoffrey Kavulya, Farrokh Jazizadeh, Burcin Becerik-Gerber, Pradeep Varakantham, and Milind Tambe. Coordinating occupant behavior for building energy and comfort management using multi-agent systems. *Automation in Construction*, 22:525–536, 2012.

[41] Rui Yang and Lingfeng Wang. Multi-zone building energy management using intelligent control and optimization. *Sustainable Cities and Society*, 6(1):16–21, 2013.

[42] Yuvraj Agarwal, Bharathan Balaji, Rajesh Gupta, Jacob Lyles, Michael Wei, and Thomas Weng. Occupancy-driven energy management for smart building automation. *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building - BuildSys '10*, pages 1–6, 2010.

[43] Sunil Mamidi, Yu H. Chang, and Rajiv Maheswaran. Improving building energy efficiency with a network of sensing, learning and prediction agents. *AAMAS '12 Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, 1:42–52, 2012.

[44] John Krumm Brian Meyers Mike Hazas Steve Hodges Nicolas Villar James Scott A.J. Brush. PreHeat: Controlling Home Heating Using Occupancy Prediction. *UbiComp '11 Proceedings of the 13th international conference on Ubiquitous computing*, pages 281–290, 2011.

[45] Tobore Ekwevugbe, Neil Brown, and Denis Fan. A design model for building occupancy detection using sensor fusion. *2012 6th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*, pages 1–6, 2012.

[46] Ebenezer Hailemariam, Rhys Goldstein, Ramtin Attar, and Azam Khan. Real-time occupancy detection using decision trees with multiple sensor types. *Procs. of the 2011 Symposium on Simulation for Architecture and Urban Design*, pages 141–148, 2011.

[47] Nan Li, Gulben Calis, and Burcin Becerik-Gerber. Measuring and monitoring occupancy with an RFID based system for demand-driven HVAC operations. *Automation in Construction*, 24:89–99, 2012.

[48] Zheng Yang, Nan Li, B Becerik-Gerber, and Michael Orosz. A multi-sensor based occupancy estimation model for supporting demand driven HVAC operations. *SimAUD '12 Proceedings of the 2012 Symposium on Simulation for Architecture and Urban Design*, pages 98–105, 2012.

[49] Hwataik Han, Kyung-Jin Jang, Changho Han, and Junyong Lee. Occupancy Estimation Based on Co2 Concentration Using Dynamic Neural Network Model. *Aivc.Org*, 2013.

[50] Eldar Naghiyev, Mark Gillott, and Robin Wilson. Three unobtrusive domestic occupancy measurement technologies under qualitative review. *Energy and Buildings*, 69:507–514, 2014.

[51] Ali Karamath and George Amalarethinam. Automatic Construction of Finite Automata for Online Recognition of User Activities in Smart Environments. *IOSR Journal of Engineering (IOSRJEN)*, 3(12):40–45, 2013.

[52] Tobore Ekwevugbe, Neil Brown, Vijay Pakka, and Denis Fan. Real-time building occupancy sensing using neural-network based sensor network. *2013 7th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*, pages 114–119, 2013.

[53] James Howard and William Hoff. Forecasting building occupancy using sensor network data. *Proceedings of the 2nd International Workshop on Big Data, Streams and Heterogeneous Source Mining Algorithms, Systems, Programming Models and Applications - BigMine '13*, pages 87–94, 2013.

[54] Kaibin Bao, Florian Allerding, and Hartmut Schmeck. User behavior prediction for energy management in smart homes. *2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, 2:1335–1339, 2011.

[55] Jared Langevin, Patrick L. Gurian, and Jin Wen. Tracking the human-building interaction: A longitudinal field study of occupant behavior in air-conditioned offices. *Journal of Environmental Psychology*, 42:94–115, 2015.

[56] Luis M Candanedo and Veronique Feldheim. Accurate occupancy detection of an office room from light, temperature, humidity and CO2measurements using statistical learning models. *Energy and Buildings*, 112:28–39, 2016.

[57] Liming Chen and Jesse Hoey. Sensor-based activity recognition. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 42(6): 790–808, 2012.

[58] Tomomi Yasuda, Seiichiro Yonemura, and Akira Tani. Comparison of the characteristics of small commercial NDIR CO2 sensor models and development of a portable CO2 measurement device. *Sensors (Basel, Switzerland)*, 12(3):3641–3655, 2012.

[59] Victor M. Zavala. Inference of building occupancy signals using moving horizon estimation and Fourier regularization. *Journal of Process Control*, 2013.

[60] Maryam Ziaeefar and Robert Bergevin. Semantic human activity recognition: A literature review. *Pattern Recognition*, 48(8):2329–2345, 2015.

[61] Ryan Aipperspach, Elliot Cohen, and John Canny. Modeling human behavior from simple sensors in the home. *Pervasive Computing*, 2006.

[62] Julie a. Kientz, Shwetak N. Patel, Brian Jones, Ed Price, Elizabeth D. Mynatt, and Gregory D. Abowd. The Georgia Tech Aware home. *Extended Abstracts on Human Factors in Computing Systems*, pages 3675–3680, 2008.

[63] Geetika Singla, Diane J Cook, and Maureen Schmitter-Edgecombe. Recognizing independent and joint activities among multiple residents in smart environments. *Journal of ambient intelligence and humanized computing*, 1(1):57–63, 2010.

[64] Diane J. Cook, Aaron S. Crandall, Brian L. Thomas, and Narayanan C. Krishnan. CASAS: A smart home in a box. *Computer*, 46(7):62–69, 2013.

[65] Tim van Kasteren, Gwenn Englebienne, and Ben Kröse. Transferring knowledge of activity recognition across sensor networks. *Pervasive computing*, pages 283–300, 2010.

[66] Tim van Kasteren. Activity recognition using semi-markov models on real world smart home datasets. *Journal of Ambient Intelligence and Smart Environments*, 2(3):311–325, 2010.

[67] Enamul Hoque and John Stankovic. AALO: Activity recognition in smart homes using Active Learning in the presence of Overlapped activities. *Proceedings of the 6th International Conference on Pervasive Computing Technologies for Healthcare*, pages 139–146, 2012.

[68] Juan Ye, Graeme Stevenson, and Simon Dobson. KCAR: A knowledge-driven approach for concurrent activity recognition. *Pervasive and Mobile Computing*, 2014.

[69] Hongqing Fang and Lei He. BP Neural Network for Human Activity Recognition in Smart Home. *2012 International Conference on Computer Science and Service System*, pages 1034–1037, 2012.

[70] Iram Fatima, Muhammad Fahim, YK Lee, and Sungyoung Lee. A Genetic Algorithm-based Classifier Ensemble Optimization for Activity Recognition in Smart Homes. *TIIS*, X(X):134–154, 2013.

[71] Hui Li, Qingfan Zhang, and Peiyong Duan. A novel one-pass neural network approach for activities recognition in intelligent environments. *2008 7th World Congress on Intelligent Control and Automation*, pages 50–54, 2008.

[72] Jose Luis G. Ortega, Liangxiu Han, and Nicholas Bowring. Modelling and Detection of User Activity Patterns for Energy Saving in Buildings. *Emerging Trends and Advanced Technologies for Computational Intelligence: Extended and Selected Results from the Science and Information Conference 2015*, pages 165–185, 2016.

[73] Maria Ros, Manuel P. Cuéllar, Miguel Delgado, and Amparo Vila. Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows. *Information Sciences*, 220:86–101, 2013.

[74] Narayanan C. Krishnan and Diane J. Cook. Activity recognition on streaming sensor data. *Pervasive and Mobile Computing*, 10:138–154, 2014.

[75] George Okeyo, Liming Chen, and Hui Wang. Combining ontological and temporal formalisms for composite activity modelling and recognition in smart homes. *Future Generation Computer Systems*, 39:29–43, 2014.

[76] Ming Xu, Longhua Ma, Feng Xia, Tengkai Yuan, Jixin Qian, and Meng Shao. Design and Implementation of a Wireless Sensor Network for Smart Homes. *2010 7th International Conference on Ubiquitous Intelligence & Computing and 7th International Conference on Autonomic & Trusted Computing*, 2(2):239–243, 2010.

[77] Jose Luis G. Ortega and Liangxiu Han. Modelling Occupant Activity Patterns For Energy Saving In Buildings Using Machine-Learning Approaches. *ZEMCH 2015, International Conference Proceedings*, pages 241–253, 2015.

[78] Husheng Guo and Wenjian Wang. An active learning-based SVM multi-class classification model. *Pattern Recognition*, 48(5):1577–1597, 2015.

[79] Daniel Leite, Pyramo Costa, and Fernando Gomide. Evolving granular neural network for semi-supervised data stream classification. *Proceedings of the International Joint Conference on Neural Networks*, 2010.

[80] Junjing Yang, Mattheos Santamouris, and Siew Eang Lee. Review of occupancy sensing systems and occupancy modeling methodologies for the application in institutional buildings. *Energy and Buildings*, 121:344–349, 2016.

[81] Diane J. Cook, Narayanan C. Krishnan, and Parisa Rashidi. Activity Discovery and Activity Recognition: A New Partnership. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, pages 1–9, 2012. ISSN 1083-4419. doi: 10.1109/TSMCB.2012.2216873. URL `http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=6313931`.

[82] Ben Kröse, Tim van Kasteren, and Christian. Gibson. CARE: Context Awareness in Residences for Elderly. *International Conference of the International Society for Gerontechnology*, pages 101–105, 2008.

[83] Stephen S. Intille, Kent Larson, Emmanuel Munguia Tapia, Jennifer S. Beaudin, Pallavi Kaushik, Jason Nawyn, and Randy Rockinson. Using a live-in laboratory for ubiquitous computing research. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3968 LNCS:349–365, 2006.

[84] Mathieu Gallissot, Jean Caelen, Nicolas Bonnefond, Brigitte Meillon, and Sylvie Pons. Using the Multicom Domus Dataset. Research Report RR-LIG-020, LIG, Grenoble, France, 2011.

[85] Carlos Duarte, Kevin Van Den Wymelenberg, and Craig Rieger. Revealing occupancy patterns in an office building through the use of occupancy sensor data. *Energy and Buildings*, 67:587–595, 2013.

[86] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9(2008):1871–1874, 2008.

[87] Mahesh Pal. Multiclass approaches for support vector machine based land cover classification. *Proc. 8th Annu. Int. Conf., Map IndiaProc. 8th Annu. Int. Conf., Map India*, 2005.

[88] Piotr Indyk. Approximate Nearest Neighbors : Towards Removing the Curse of Dimensionality RAJEEV MoTwd Department of Computer Science Stanford University. *Department of Computer Science Stanford UniversityStanford, CA 94305*, pages 604–613, 1998.

[89] Shun Zheng Yu and Hisashi Kobayashi. A hidden semi-Markov model with missing data and multiple observation sequences for mobility tracking. *Signal Processing*, 83(2):235–250, 2003.

[90] Takashi Yamazaki, Naotake Niwase, Junichi Yamagishi, and Takao Kobayashi. Human walking motion synthesis based on multiple regression hidden semi-Markov model. *Proceedings - 2005 International Conference on Cyberworlds, CW 2005*, 2005:445–452, 2005.

[91] Shun-zheng Yu, Hisashi Kobayashi, and Life Fellow. Practical Implementation of an Efficient Forward – Backward Algorithm for an. *ieee transactions on signal processing*, 54(5):1947–1951, 2006.

[92] Wael Khreich, Eric Granger, Ali Miri, and Robert Sabourin. A survey of techniques for incremental learning of HMM parameters. *Information Sciences*, 197:105–130, 2012.

[93] Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. Information-theoretic metric learning. *Icml*, pages 209–216, 2007.

[94] Amir Saffari, Christian Leistner, Martin Godec, and Horst Bischof. On-line Random Forests. *2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1393–1400, 2009.

[95] Nir Friedman and Moises Goldszmidt. Sequential update of Bayesian network structure. *UAI'97 Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence*, 1:165–174, 1997. URL `http://dl.acm.org/citation.cfm?id=2074246`.

[96] David J Hill, Barbara S Minsker, and Eyal Amir. Real-Time Bayesian Anomaly Detection for Environmental Sensor Data. *Proceedings of the Congress-International Association for Hydraulic Research*, 2:503–508, 2007.

[97] Daniel B Neill, Gregory F Cooper, Kaustav Das, Xia Jiang, and Jeff Schneider. Bayesian Network Scan Statistics for Multivariate Pattern Detection. *Scan Statistics: Methods and Applications*, pages 221–249, 2009. doi: 10.1007/978-0-8176-4749-0_11. URL `http://www.springerlink.com/index/10.1007/978-0-8176-4749-0`.

[98] Tiberiu Chis. Sliding hidden markov model for evaluating discrete data. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8168 LNCS:251–262, 2013.

[99] Robert A. Jacobs. Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1(4):295 – 307, 1988.

[100] Adrian G. Fischer and Markus Ullsperger. When is the time for a change? decomposing dynamic learning rates. *Neuron*, 84(4):662 – 664, 2014.

[101] Kevin P. Murphy. Conjugate Bayesian Analysis of the Gaussian Distribution. *Def*, 1(7):1–29, 2007.

[102] Zheng Yang and Burcin Becerik-Gerber. Modeling personalized occupancy profiles for representing long term patterns by using ambient context. *Building and Environment*, 78(August):23–35, 2014.

[103] Yoon Soo Lee and Ali M. Malkawi. Simulating multiple occupant behaviors in buildings: An agent-based modeling approach. *Energy and Buildings*, 69:407–416, 2014.

[104] Aaron S Crandall and Diane J Cook. Resident and Caregiver: Handling Multiple People in a Smart Care Facility. *AI in Eldercare: New Solutions to Old Problems*, pages 39–47, 2008.

[105] Rong Chen and Yu Tong. A two-stage method for solving multi-resident activity recognition in smart environments. *Entropy*, 16(4):2184–2203, 2014.

[106] Kuo Chung Hsu, Yi Ting Chiang, Gu Yang Lin, Ching Hu Lu, Jane Yung Jen Hsu, and Li Chen Fu. Strategies for inference mechanism of conditional random fields for multiple-resident activity recognition in a smart home. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6096 LNAI(PART 1):417–426, 2010.

[107] Liang Wang, Tao Gu, Xianping Tao, Hanhua Chen, and Jian Lu. Recognizing multi-user activities using wearable sensors in a smart home. *Pervasive and Mobile Computing*, 7(3):287–298, 2011.

[108] Tiberiu Chis and Peter G. Harrison. Adapting Hidden Markov Models for Online Learning. *Electronic Notes in Theoretical Computer Science*, 318:109–127, 2015.

[109] Jose Luis G. Ortega, Liangxiu Han, and Nicholas Bowring. A novel dynamic hidden semi-markov model (d-hsmm) for occupancy pattern detection from sensor data stream. *2016 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pages 1–5, 2016.

[110] Predrag Klasnja, Sunny Consolvo, Tanzeem Choudhury, Richard Beckwith, and Jeffrey Hightower. Exploring Privacy Concerns about Personal Sensing. *Pervasive Computing: 7th International Conference, Pervasive 2009, Nara, Japan, May 11-14, 2009*, pages 173–186, 2009.

[111] Tuan Anh Nguyen and Marco Aiello. Energy intelligent buildings based on user activity: A survey. *Energy and Buildings*, 56:244–257, 2013.

[112] Zheng Yang and Burcin Becerik-Gerber. Cross-Space Building Occupancy Modeling by Contextual Information Based Learning. *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments - BuildSys '15*, pages 177–186, 2015.