Activity-driven Detection of Cognitive Impairment using Deep Learning

by

Damla Arifoglu

A thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Science and Technology)
in Bournemouth University
2019

Supervisory Team:

Professor Hamid Bouchachia, First Advisor
Dr. Hammadi Nait-Charif, Second Advisor

To my dear parents, Ziya and Ayse,

for their endless love and support

ACKNOWLEDGEMENTS

First of all, I would like to thank to my supervisor Prof. Abdelhamid Bouchachia from whom I have learned a lot due to his supervision, patient guidance, and support during this research. Without his invaluable assistance and encouragement, this thesis would not be possible. Moreover, I would like to thank to my second advisor Dr. Hammadi Nait-Charif for his input and comments during this research. I am indebted to the members of my thesis committee for accepting to review my thesis and for their valuable comments.

I thank to the members of P256 office, Yan, Soumya, Roman and Waqas for their good companionship. I am grateful to Dr. Emili Balaguer-Ballaster for his cheerful, kind and considerate manners towards me. I would like to thank my friends Jaspreet, Kim, Marika and Leticia for making my life more enjoyable in Bournemouth. I would like to thank Ines for sending her support and prays for me. I am grateful to Esma F. Bilgin for our insightful conversations about social and PhD life. Moreover, I would like to express my gratitude to my friend Mehmet for supporting me in every aspect of my life since 2003.

Last but not the least, I would like to thank my family for supporting me spiritually throughout my life in general. I would like to thank to my little nephew Tugberk specially, for his unconditional love and making me smile during our video chats. Lastly, I would like to express my gratitude to Ozan, who has been a constant source of support and encouragement during the challenges of graduate school and life. Without his care, love and support, it wouldn't be possible to complete this thesis.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

MCI	Mild Cognitive Impairment
ADL	Activities of Daily Living
IADLs	Instrumental Activities of Daily Living
SVMs	Support Vector Machines
NB	Naïve Bayes
CRFs	Conditional Random Fields
HMMS	Hidden Markov Models
RBMs	Restricted Boltzmann Machines
HCRF	Hidden State Conditional Random Field
HSMM	Hidden Semi-Markov Model
DBNS	Deep Belief Networks
DNN	Deep Neural Network
RNN	Recurrent Neural Network
V-RNN	Vanilla Recurrent Neural Network
GRU	Gated Recurrent Unit
LSTM	Long Short Term Memory
ReLU	Rectified Linear Unit
CNN	Convolutional Neural Network
RAE	Recursive Auto-Encoder
GCN	Graph Convolutional Network

TPR	True Positive Rate
FPR	False Positive Rate
TNR	True Negative Rate
BOS	Bag-of-sensors
RSM	Raw Sensor Measurement
ROC	Receiver Operating Characteristic
AUC	Area Under Curve
PCA	Principal Component Analysis

ABSTRACT

While life expectancy is on the rise all over the world, more people face health related problems such as cognitive decline. Cognitive impairment is a collective name for progressive brain syndromes which affect memory, cognition, behaviour and emotion. People suffering from cognitive impairment may lose their abilities to perform daily life activities and they get dependent on their caregivers. Although some medications can slow the progress of the disease, currently there is no way to stop its development. Sufferers may require special needs which increase the cost of care. Thus, detecting the indicators of cognitive decline before it gets worse would be very crucial. Current assessment methods mostly rely on queries from questionnaires or in-person examinations, which depend on recall of events that may poorly represent a person's typical state.

The aim in this thesis is to adapt deep learning techniques for analysing daily activities of elderly people and detecting abnormalities in the activity patterns. Recent studies suggest that indicators of cognitive decline can be observed in daily life activity patterns. The spatio-temporal and hierarchical relationship of activities and their intrinsic structures are important in the context of cognitive decline analysis. Existing studies treat each activity as an atomic unit and fail to capture the relationship among sub-activities. Also, existing studies rely on fixed length features to model activities, ignoring the granular level information coming from raw sensor activations. Moreover, there exists no daily activity dataset representing the behaviour of dementia sufferers because producing such datasets requires time and adequate experimental environment. Given these challenges, the present thesis addresses the following research questions: How can we cope with the scarcity of dataset reflecting on cognitive status of elderly people? How can activities be modelled taking into account their spatio-temporal neighbourhood and hierarchical information? How can we represent raw data to encode the granular level details? These research questions are addressed in the following way. Firstly, two methods are proposed to cope with the scarcity of data: (i) synthetic data generation and (ii) transfer learning adoption. Secondly, the activity recognition problem is emulated (i) as a sequence labelling problem to model spatio-temporal patterns. (ii) as a hierarchical learning problem to model sub-activities. (iii) as a graph labelling problem to encode granular level details. Thirdly, raw sensor measurements stemming from sequential data are used to model sensor activation relationships. The proposed methods are also compared against the state-of-art methods. The preliminary results obtained indicate that proposed data simulation and transfer learning approaches are useful to cope with the scarcity of data reflecting cognitive status of elderly people. Moreover, experiments show that the proposed deep learning methods are promising to detect abnormalities in the context of cognitive decline. Proposed methods are not only promising to detect abnormal behaviour at a fine-grained level, but some of them can also model activities hierarchically by taking sub-activities into account and then can detect abnormal behaviour occurring at granular levels.

CHAPTER I

Introduction

1.1 Introduction

In this chapter, we present the challenges and the motivation of this research. For this purpose, we provide the definition and the types of cognitive decline and its prevalence across the world. Also, we describe the indicators of dementia that can be observed in daily life activities of elderly people. We describe research questions and objectives as well as the outcomes of this research as publications. The organisation of this chapter is as follows. In Section 1.2, the definition and prevalence of dementia across the world is presented. Moreover, some statistics and facts about the prevalence of dementia are provided. Then the indicators of dementia that can be seen in daily life activity patterns are summarised in Section 1.3. Research objectives and questions are listed in Section 1.4. Research methodology is described in Section 1.5. Research contributions are presented in Section 1.6. Structure of the thesis is described in Section 1.7, while research outcomes are listed in Section 1.8.

1.2 Dementia and its Prevalence

In recent years, cognitive decline and mental diseases in ageing persons have become a major public health concern all over the world. Alzheimer's Disease International estimated that the number of people suffering from dementia worldwide in 2018 to be 50 million and this number will increase to 82 million by 2030 and to 150 million by 2050 [62]. These numbers underline a situation that presents a certain level of criticality.

Elderly people may suffer from the consequences of cognitive decline, which causes problems with mobility, physical and mental abilities such as memory and cognition [3]. It may also affect the ability of speaking, writing, distinguishing objects, and performing complex functional tasks (paying bills, preparing a meal, shopping, etc.) [79]. Elderly people need special care and help from their caregivers leading to a social, psychological, physical and economic challenge to family members, caregivers and the society as a whole. Dementia comes with one or more of the following symptoms:

- difficulties in performing motor activities
- difficulties for reasoning abstractly, making valid judgements and planning for complicated tasks
- decrease in the ability of speaking or comprehending spoken or written language
- difficulties to distinguish or identify objects

Dementia is more likely to occur in old age and the number of people with dementia is increasing rapidly as depicted in Figure 1.1 [56]. Although some medications can slow the progress of the disease, currently there is no way to stop its development altogether or reverse its impacts on the brain cells. The pace of the dementia progress is specific to the individuals and even the symptoms may not be the same for all individuals. However, cognitive impairment is categorised into 7 stages [4]:

- absence of cognitive decline

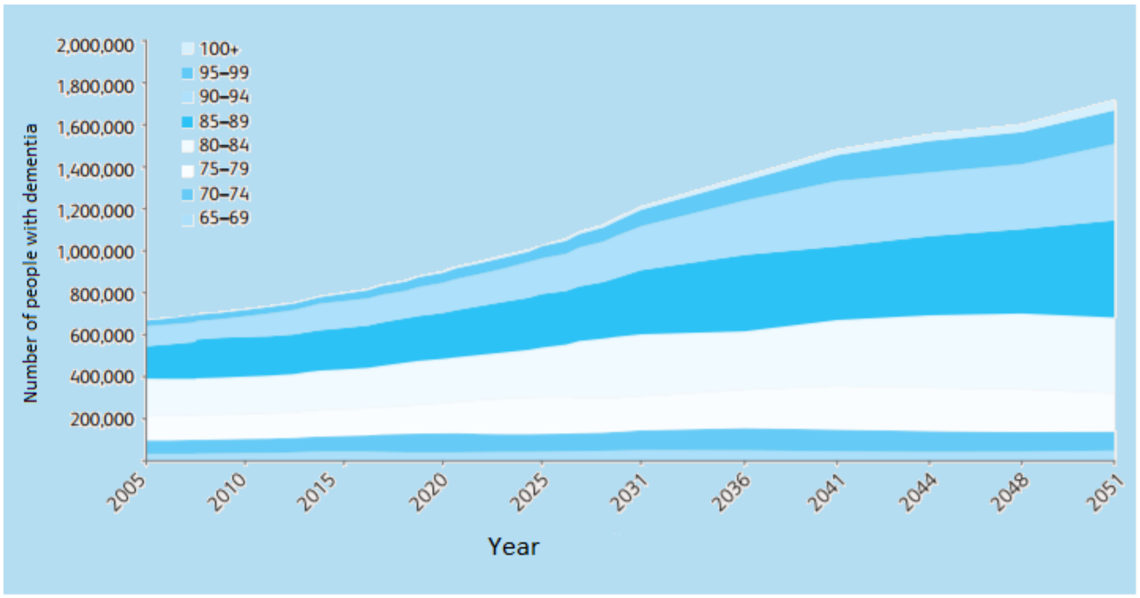


Figure 1.1: Dementia progress in the UK.

- very mild cognitive decline
- mild cognitive decline (Mild Cognitive Impairment (MCI))
- moderate cognitive decline (Mild Dementia)
- moderately severe cognitive decline
- severe cognitive decline
- very severe cognitive decline

Dementia may progress from early to late stages in typically 5 to 10 years. Alzheimer's disease is the most common type of severe cognitive decline. In many cases, elderly people with Mild Cognitive Impairment (MCI) will never progress to Alzheimer's disease and can live with an acceptable degree of independence [5]. However, sufferers from each level may require special needs and the cost of social care may differ. As depicted in Table 1.1 (£, in millions, 2012/2013 prices), the cost to the society increases as the situation gets worse [56]. The total cost of dementia in the UK is £26.3 billion, with an average cost of £32.250 per person, while £4.3 billion is spent on health-care costs. £10.3 billion is spent on social care (publicly and privately funded) and £11.6 billion is contributed by the work of unpaid carers.

Table 1.1: Total annual cost (£, in millions) for dementia care, by severity and setting.

	Health	**Social**	**Unpaid**	**Other**	**Total**
Mild	1,332	2,436	8,079	62	11,910
Moderate	2,055	5,626	2,587	36	10,303
Severe	926	2,209	954	14	4,102
All levels	4,314	10,271	11,620	111	26,316
Cost (% of total)	16.4%	39.0%	44.2%	0.4%	100.0%

Most elderly people prefer to stay in their own homes and to be in contact with their families. Studies show that it is better for the health of these people to stay in a self-determined private home environment while ageing. It has been observed that age-in-place can reduce the speed of dementia progress, thus improving people's quality of life [18]. However, informal care at home can be excessively expensive, and in some cases, is not possible at all [9]. On the other hand, the use of assisted living technologies such as smart homes can substantially help people with dementia to live independently. A smart home is a house which is equipped with sensors to enable monitoring the occupants capturing their behaviour and understanding their activities. In this way, the system can inform about risky situations and take actions on behalf of the occupant. Moreover, this system may be helpful to detect the early indicators of dementia and then warn the caregivers and medical doctors for further diagnosis. Unfortunately, currently there are no dementia friendly smart homes addressing these people's special needs.

1.3 Indicators of Dementia

Cognitive diseases like dementia need to be detected at an early stage so that early treatment will be possible. However, research shows that 75% of dementia and early dementia cases go unnoticed [38] and many such cases are only diagnosed when the impairment reaches moderate or advanced stage. Recent studies [89] suggest that changes in complex daily life tasks can be indicators of early decline. The best markers of cognitive decline may not necessarily be detected based on a person's performance at any single point in time, but rather by monitoring the trend over time and the variability of change in a duration. Most common types of dementia can be identified by behavioural changes like sleep disturbances, difficulty of walking and inability to complete tasks [6]. Thus, such changes can provide key information about memory, mobility and cognition of a person. For instance, an inhabitant suffering from Alzheimer may forget to have his lunch, take multiple lunches instead, wake up in the middle of the night or go to the toilet frequently. Moreover there can be abnormality in eating habits (for example having snacks in the middle of the night) or in dehydration because of forgetting to drink daily amount of water. Sufferers may also have problems which involve risky situations such as forgetting to turn off the heater, the oven, or the stove [6].

Moreover, elderly people with cognitive decline may suffer from the consequences of confusion (for example not being able to run the dishwasher or confusing names on a phone book). Some other symptoms of dementia include sleep disorders, restlessness, wandering around in the night, misunderstanding of time, vocalisation and shouting, higher rate of falls, propensity to other accidents. Sleep disturbances are common for people with dementia, and often lead also to carers experiencing problems with their sleep. A person with dementia may get up repeatedly during the night and may become disorientated when she/he wakes up. She/he may get dressed and try to leave the house in the middle of the night. This may make the person tired during the day and they may sleep for long periods.

In particular, changes in daily life activity (preparing food, showering, walking, sleeping, watching television, going to the toilet, having dinner/lunch/breakfast, cleaning the house, etc.) patterns, their occurrence time and frequency are key indicators in determining the cognitive status of elderly people. Thus, a smart system is needed to track an elderly person's daily life by recording the basic activities at home. Tracking an elderly person's activities over time in a specially designed smart home, doing in-home health assessment and detecting the indicators of dementia at an early step would be beneficial for further diagnosis. Detecting early signs of motion and cognitive impairment (MCI) via activity recognition will be useful to evaluate motion and cognitive capabilities of the elderly, in order to take action towards improving their life quality and financial saving. Early detection of cognitive decline indicators would be helpful to warn caregivers, medical doctors and clinicians so that early treatment would be possible.

In this research, we aim to do in-home assessment of cognitive health status for the elderly people with dementia in a daily life scenario by exploiting machine learning techniques. We will be focusing on indicators of cognitive impairment which can be observed in daily life activities. The development of ambient home assessment environments has begun to provide the opportunity to assess change continuously, unobtrusively and in real-time [21, 20, 38, 89, 20, 69, 48, 46, 25]. Prevention or delay of cognitive decline onset is contingent upon the ability to detect early, meaningful, cognitive change during the life course. The translation of the current knowledge into smart homes still requires more dedication and work. Current assessment methods mostly rely on queries from questionnaires or in-person examinations, that depend on recall of events or brief snap-shots of the function that may poorly represent a person's typical state. Moreover, these studies include some pre-defined tasks given to the patients to do automatic assessment of cognitive decline by trained experts.

1.4 Research Questions and Objectives

The main motivation behind this work is that indicators of cognitive decline can be observed in daily activities and routines. Real-time monitoring of activities in a smart home would be beneficial for the early detection of such decline. The aim in this thesis is to develop adapted techniques for analysing daily activities of elderly people, detecting and tracking changes of activities over time, categorising and presenting any changes and abnormalities in the activity and cognitive patterns. In the context of early detection of dementia, the family members and caregivers will be warned about the status of patient and medical doctors can take action for further diagnosis and treatment before the condition of the person worsens.

This research addresses the following research questions:

1. Can any of the early signs of Mild Cognitive Impairment (MCI) be observed through abnormalities in daily life routines of an elderly person, such as sleeping, cooking, grooming, eating and working? How can we analyse and categorise these indicators?

2. Given the difficulty of obtaining a real-world dataset, how can we simulate abnormal behaviour of dementia sufferers in daily life? How can we modify publicly available daily life activity recognition datasets to obtain data related to cognitive decline?

3. How do we assess behaviour and cognitive status of an elderly person remotely using sensors placed carefully in a dementia smart home through observations of the person's activities? How can daily life activities be modelled taking their temporal and spatial neighbourhood into account? Is the temporal ordering and the spatial information of sensor activations important in terms of flagging abnormal behaviour related to dementia?

4. Which machine learning methods are the best for the activity recognition of dementia friendly houses, generative or discriminative?

5. How can daily life activities be modelled taking their sub-activities and their

inter-relationships into account? Is the order and the frequency of these sub-activities important in terms of detecting abnormal behaviour related to dementia?

6. In the absence of training data, can we learn the normal behaviour and daily life patterns of a (cognitively) healthy person and use them as a basis for tracking other patients? Can we adapt Recursive Auto-Encoders (RAE)-based transfer learning to cope with the problem of scarcity of data in the context of abnormal behaviour detection?

7. Can we model individual sensor activations and their relationship with each other in a graph? How can we exploit properties of graphs to model activities and detect outlier behaviour in a graph of sensor activations?

This research will be about merging the tasks of activity recognition and abnormal behaviour detection to detect early signs of cognitive decline. The process consists of 1) modelling activities by deep learning methods to model the daily behaviour routines of a person and 2) detecting any abnormality deviating from these regular behaviour whenever a new sequence is introduced to the classifier. The objectives of this research will be to address the aforementioned research questions.

1.5 Research Methodology

Unfortunately, there exists no data reflecting on cognitive status of elderly people. Thus, in this study, we cope with the scarcity of the data reflecting on abnormal behaviour of elderly people in two ways: (i) data generation and (ii) transfer learning. When there is lack of dataset, simulation of abnormal behaviour [28, 53] or exploiting transfer learning can be solution [63, 13, 43]. Moreover, we aim to recognise daily life activities by taking their temporal, spatial, hierarchical and granular level information into account. The spatio-temporal information of activities, their ordering and frequency are important in the context of cognitive decline. Thus, we emulate activity recognition and abnormal behaviour detection problems as a (i) sequence labelling problem, (ii) hierarchical learning problem and (iii) a graph labelling problem. The main idea is to build high-level activities hierarchically from their low-level activity patterns, which are called sub-activities. Then, abnormal behaviour related to dementia will be detected by considering the sub-activities, their relationship with each other such as their occurrence order, their frequency, etc. In images, there are pixels, then these pixels form edges and edges construct shapes. Similar to image recognition, there are granular-level patterns in daily life activities. For example; the activity *wash clothes* implies the following actions: *get clothes from basket, fill up washing machine, turn on washing machine*. Some daily life activities such as *sleeping* or *wash dishes* may not have explicit sub-activities involved in. But, we can exploit motion sensors replaced at home and their relative location with each other as sub-activities. For example; an occupant in a house may mainly move around the kitchen sink in the *wash dishes* activity and stay around the bedroom area during *sleeping* activity. In *wash dishes* activity, the movement between kitchen range and the sink is observed and this leads to triggering of sensors next to the sink and kitchen in some order. As depicted in Figure 1.2, the activity *wash dishes* consists of sub-activities *move to the kitchen area*, *move to the kitchen sink* and *use water*. The ordered sequence of motion sensors M_3, M_4, M_6 form these sub-activities hierarchically and then they result in *wash dishes* activity. In the present study, we first model sub-activities and their relationship with the help of deep learning techniques. Then we use these sub-activities, their frequency, their hierarchical and spatio-temporal relationship and relative order to detect abnormal behaviour related to dementia. Recognising these sub-activities and constructing upper level activities based on a hierarchical relationship of these granular level structures would be helpful to better understand and model abnormal behaviour related to dementia. Moreover, since there is no publicly available dataset, we propose a method to artificially produce data related to abnormal behaviour activities of dementia sufferers.

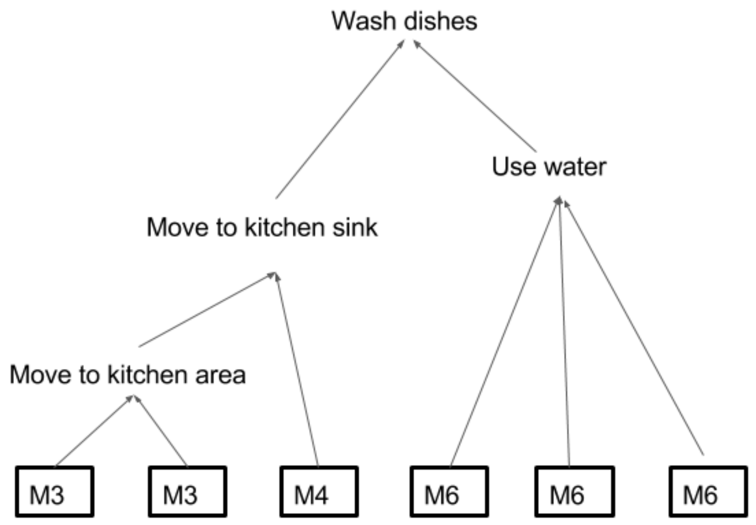


Figure 1.2: Activity *washing dishes* and its sub-activities.

In terms of investigation methodology, the thesis adopts an explorative approach which consists of formulating hypotheses, designing algorithms, evaluating these algorithms on data synthesised and validating the hypotheses. These methodologies are used in the following way.

1. In the first phase (exploratory) a comprehensive literature survey is conducted where the dementia symptoms were collected from the existing literature.

2. In the second phase (hypothesis), deep learning methods are proposed to recognise daily life activities and detect abnormal behaviour related to dementia.

3. In the third phase (experimental), the experiment set-up is designed, the deep learning methods are implemented.

4. In the fourth phase (evaluation), the proposed methods are evaluated for both recognising activities and detecting abnormal behaviour.

1.6 Contributions

The contributions of this research are listed as below.

1. In Chapter III, a method is proposed to generate synthetic data that simulates the abnormal behaviour of people with dementia since there is no daily activity dataset reflecting behaviour of dementia sufferers in the literature. These abnormalities are generated to present both activities and sub-activities within the activity sequences. These kinds of abnormal behaviour more specifically reflect on repetition, confusion and sleep disorder anomalies stemming from cognitive decline indicators.

2. In Chapter IV, daily activity recognition problem is emulated as a sequence labelling problem to fit deep learning techniques. Convolutional Neural Networks and Recurrent Neural Networks are exploited to model spatio-temporal patterns for daily life activity recognition and detect abnormal behaviour reflecting cognitive status of elderly people.

3. In Chapters V and VI, instead of extracting traditional features from each time slice, raw sensor measurements coming from sequential data are processed to represent temporal data. In this way, we don't lose the frequency of each sensor activation and their order, which is important in the context of detecting cognitive decline indicators.

4. In Chapter V, activities are modelled in a hierarchical model exploiting sub-activities and their relations. After activities are modelled by Recursive Auto-Encoders (RAE), abnormal behaviour reflecting cognitive status of elderly people is detected.

5. In Chapter V, Recursive Auto-Encoders (RAE)-based transfer learning is exploited to cope with the problem of scarcity of data in the context of abnormal

behaviour detection. In the absence of training data, learning the normal behaviour and daily life patterns of a (cognitively) healthy person in a house and transferring this knowledge to another house for the detection of abnormal behaviour is helpful.

6. In Chapter VI, we emulate activity recognition as a graph labelling problem and exploit Graph Convolutional Networks (GCNs) to model activities based on their fine-grained sensor activations and sub-activities. Then abnormal behaviour related to dementia is detected exploiting the relationship between nodes of the graph.

1.7 Structure of the Thesis

The rest of this thesis is organised as follows.

- Chapter II (Literature Overview and Background) provides an overview of the smart home settings focused in this study. Moreover, literature work on data simulation and sensor representation, cognitive assessment, activity recognition and abnormal behaviour detection is presented. Also, the literature work summarising data simulation, sensor data representation, cognitive status assessment and machine learning methods as well as deep learning methods is presented along with transfer learning methods. Also, generative and discriminative methods that are used for comparison with the proposed methods are summarised in this chapter.

- Chapter III (Data Generation) describes the datasets used and then provides information about data simulation with a specific focus on two types indicators of cognitive decline, namely activity and sub-activity related abnormal behaviour.

- Chapter IV (Spatio-temporal Activity Recognition and Abnormal Behaviour Detection) describes how Recurrent Neural Networks and Convolutional Neural Networks are adapted to detect abnormal behaviour related to the indicators of cognitive decline along with the experiments and discussion.
- Chapter V (Abnormal Behaviour Detection using Recursive Auto-encoders and Transfer Learning) describes how sub-activities in an activity are merged hierarchically via Recursive Auto-Encoders (RAEs) and presents experimental results followed by a discussion. Moreover, this chapter describes how transfer learning with RAEs is adapted to transfer knowledge from a source household to a target household to detect abnormal behaviour related to dementia.
- Chapter VI (Fine-grained Activity Recognition and Abnormal Behaviour Detection) describes Graph Convolutional Networks (GCNs) and presents how it is adapted to recognise daily life activities and then detect abnormal behaviour in the context of cognitive decline in a smart home.
- Chapter VII (Conclusions and Future Work) concludes the thesis, summarise the key findings and drawbacks and then outlines future work.

1.8 List of Publications

The contributions of this thesis are presented in the following publications (with some of them still under review or preparation):

1. D. Arifoglu and H. Bouachachia, *Literature Review on Detection of Abnormal behaviour for Dementia Sufferers*, Expert Systems with Applications, to be submitted
2. D. Arifoglu and H. Bouachachia, *Abnormal Behaviour Detection for Dementia Sufferers using Recursive Auto-Encoders*, Expert Systems with Applications,

submitted on September 28, 2019.

3. D. Arifoglu and H. Bouachachia, *Activity Recognition and Abnormal Behaviour Detection with Graph Convolutional Networks for Dementia Sufferers*, Engineering Applications of Artificial Intelligence, submitted on February 26, 2019.

4. D. Arifoglu and H. Bouachachia, *Detection of Abnormal Behaviour for Dementia Sufferers using Convolutional Neural Networks*, Artificial Intelligence in Medicine, Volume 94, Pages 88-95, 2019.

   URL: *www.sciencedirect.com/science/article/pii/S*0933365718300617

5. D. Arifoglu and H. Bouachachia, *Abnormal Behaviour Detection for Dementia Sufferers via Transfer Learning and Recursive Auto-encoders*, 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pages 529-534, Kyoto, March 11-15, 2019.

   URL: *h* − *suwa.github.io/percomworkshops*2019*/papers/p*529 − *arifoglu.pdf*

6. D. Arifoglu and H. Bouachachia, *Activity Recognition and Abnormal Behaviour Detection with Recurrent Neural Networks*, 14th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2017), pages 86-93, Leuven, Belgium, July 24-26, 2017.

   URL: *https* : *//www.sciencedirect.com/science/article/pii/S*1877050917313005

CHAPTER II

Literature Overview and Background

2.1 Introduction

In this research, we aim to detect the early indicators of cognitive decline at a smart home scenario where a resident is observed and his/her activities are tracked with the help of sensors. Thus, in this chapter, we firstly describe the smart home setting including sensor and activity types in Section 2.2 and then summarise literature work in Section 2.3. Moreover, in Section 2.4, we describe the machine learning methods used to compare our proposed methods.

2.2 Smart Home Setting

2.2.1 Sensors

Sensors at a smart home are used to collect various types of data related to activities of the residents, states of the objects and states of the environment. Sensors can capture pressure, position, direction and motion, light, radiation, sound, image and video and state of the object (present, not present) and physiological measurements (e.g. blood sugar, blood pressure) [6].

These sensors can be categorised into two classes based on their state:

- Discrete state sensors where the output is either 0 or 1, where 0 indicates sensor is not triggered and 1 otherwise. Door sensors (closed/open), light sensors (ON/OFF) are examples in this class.
- Continuous state sensors where the output is a real value. Examples of continuous sensors are water sensors and temperature sensors (sensors attached to household appliances like microwave, kettle, toaster, heater and oven).

Sensors are designed to detect changes in the environment. These sensors can be wearable sensors or pervasive sensors attached on objects and in the environment. Wearable sensors are impractical for situations in which individuals are opposed to wearing the sensors. They may forget to wear these sensors which is the case for elderly people with cognitive diseases (e.g. Alzheimer). In contrast, pervasive infrastructure sensors offer the advantage of being non-obtrusive to the users as they are placed in the environment. Moreover, using the latter ones allows the residents to live as normally as possible and not to get distracted by the technology that surrounds them while performing their activities in the natural flow of daily living at home. Therefore, this study will be focusing on pervasive sensors. These sensors will be item, motion and door sensors, which only have ON/OFF status. These sensors are helpful to track the changes in the activity trends of elderly people suffering from cognitive decline.

2.2.2 Activities

Many people with dementia may need help with tasks that are called *Instrumental Activities of Daily Living* (IADLs) such as:

- Managing money (writing cheques, handling cash, keeping a budget, etc.)
- Managing medication (taking the appropriate dose of medication at the right time, etc.)

- Cooking (preparing meals, microwave/stove usage, etc.)
- Housekeeping (performing light and heavy chores such as dusting or mowing the lawn, etc.)
- Using appliances (the telephone, television, vacuum, etc.)
- Shopping (purchasing, discerning between items, etc.)
- Extra-curricular (maintaining a hobby or some leisure activities, etc.)

Instrumental activities of daily living are not necessary for fundamental living, but they let an individual live independently in a community. In this research, we focus on sequential daily life activities, where each activity is performed after another one in a sequential fashion without any interweaving (e.g. cooking, make a phone call and then cooking again). We also assume that there is only one occupant living in a smart house. Cooking, sleeping, going to toilet, working, cleaning, washing dishes, etc. are examples for daily life activities. We focus on these type of activities since they are promising to reflect on cognitive status of elderly people in a scenario of daily living in a smart home.

2.3 Literature Review

In this section, we cover the literature work on data generation (Section 2.3.1), sensor representation (Section 2.3.2) and assessment of cognitive status (Section 2.3.3). Moreover, we summarise deep learning (Section 2.3.4) and transfer learning literature related to our proposed methods (Section 2.3.5).

2.3.1 Data Simulation

In the literature, there is some work dedicated to the synthesis of activity related data [86, 28, 53]. In [28], the authors modified a real-world dataset in order to synthesise health related abnormal behaviour for their experiments. Eight daily activities such as sleeping, waking up, walking, eating are chosen and health related abnormal behaviour like frequent toilet visit, no exercise, slept without dinner are synthesised. In [53], more data is synthesised using Hidden Markov Models (HMMs) based on a small set of real data collected. To increase the realism of data simulation, the sensor events were modelled by a combination of Markov chains and the Poisson distribution. However, in both [28, 53], it is not mentioned in detail how the data synthesis was done. In [86], the authors modified a real-life data set of an older adult converting basically the rooms into activities. The authors focused on walking and eating in conjunction with the sleeping activity and samples of these activities are manually inserted in the XML data set. In [60], abnormal sensor readings are manually identified by a trained expert. A real dataset is taken as a base and synthetic errors were generated.

2.3.2 Sensor Data Representation

The studies in the literature exploit raw, change and last-fired features [83]. The raw sensor representation gives a 1 when a sensor is triggered and gives 0 when that sensor is OFF. Change representation is assigned 1 when a sensor changes its state (from ON to OFF or OFF to ON) and a 0 otherwise. The last-fired sensor representation gives information about which sensor is fired last. The sensor that changed state last continues to give 1 and updated to 0 when another sensor's state is updated. However, these features are extracted from time-slice chunks within a given time and neglect the interaction between sensors, their triggering order and frequency. In [82], the authors try to capture the relationship between the sensor activations by learning an adjacency matrix reflecting the sensor topology in the house. In [19], the authors extract sensor based features such as the duration of the activity and the number of sensors triggered.

In [2], motion sensors in a smart home are represented as nodes in a graph and resident's movements are encoded as edges. If sensor A is activated after sensor B, then there is an edge between the corresponding nodes. Each triggered edge is considered in the data as a feature. If an edge exists in this graph for the current activity, then corresponding edge attributes are assigned as the value in the feature vector. If an edge does not exist in the graph, default value for that feature is zero.

Lundstrom et al. [52] exploited spatio-temporal features where binary events during a period of time are represented in a matrix. In this representation, a certain time interval $I_j = (t_j, t_j + k)$ of a day, where k is the length of the interval and t_j is a time stamp, is considered. A matrix M_j contains all the events for the time interval I_j, where the number of rows is equal to the number of sensors n and the number of columns is given by the number of sampling times during I_j. To reduce the dissimilarity of patterns occurring due to the time shift, data matrices are convolved with a linear function. The convolution operation is applied separately to each row of the matrix M_j.

2.3.3 Cognitive Status Assessment

In-home automatic assessment of cognitive decline has been the subject of many studies [54, 76, 20, 69, 40, 34, 30, 39] and many machine learning approaches such as Support Vector Machines (SVMs) and Naïve Bayes classifier [60, 17], Restricted Boltzmann Machines (RBMs) [34, 15], Markov Logic Networks [69, 40, 30], Hidden Markov Models (HMMs) [28, 41], Random Forest methods [53], Hidden Conditional Random Fields [81] and Recurrent Neural Networks [51].

Current cognitive assessment methods mostly rely on queries from questionnaires or in-person examinations, which depend on recall of events or brief snap-shots of function that may poorly represent a person's typical state of function. Also the clinical methods have some limitations such as their episodic nature, and possible biased reporting. For example, in [54], the assessment scoring is done by asking elderly people to complete a sequence of scripted actions. The participants are monitored via Web camera while they perform tasks. In [57], authors focus on kettle and fridge usage and sleep patterns. The cognitive status of a person is assessed based on the kettle's and fridge's usage time, duration and frequency as well as the duration of sleep. In [87], authors design games in order to assess the cognitive status of an elderly person. In [19], the authors first extract sensor based features (the duration of the activity and the number of sensors triggered) and then assess the activity quality and cognitive status of elderly people. Note that, participants, in this study, are provided with a brief description of each task that they should refer to during the simulated activities. These studies fail to provide an unobtrusive way of assessment since they are not done in the natural flow of daily living and in real life scenarios. Moreover, using rule-based systems, an expert is needed to manually integrate specific rules to the system since every person has own daily life routines. For example, waking up and drinking water in the middle of the night might be normal for a person, while abnormal for some other person. However, our approach does not require any expert knowledge, since it learns what is normal and abnormal from the training data automatically. Specifically, we aim in this study to detect anomalies in the natural flow of daily living without giving any instruction and considering not only some time interval, but everyday living scenario.

In [21], machine learning approaches such as Support Vector Machines (SVMs) and Naïve Bayes are used to assess the cognitive status of elderly. In [34], Parkinson's Disease state assessment in home is explored by means of RBMs using data from body worn sensors. In [70], the authors use Markov Logic Network, which is a probabilistic logic that unifies statistical and symbolic reasoning to detect anomalies. In [21], some instructions to perform some tasks (e.g., sweeping the kitchen and dusting the living room, etc.) are given to the patients who then receive scores after completing those tasks. These scores are calculated based on the time spent, the frequency of the sensor triggered, etc.

In [22], the authors introduce *activity curves* which model daily activity routines of individuals. Abnormal behaviour is detected by comparing these *activity curves* against the actual behaviour. In [7], the authors first introduce an abstraction layer to create a common ground for home sensor configurations. Then, they build a probabilistic spatio-temporal model to summarise daily behaviour. The probabilistic model takes into account the location of the subject at each hour of the day and defines a likelihood of the subject's behaviour based on her location and outings. This model, computed over a long period of sensor data, indicates where the subject spends her time as part of a daily routine. Anomalies, such as staying in bed for a long time or not using the bedroom for sleeping during the night, are then defined as significant changes from the learned behavioural model and detected using a cross-entropy measure. In some studies, the assessment is done by attaching motion sensors on kitchen utilities and observing their usage frequency and time [86, 76]. In [28], the authors exploit Hidden Markov Model (HMM) and fuzzy rules to detect duration, time and frequency related anomalies. In [60], behavioural patterns of the residents are extracted using Bayesian statistics. These patterns are used to detect abnormal behaviour that potentially indicate changes in health status of the user. In [24], a Markov chain model is employed to model the daily routines based on historical data. An entropy rate is calculated to detect the unusual patterns of people with dementia in their day-to-day life.

In [60], behavioural patterns of the residents are extracted using Bayesian statistics and these patterns are used to detect anomalous behaviour signs which reflect changes in health status of the user. Aztiria et al. [10] developed an algorithm that compares the behaviour of a user with a set of previously discovered frequent behaviour to identify possible shifts. Employing a set of atomic actions, the authors defined a likelihood value for the current behaviour of the user. The number of modifications required to turn the current behaviour into a frequent behaviour is used as a metric to classify a sequence of actions as an anomaly.

In [17], the authors propose a method to recognise activities in a smart home and to identify errors and inconsistencies in the performed activity. First, normal activities are modelled by NB and Markov model classifiers and then the closeness of each performance to each activity is measured. The algorithm calculates the model likelihood for the observed sequence of sensor events. If the generated probability falls outside two standard deviations of the mean then the activity is flagged as anomalous, otherwise it is labelled as consistent with normal execution of the activity.

In [81], the authors exploit Hidden State Conditional Random Field (HCRF) method to detect abnormal activities that often occur in homes of elderly by considering sub-activity relations. First HCRF is used to recognise activities by producing a recognition confidence value for each activity. Then a threshold based method is used to decide the activities as normal and abnormal. In [69], the authors propose to detect anomalies by exploiting the Markov Logic network. They use a hybrid technique including supervised learning, rule-based reasoning and probabilistic reasoning. However, steps of each action are defined prior to the construction of the model. The inference engine evaluates the rules (e.g. the patient has taken a medicine that was not prescribed) which are extracted from a medical knowledge base of Mild Cognitive Impairment (MCI) models and indicators. On the other hand, those rules strongly depend on the specific home environment, on the used sensors, and on the particular habits of the elderly; hence, their definition is time-expensive, and rules are not portable to different environments. In order to address this issue, the same authors propose a method to automatically learn the rule-based definitions of behavioural anomalies [40]. They exploit formal rule induction methods and a training set of normal and abnormal behaviour. However, the authors claim that their proposed rule learning method infers deterministic rules, which are prone to generate anomaly mispredictions in the presence of noise from the sensor infrastructure. In our study, we learn normal daily life patterns for each individual from training data automatically and without the integration of any rules. Similar to [81, 69], in our proposed work, we define anomaly not activities alone but in the context of sequences, with other activities happened before and after.

2.3.4 Deep Learning

Recently, there has been growing interest in Convolutional Neural Networks (CNNs) [90, 92, 59, 35, 33, 71, 91, 12, 67, 88], Deep Belief Networks (DBN) [15], Restricked Boltzman Machines (RBMs) [65, 34, 15, 26] and Recurrent Neural Networks (RNNs) [59, 35, 27]. In [65], RBMs are used for feature extraction and selection from sequential data. In [26], results with RBM on CASAS dataset outperformed HMMs and NB in most of the cases. In [59], the authors used a combination of CNNs and Long Short Term Memory (LSTM) Recurrent Neural Networks to do multi-modal wearable activity recognition. In [35], the authors explore deep, convolutional and recurrent approaches on movement data captured with wearable sensors. In [12], the authors utilised convolutional networks to classify activities using time-series data collected from smart phone sensors. In [67], in a real world setting, an automatic stereotypical motor movement in Autism detection systems is developed exploiting CNNs. The discriminating features from multi-sensor accelerometer signals are learnt via CNNs and this knowledge is transferred to a new dataset. In [88], CNNs are exploited to learn features from raw physiological signals in an unsupervised manner analysis and then using multivariate Gaussian distribution, anomalies are detected to identify latent risks.

CNNs have been exploited for activity recognition using movement datasets that are generated by wearable sensors [65, 90, 92, 59, 33, 12, 15]. Except the work by

Fang et al. [26], none of these studies focus on daily activity datasets collected by sensors placed at home. Previous work on activity recognition based on wearable sensor datasets shows that CNNs and RNNs are useful to recognise activities, but leaves a lot of room for improvement.

Interesting results are obtained with recursive models in hierarchical learning problems such as parsing, sentence-level sentiment analysis and paraphrase detection as well as scene parsing [74, 64, 75]. Auto-encoders are being used for anomaly detection on time-series data [94, 72]. In [75], the authors use Recursive Auto-Encoders for predicting sentiment distributions. Instead of using a bag-of-words model, this model exploits a hierarchical structure and uses compositional semantics to understand sentiment.

One evidence that there are granular-level information in daily life activities is provided by Zhang et. al on the the the Aruba dataset [93]. They extract movement patterns for representing the occupant's moving segments during the performance of an activity. It is shown that the movement vector can distinguish different high-level activities. The occupant tends to have the same routine of performing the same activities, but has different movement patterns in different activities. For example, the occupant may mainly move around the kitchen sink in *wash dishes* activity, and stay around the bedroom area during *sleeping* activity. A combination of some motion sensors are mostly seen in the instances of *relax* activity while some other motion sensors indicates the movement between kitchen range and the sink and this movement pattern can be seen in the instances of *wash dishes*.

Graph-structured data appears frequently in domains such as chemistry, natural language semantics, social networks, and knowledge bases [23, 73, 85]. Graph convolutions have been widely used to learn high-level features by considering spatio-temporal relationships among nodes of a graph. In [58], graph kernels are used to embed meaningful local neighbourhoods of the graphs in a continuous vector space.

In [80], the authors represent graphs as multi-channel image-like structures that allows them to be handled by 2D Convolutional Neural Networks (CNNs). In [50], molecules are represented as an undirected Graph Convolution Networks (GCNs) to predicate molecular properties. Moreover, in anomaly detection literature, graph-based methods are preferred when there is inter-dependent data since graphs can offer powerful representation abilities [1, 93, 2]. The most similar approach to ours is [2], where motion sensors in a smart home are represented in a graph and resident's movements are encoded as edges in the graph. Then graph-based features are extracted and used as input for a SVM. In this study, each sensor is represented as a node which appears only once. If sensor A is activated after sensor B, then there is an edge between their nodes. However, in our case instead of taking sensors as nodes, we encode each activation as a node, which allows us to capture the further ordering of sensor activations.

2.3.5 Transfer Learning

In transfer learning literature, most of the activity recognition models are supervised models that require labelled data to learn the model parameters [63, 13, 43]. Good results are obtained using generative models such as HMM [63, 43] and discriminative models such as CRF [13, 43]. In [84], a method is proposed to learn the parameters of a HMM using labelled data from the source domain, and unlabelled data from the target domain. The study ignores the activities' important features such as the activity structure and related temporal features. They also assume that the structure of HMMs is given and pre-defined. Later they extend this work to learn hyper-parameter priors for HMM instead of learning the parameters directly [42].

In transfer learning, sensors and activities in different households are needed to be mapped. In [84], a comparison of feature mappings was done. The mapping that combined sensor readings in a single feature based on their function (e.g. sensors used during cooking) gave the best results. In some cases, meta-features are first manually introduced into the feature space and then the feature space is automatically mapped from the source domain to the target domain [42].

In [68], the authors first assign a location label to each sensor indicating in which room or functional area the sensor is located. Then activity templates are constructed from the data for both the source and target data, finally a mapping is learnt between the source and target datasets based upon the similarity of activities and sensors.

2.4 Background

In this section, we will give a brief summary of the state-of-the-art generative (Naïve Bayes, Hidden Markov Model, Hidden Semi-Markov Model) and discriminative (Conditional Random Field, Support Vector Machines) methods that we use for comparison in later chapters.

2.4.1 Generative Methods

2.4.1.1 Naïve Bayes

Naïve Bayes classifier does not consider the temporal dependency between input instances. In this method, all instances are assumed to be independent and identically distributed. The joint probability over instances are calculated as follows.

$$p(y_{1:T}, X_{1:T}) = \prod_{t=1}^{T} p(\overrightarrow{x}|y_t)p(y_t) \tag{2.1}$$

where $p(y_t)$ is the prior probability over an activity instance. It indicates the probability of that activity without any observation taken into account. $p(\overrightarrow{x}|y_t)$ is the observation distribution that represents the probability that the activity y_t would generate observation vector $\overrightarrow{x}$. According to Naïve Bayes assumption, each sensor reading is considered separately, which results in N parameters for each activity, where N is the number of sensors. Then, each sensor observation is modelled as an independent Bernoulli distribution and observation distribution for activity i becomes

$$p(\overrightarrow{x}|y_t = i) = \prod_{n=1}^{N} p(x_t^n|y_t = i) \tag{2.2}$$

2.4.1.2 Hidden Markov Model

In this model, Markov assumption is used to model the temporal relation between consecutive time slices. In the first order Markov assumption, the hidden variable y_t, at time t, depends on the previous hidden variable y_{t-1} at time $t-1$. In the second order Markov assumption, the observable variable $\overrightarrow{x}$ at time t depends only on the hidden variable y_t at that time slice. Then the joint probability becomes

$$p(y_{1:T}, X_{1:T}) = \prod_{t=1}^{T} p(\overrightarrow{x}_t|y_t)p(y_t|y_{t-1}) \tag{2.3}$$

In the observation model $p(\overrightarrow{x}_t|y_t)$, the same assumptions with the Naïve Bayes are used. The transition probability $p(y_t|y_{t-1})$ represents the probability of going from one state to another.

HMMs are more suitable for datasets that require temporal information encoding since it can relate each input with previous one.

2.4.1.3 Hidden Semi-Markov Model

Semi-Markov models relax the Markov assumption by explicitly modelling the duration of an activity. The HMM models the duration of activity implicitly by means of self-transitions of states, but this introduces some limitations. Hidden Semi-Markov Model (HSMMs) use an additional variable d_t to model the duration. This variable represents the remaining duration of state y_t. The value of this variable decreases by one at each time-step and the model stays at that state until this variable becomes zero. When it becomes zero, a transition to a new state is made and the duration of the new state is obtained from the duration distribution. Then the joint probability of the HSMM is

$$p(y_{1:T}, X_{1:T}, d_{1:T}) = \prod_{t=1}^{T} p(\overrightarrow{x}_t|y_t)p(y_t|y_{t-1}, d_{t-1})p(d_t|d_{t-1}, y_t) \tag{2.4}$$

2.4.2 Discriminative Methods

2.4.2.1 Conditional Random Field

NB, HMM and HSMM models learn the parameters by maximising the joint probability. Conditional random fields do not model the full joint probability, but model the conditional probability. A CRF which uses the first order Markov assumption is called a linear-chain CRF and resembles to HMM in terms of structure. In this research, linear-chain CRF is used, in which the conditional distribution is

$$p(y_{1:T}|X_{1:T}) = 1/(Z(X_{1:T})) \prod_{t=1}^{T} exp \sum_{k=1}^{K} \phi_k f_k(y_t, y_{t-1}, \overrightarrow{x}_t) \tag{2.5}$$

where K is the number of feature functions used to parameterise the distribution, ϕ is a weight parameter and $f_k(y_t, y_{t-1}, \overrightarrow{x}_t)$ is a feature function. The product of the parameters and the feature function is called energy function and the exponential of that term is called a potential function. $Z(X_{1:T})$ is called the partition function and used as a normalisation term. It ensures that the distribution sums up to one and obtains a probabilistic interpretation. It is calculated by summing over all possible state sequences.

2.4.2.2 Support Vector Machines

A Support Vector Machine (SVM) is a supervised machine learning algorithm which is based on the idea of finding a hyperplane that best divides a dataset into two classes, as shown in Figure 2.1. The data points nearest to the hyperplane are called support vectors. if these support vectors are removed, the position of this hyperplane changes, which affects the classification results.

SVMs maximise the margin around the separating hyperplane to classify the instances in a dataset. The margin is the distance from the hyperplane to support vectors. Different optimisation methods can be used to find the optimal hyperplane by maximising the margin. For this purpose, SVMs use kernel functions such as radial basis kernel, polynomial kernel to map the non-linearly separable data from the input space into a higher space where data become linearly separable.

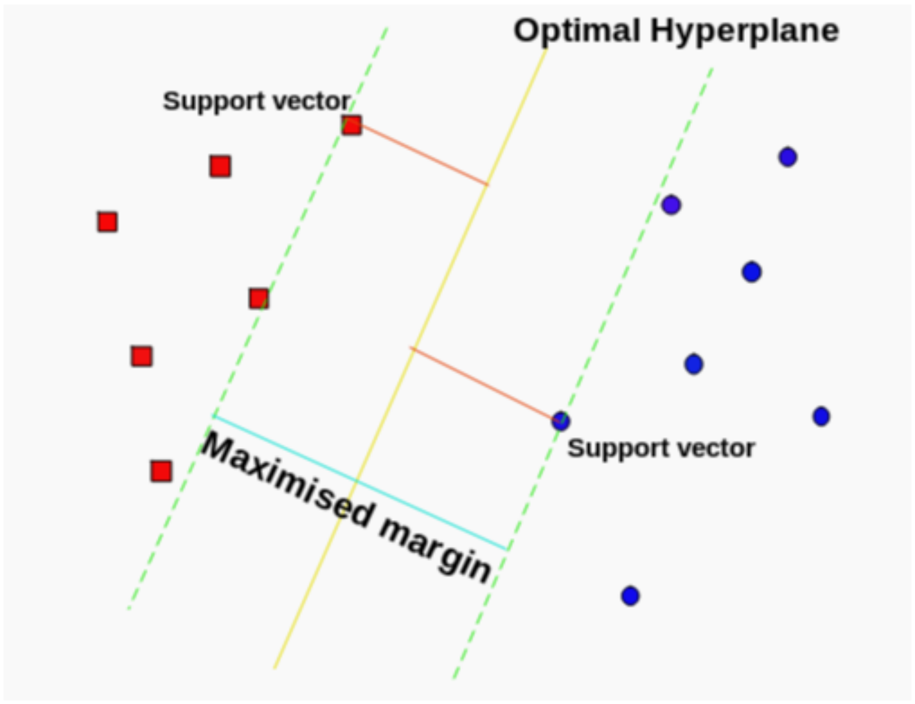


Figure 2.1: Hyperplane and support vectors in SVM algorithm.

CHAPTER III

Data Generation

3.1 Introduction

In this chapter, we describe the datasets used to evaluate our proposed methods. These datasets are namely Van Kasteren and CASAS datasets, which include daily life activities collected from pervasive sensors in smart homes. These datasets are chosen because they consist of activities that can be observed in an elderly person's daily life. However, these datasets don't include any abnormal behaviour reflecting on cognitive status of elderly people. Moreover, in this chapter, we describe our method to simulate abnormal behaviour of elderly people suffering from dementia.

This chapter is organised as follows. The datasets used are described in Section 3.2 before the details about generation of artificial activities are introduced in Section 3.3.

3.2 Datasets

3.2.1 Van Kasteren Datasets

The first dataset used in this research is the popular dataset collected by Van Kasteren [83]. The data captures daily-life activities such as sleeping, cooking, leaving home, etc. using sensors placed at the homes in less than a month. In this dataset, there are 3 households which are denoted as dataset A, B and C (see Figure 3.1).

These households are equipped with motion and door sensors. These sensors are placed on toilet, next to stove, kitchen sink, bathroom sink, bathroom tube, bed, working desk and on doors in the rooms. A more detailed overview of these households are provided in Table 3.1. For example, 14 sensors are used in household A and there are 10 different daily life activities involved.

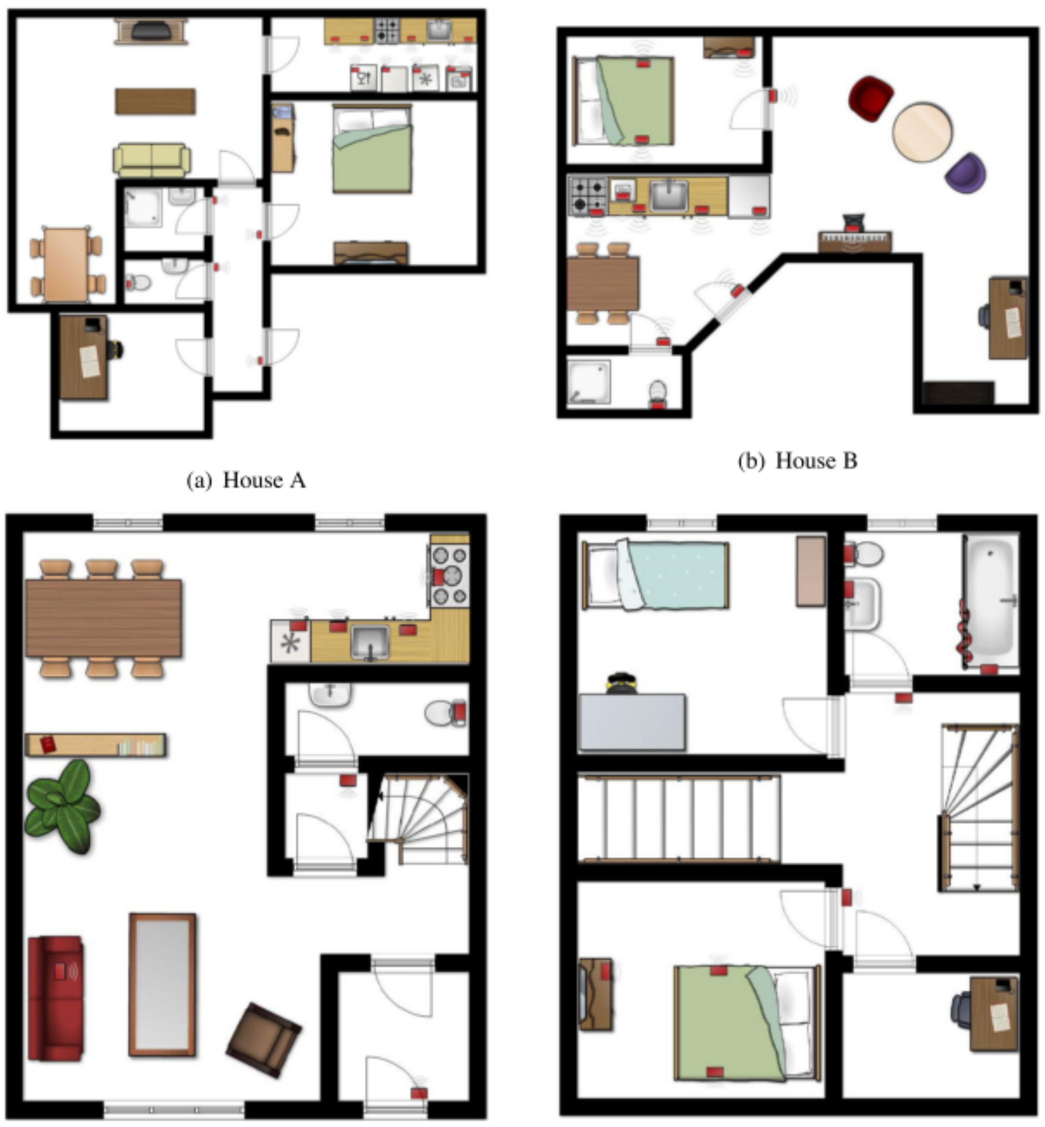

(a) House A
(b) House B
(c) House C, First floor
(d) House C, Second floor

Figure 3.1: Van Kasteren dataset.

An excerpt from raw sensor measurements is depicted in Table 3.2. Here, each

Table 3.1: Some statistics of Van Kasteren dataset.

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

measurement is associated with date and time, sensors ID and sensor status (ON/OFF) respectively.

Table 3.2: Raw sensor reading data.

Date	Time	Sensor ID	Sensor Status
2011-04-01	01:16:10.814699	M004	ON
2011-04-01	01:16:11.429192	M007	ON
2011-04-01	01:16:16.462383	M004	OFF
2011-04-01	01:16:16.599859	M005	ON
2011-04-01	01:16:19.899843	M003	ON
2011-04-01	01:16:22.102316	M005	OFF

3.2.2 CASAS Datasets

We used two testbeds of CASAS smart home project [17], namely Aruba [16] and WSU testbeds.

In Aruba testbed, motion, door and temperature sensors are used. However, we exclude temperature sensors in our study since they do not bring any additional information about the cognitive status of elderly. Thus, in total we used 34 sensors (3 door and 31 motion sensors). The data is provided as a list of (sensor, time-stamp) sensor measurements. There are 11 daily activities performed by a single user spanning 224 days. These activities are *Meal Preparation* (1606 instances), *Relax* (2910 instances), *Eating* (257 instances), *Work* (171 instances), *Sleeping* (401 instances), *Wash dishes* (65 instances), *Bed to toilet* (157 instances), *Enter home* (431 instances), *Leave home* (431 instances), *Housekeeping* (33 instances) and *Respirate* (6 instances).

In WSU testbed, there are 5 activities, which are *Make a phone call*, *Hand washing*, *Meal Preparation*, *Eating*, *Cleaning.* There are 20 instances of each activity performed by 20 students in both *adlerror* and *adlnormal* versions. *adlnormal* version consists of totally normal behaviours while in *adlerror*, there are specific errors in the task completion of these activities. Errors were selected to reflect common difficulties that can compromise everyday functional independence. The participants are told to include these errors during their performance. These errors can be seen in daily life activities and activity patterns of elderly people who are suffering from the consequences of cognitive decline.

3.3 Data Generation

We focus on two kinds of anomalies that can be seen in daily life routines of elderly people with dementia: 1) *activity* related anomalies and 2) *sub-activity* related anomalies. In *activity* related anomalies, an activity itself is totally normal while there is an anomaly related to its frequency or its timing in a day. On the other hand, *sub-activity* related anomaly is related to the context and the quality of activity performed such as frequency of sensor activations involved as well as their order and correlation. In the first one, activities as a whole are repeated or forgotten (e.g. having dinner); while in the second one, some steps (sensor activations) of activities are forgotten or repeated (e.g. adding salt to the dish).

3.3.1 Activity Related Abnormal Behaviour

Frequency sensitive activities such as *having a snack or drink*, *brushing teeth*, *taking medicine multiple times*, etc. are the ones where only the number of occurrences matters in terms of medical assessment. To simulate these type of indicators, we insert a specific set of activities within the whole day sequence of activities (see Algorithm 1). This will result in multiple occurrences of the same activity. Moreover, insertion in some inadequate time of the day will generate time-related abnormality such as having dinner in the middle of the night. We insert the instances of the following activities: *preparing meal, eating, working, washing dishes, leaving home, entering home* into the normal activity sequences of the Aruba dataset.

We simulate sleep disorders and night time wandering anomalies by inserting some synthetic activities in the normal night-time activity sequences of a person. More specifically, we insert *eating, bed to toilet, respirate* into the *sleeping* activity of normal activity sequences. This will emulate the activities of getting drink and going to the toilet frequently in the middle of the night.

```
**Input:** A sequence $S$ of sensor activations in a day such as
        $S = < s_1, s_2, \dots, s_n >$ where each $s_i$ is a sensor activation.
        An activity $A = < a_1, a_2, \dots, a_m >$ where each $a_j$ is a sensor
    activation. /* $A$ is chosen specially (e.g. *eating*) to reflect a
    dementia related abnormal behaviour.                              */
**Output:** $S = < s_1, s_2, \dots, s_l, a_1, a_2, \dots, a_m, s_{l+1}, \dots, s_n >$
while true do
    Choose a random position $l$ in $S$;
    Insert $A$ into $S$ at position $l$;
end
```

Algorithm 1: Simulation of *activity* related abnormal behaviour

As described in Algorithm 1, all insertions are done randomly. Firstly a random instance of a given activity type (for example *meal preparation*) from whole dataset is chosen, and then it is injected in a random location. Please note that these activities

Table 3.3: Examples of abnormal behaviour.

Type	Original	Modified	Abnormality
Activity	S - M - E	S - **B** - S - M - E	Sleep disorder
	S - M - E	S - **R**- S - M - E	Sleep disorder
	S - M - E	S - **R** - S - M - E	Sleep disorder
	S - M - E	S - **M** - S - M - E	Repetition
	S - M - E - H	S - M - E - H - **E** - **E**	Repetition
Sub-activity	W: M_{26}, M_{28}, M_{27}	M_{26}, M_{28}, $\boldsymbol{M_{26}}$, $\boldsymbol{M_{26}}$, M_{27}	Confusion
	B: M_4, M_5, M_7	M_4, $\boldsymbol{M_5}$, $\boldsymbol{M_5}$, M_7, $\boldsymbol{M_5}$	Confusion
	W: M_{18}, M_{20}, M_{15}	M_{18}, $\boldsymbol{M_{15}}$, $\boldsymbol{M_{15}}$, M_{20}, M_{15}	Confusion
	E: M_{14}, M_{19}, M_{18}	M_{14}, $\boldsymbol{M_{14}}$, $\boldsymbol{M_{14}}$, M_{19}, M_{18}, $\boldsymbol{M_{14}}$	Confusion
	M : M_{18}, M_{19}, M_{15}	M_{18}, $\boldsymbol{M_{18}}$, $\boldsymbol{M_{18}}$, M_{19}, M_{15}, $\boldsymbol{M_{18}}$	Confusion

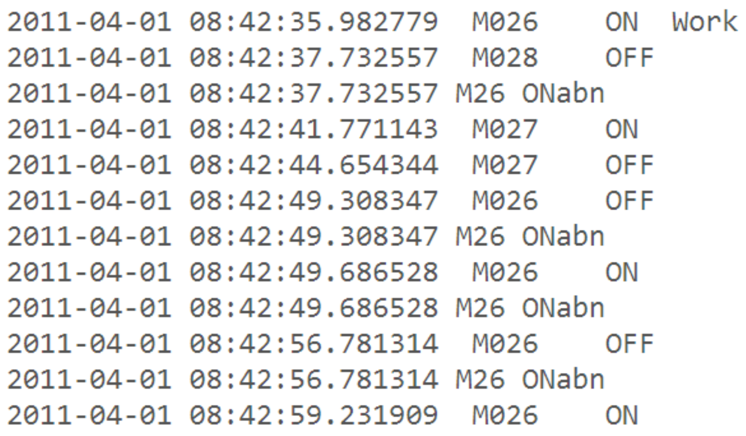
are totally normal on their own but become abnormal when they occur at a wrong time of the day and after or before a specific activity. Hence, capturing these abnormalities within the context is important. In all, we manually generate 77 abnormal activity instances on the the Aruba dataset. A set of modified abnormal behaviours is depicted in Table 3.3. The following abbreviations are used for the activities. *S: Sleeping, M: Meal preparation, E: Eating, R: Respirate, W: Working, B: Bed to toilet.* The inserted activities are shown in bold. For *sub-activity* related abnormal behaviour, because of space problem, only a subset of sensor activations are shown, where each M corresponds to a motion sensor.

Moreover, we generate these abnormal activity instances on dataset A of Van Kasteren dataset which has the following 9 activities*Leave house, use toilet, take shower, brush teeth, go to bed, prepare breakfast, prepare dinner, get snack, get drink*. In all, we manually synthesised 135 abnormal activity slices in this dataset.

3.3.2 Sub-activity Related Abnormal Behaviour

This kind of abnormal behaviour is generated by repeating specific sensor activations in a given activity. For this purpose, given random instances of *working, eating, meal preparation, bed to toilet*, we randomly insert specific sensors ($M_{26}, M_{14}, M_{18}, M_4$

respectively) involved in these activities (see Algorithm 2). For example, for *working* activity, the sensor M_{26} is repeated more than usual which can be used to emulate the usage of a computer. A snapshot for *sub-activity* related abnormal behaviour synthesis is depicted in Figure 3.2, where $ONabn$ shows the inserted sensor activations.



```
2011-04-01 08:42:35.982779  M026    ON  Work
2011-04-01 08:42:37.732557  M028    OFF
2011-04-01 08:42:37.732557 M26 ONabn
2011-04-01 08:42:41.771143  M027    ON
2011-04-01 08:42:44.654344  M027    OFF
2011-04-01 08:42:49.308347  M026    OFF
2011-04-01 08:42:49.308347 M26 ONabn
2011-04-01 08:42:49.686528  M026    ON
2011-04-01 08:42:49.686528 M26 ONabn
2011-04-01 08:42:56.781314  M026    OFF
2011-04-01 08:42:56.781314 M26 ONabn
2011-04-01 08:42:59.231909  M026    ON
```

Figure 3.2: Sensor data after *sub-activity* related abnormal behaviour is generated.

Moreover, we use *adlerror* set of WSU dataset since confusion and forgetting anomalies are reflected in this set. Some examples are leaving the water running after washing hands, leaving the burner on after cooking the oatmeal, forgetting to take medication with the meal, wiping off the dishes without using running water to clean them.

Please note that there is a difference between the first type and the second type of abnormal behaviour. The first one represents anomalies related to forgetting and repetition of activities while the second one represents anomalies related to confusion. The repetition in the first one occurs at activity level, the activities itself as a whole are repeated or forgotten. On the other hand, in the second abnormal behaviour, some sub-activities of individual activities are forgotten or repeated. For example; having dinner twice or forgetting to have dinner reflects the first type of abnormal behaviour, whereas forgetting to add salt to dinner or taking a medicine more than once after dinner is in the second type of abnormal behaviour category.

```
Input: A sequence of $S$ of sensor activations of an activity $A$ such as
        $S = < a_1, a_2, \ldots, a_n >$ where each $a_i$ is a sensor activation
        A sensor type $M$ that occurs in activity $A$.
        /* $A$ and $M$ are chosen specially (e.g. sensor $M_6$ in
    working activity) to reflect a dementia related abnormal
    behaviour.                                                   */
Output: $S = a_1, M, a_2, M, a_3, \ldots, M, a_n$
while true do
    Choose a random location $l$ in $S$
    Insert $M$ into $S$ at $l$
end
```

Algorithm 2: Simulation of *sub-activity* related abnormal behaviour.

3.4 Conclusion

In this chapter, we presented two different types of indicators reflecting the cognitive status of elderly people. These indicators are named as activity (repetition and forgetting of activities) and sub-activity (confusion and repetition of steps withing activities) related anomalies. Moreover, we presented a data simulation method to generate these kind of indicators given the scarcity of data reflecting these anomalies. However, this simulation method generalises the abnormal behaviour reflecting cognitive status of elderly people since some of the abnormal behaviour might be user specific and may not be reflected in the abnormal behaviour generated. This kind of abnormal behaviour can be taken into account by collecting real-world dataset.

CHAPTER IV

Spatio-Temporal Activity Recognition and Abnormal Behaviour Detection

4.1 Introduction

Sequence modelling has been a challenging problem in machine learning that requires models which are able to capture temporal dependencies. Recurrent Neural networks have shown promising results in some sequence modelling problems such as speech recognition [31], machine translation [77, 11], handwriting recognition and generation or translation, language modelling and protein secondary structure prediction [32]. Daily life activities can be modelled as sequences where sensor activations form a time-series data. Thus, modelling activity recognition as a sequence labelling problem makes RNNs an appealing approach. In the present study, we exploit RNNs to model activities based on their temporal information and then detect abnormal behaviour deviating from normal patterns.

Moreover, Convolutional Neural Networks (CNNs) are popular due to their ability to learn fruitful representations and capture local dependency and spatial information of granular-level patterns. For example, in image recognition, CNNs firstly detect pixels, then edges and shapes, then parts of objects as the layer level increases. Similar to images, there are granular-level patterns in daily life activities as described in

Chapter I. CNNs are good at modelling these granular-level patterns and defining their relationship with each other by using spatial information. Thus, in this research, CNNs are exploited to model sensors and their relationship with each other in daily life activity recognition.

In this chapter, we will first summarise Recurrent Neural Networks (RNNs) and their variants in Section 4.2 and Convolutional Neural Networks (CNNs) in Section 4.3. Moreover, Section 4.4 describes how these methods are adapted for activity recognition and abnormal behaviour detection. Experiments are presented in 4.5 along with discussion of the results.

4.2 Recurrent Neural Networks

RNNs can be considered as a deep neural network (DNN) with indefinitely many layers when they are folded out in time. For RNNs, the primary function of the layers is to introduce memory, not hierarchical processing. New information is added in every layer and the network can pass this information on for an indefinite number of network updates, essentially providing RNN with unlimited memory depth. Whereas in DNNs, input is presented only at the bottom layer, and output is produced only at the highest layer. RNNs generally receive input and produce output at each time step. RNNs can learn during the training phase to select what information they need to pass onwards, and what they need to discard [36].

In traditional feed-forward neural networks, it is assumed that all inputs and outputs are independent of each other, but RNNs have a recurrent hidden state whose activation at each time is dependent on that of the previous time. This architecture is recurrent as some of the connections within the network form a directed cycle, where the current time-step t depends on the previous time-step $t-1$. RNNs share parameters for different time-steps which enables them to be used in sequential data with a variable-length. Another way to think about RNNs is that they have a memory (recurrent hidden state) which captures information about what has been calculated so far.

In the following, we will give a brief summary of the three variants of RNNs, which are namely Vanilla RNNs, Long Short Term (LSTM) RNNs and Gated Recurrent Unit (GRU) RNNs.

4.2.1 Vanilla Recurrent Neural Networks

Vanilla RNNs are the traditional RNNs which don't have any gates like in LSTM RNNs or GRU RNNs. In traditional RNNs, given a sequence $x = (x_1, x_2, x_3, ..., x_n)$, the recurrent hidden state h_t is updated by

$$h_t = F_\theta(h_{t-1}, x_t) \tag{4.1}$$

where F_θ is a linear regression function, h_{t-1} is the previous hidden state. From Equation 4.1, it can be inferred that each hidden state h_t is a function which summarises all previous inputs. As depicted in Figure 4.1, x_t is the input at time step t, s_t is the hidden state at time step t which is the memory of the network since it is calculated based on the previous hidden state and the input at the current step, o is output at time step t. RNN shares the same parameters (U, V, W) across all steps, which reduces the total number of parameters we need to learn.

Unfortunately, it has been observed by [49] that it is difficult to train RNNs to capture long-term dependencies because the gradients tend to either vanish (most of the time) or explode (rarely, but with severe effects). This makes gradient-based optimisation method struggle, not just because of the variations in gradient magnitudes but because of the effect of long-term dependencies is hidden (being exponentially smaller with respect to sequence length) by the effect of short-term dependencies.

One of the methods to solve this problem was presented by a recurrent unit (LSTM unit) in [37]. More recently, another type of recurrent unit (GRU) was proposed by

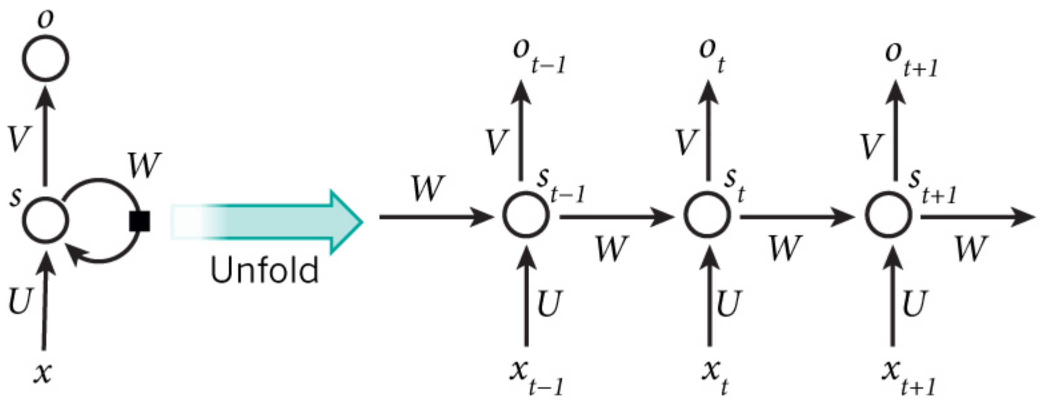


Figure 4.1: Unfolded RNN.

Bengio et al. [14]. RNNs employing either of these recurrent units have been shown to perform well in tasks that require capturing long-term dependencies.

4.2.2 Long Short Term Memory Unit

Each LSTM unit keeps track of an internal state that represents its memory. Over time the cells learn to output, overwrite, or null their internal memory based on their current input and the history of past internal states, leading to a system capable of retaining information across hundreds of time-steps [37]. LSTM blocks have 3 gates to control the flow of information into or out of their memory. An *input gate* controls the extent to which a new value flows into the memory. A *forget gate* controls the extent to which a value remains in memory. An *output gate* is used to compute the output activation of the block. As depicted in Figure 4.2, LSTM can be described as the input signals x_t at time t, the output signals y_t, the forget gate f_t, the input gate i_t and the output gate o_t.

Unlike a recurrent unit which simply computes a weighted sum of the input signal and applies a nonlinear function, each j^{th} LSTM unit maintains a memory c_t^j at time t. The output h_t^j or the activation of the LSTM unit becomes

$$h_t^j = \sigma_t^j \tanh(c_t^j) \tag{4.2}$$

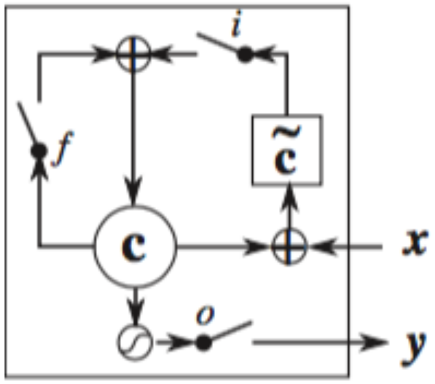


Figure 4.2: A Long Short Term Memory unit

where σ_t^j is an output gate that modulates the amount of memory content exposure. The output gate is computed by

$$\sigma_t^j = \sigma(W_0 x_t + U_0 h_{t-1} + V_o c_t) \tag{4.3}$$

where σ is a logistic sigmoid function. U, W, V are weights where U maps input unit to hidden unit, W maps hidden unit to another hidden unit and V maps hidden unit to output unit.

The memory cell c_t^j is updated by partially forgetting the existing memory and adding a new memory content $\tilde{c_t}^{\tilde{j}}$

$$c_t^j = f_t^j c_{t-1}^j + {i_t}^j \tilde{c_t}^j \tag{4.4}$$

where the new memory content is

$$\tilde{c_t}^{\tilde{j}} = \tanh(W_c x_t + U_c h_{t-1})^j \tag{4.5}$$

The extent to which the existing memory is forgotten is modulated by a forget gate f_t^j, and the degree to which the new memory content is added to the memory cell is modulated by an input gate i_t^j. Gates are computed by

$$f_t^j = \sigma(W_f x_t + U_o h_{t-1} + V_i c_{t-1})^j \tag{4.6}$$

$$i_t^j = \sigma(W_i x_t + U_i h_{t-1} + V_f c_{t-1})^j \tag{4.7}$$

Unlike traditional recurrent unit which overwrites its content at each time-step, an LSTM unit is able to decide whether to keep the existing memory via the introduced gates. Intuitively, if the LSTM unit detects an important feature from an input sequence at early stage, it easily carries this information over a long distance, hence, capturing potential long-term dependencies.

4.2.3 Gated Recurrent Unit

Bengio et al. [14] recently proposed Gated Recurrent Unit (GRU), which is like LSTM but it has fewer parameters than LSTM, as GRUs lack an *output gate*. In GRU, each hidden unit has two gates, which are called *update* and *reset gates*. GRU also controls the flow of information to prevent vanishing gradient problem, but without having to use a memory unit. As depicted in Figure 4.3, GRU can be described in terms of two internal variables z and r, where z is an *update gate* and r is a *reset gate*, x is input and y is output.

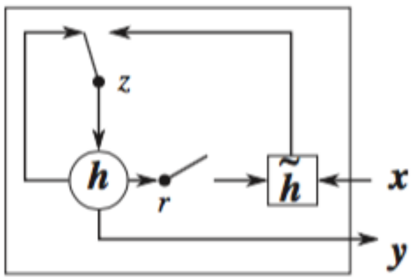


Figure 4.3: A Gated Recurrent unit.

Each hidden state h_t at time-step t is computed as follows:

$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t \tag{4.8}$$

where $\circ$ is an element wise product, z_t is *update gate* and $\tilde{h}_t$ is the candidate activation which given as follows:

$$\tilde{h}_t = tanh(Wx_t + U(r_t \circ h_{t-1})) \tag{4.9}$$

where r_t is the *reset gate*. Both *update* and *reset gates* are computed using a Sigmoid function. Thus, z_t and r_t become

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{4.10a}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{4.10b}$$

4.3 Convolutional Neural Networks

Convolutional neural network (CNN) is a feed-forward deep neural network which involves convolution operation in some layers. CNN typically consists of a combination of three different layers; convolutional layer, subsampling layer and fully-connected layer.

In a convolutional layer, convolution is applied which reduces parameters by following a weight sharing technique. This also reduces the training time and avoids an overfitting issue. In subsampling, we apply non-linear function and pooling operation to obtain high-level distortion-invariant features. After alternately applying convolutional and subsampling layers multiple times, shallow fully connected-layer is applied at the end. For classification task, softmax function is usually used here, and the output layer as one neuron per each class. One benefit of using CNN is that it automatically learns features without any prior knowledge about the data.

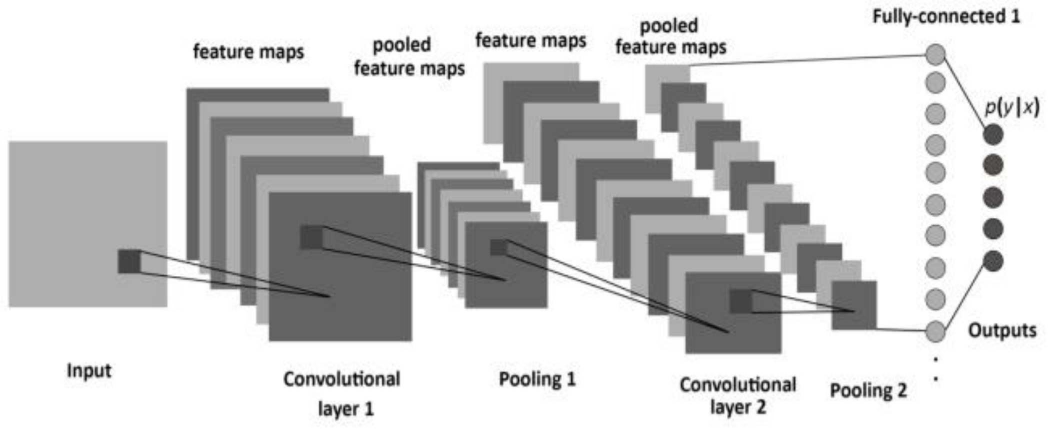


Figure 4.4: An example of convolution layers in a CNN network.

CNN network takes inputs of dimensions $h \times w \times d$, where h is the height of the input matrix, w is the width of the input matrix and d is the depth or the number of different channels of the input matrix. In our study, d is 1 since time-slice input matrices has only one channel.

In a convolutional layer, a local filter (kernel) is used as a feature detector in order to extract the fruitful feature patterns on the given input (see Figure 4.4). These filters capture the local dependencies in the input. Different filters generate different feature maps from the same original input. The more filters, the more features get extracted, the better the network becomes at recognising patterns in unseen instances. The value of these filters are initialised randomly in the beginning and then CNN learns these values on its own during the training process by optimising the values. In this research, random uniform initialisation is used to initialise the filters and Stochastic gradient descent is used to optimise the values during the training.

In this research, time-series data is convolved with both 1-D convolution and 2-D convolution. To obtain distinguishable features from multiple sensors, spatial dependency over sensors and local dependency over time are both important. However, CNNs using 1D convolution kernel and 1D pooling kernel simply capture local dependency over time of each signals. To capture both kinds of dependencies, 2D convolution kernel and 2D pooling kernel will be used. In the following, we will describe these convolutions.

4.3.1 1-D Convolution

1D convolution on position (i, j) of k^{th} feature map in l^{th} layer (convolutional layer), $z_{i,j}^{l,k}$ is computed as

$$z_{i,j}^{l,k} = \sum_{k=1}^{K} \sum_{y=1}^{Y} w_{y,k}^{l-1,k} z_{i,j+u-1}^{l-1,k} \tag{4.11}$$

where Y is a convolution kernel size over time, K is the number of feature maps at $(l-1)^{th}$ layer and $w^{l-1,k} \in \mathbb{R}^{YxK}$ is a weight matrix that form a convolution kernel. 1D pooling of k^{th} feature map in l^{th} layer (subsampling layer) is

$$z_{i,j}^{l,k} = \sigma(\max_{(j-1)Q+1 \leq j' \leq jQ} z_{i,j'^{l-1,k}}) \tag{4.12}$$

where Q is the size of the pooling kernel, and $\sigma(.)$ is an activation function. However, 1-D convolution along temporal dimension can only capture the dependency between activity slices in time. To capture the relationship between the sensors, 2-D convolution is applied on both temporal dimension and feature dimension.

4.3.2 2-D Convolution

2-D convolution on position (i, j) of k^{th} feature map in l^{th} layer is

$$z_{i,j}^{l,k} = \sum_{k'=1}^{K'} \sum_{x=1}^{X} \sum_{y=1}^{Y} w_{x,y,k'}^{l-1,k}, z_{i+x-1,j+y-1}^{l-1,k'} \tag{4.13}$$

where X and Y are a convolution kernel size over signals and over time respectively,

K' is the number of feature maps at $(l-1)^{th}$ layer, and $w^{l-1,k'} \in \mathbb{R}^{XxYxK'}$ is a weight tensor that form a convolution kernel.

Using these filters are not only good for extracting meaningful patterns in the input but also good at reducing the computational complexity since it reduces the dimensionality of the input space. The output of such a set of local filters constitute a feature map (activation map). An additional operation called activation function has been used after every convolution operation. While other functions such as *tanh* or *sigmoid* can be used, generally Rectified Linear Unit (ReLU) which is a non-linear operation is preferred. The resultant map is called Rectified feature map or activation (feature) map.

Once a feature is detected in the convolutional layer, its exact location becomes less important as long as its position related to other features is preserved and additional layer which performs max-pooling is stacked over convolutional layer to reduce the sensitivity of the output. Max-pooling layer achieves scale-invariant feature preservation. A spatial neighbourhood is defined in the rectified feature map within that window. Instead of taking the largest element we could also take the average or sum of all elements in that window, but generally max pooling is preferred.

A window of $n \times t$ is slided over the feature map by shifting by s cells (which are called stride) and the maximum value in each region is taken. This process reduces the spatial size of the input representation. Moreover, it reduces the number of parameters and computations in the network, therefore, controlling overfitting. It also makes the network invariant to small transformations, distortions and translations in the input image (a small distortion in input will not change the output of pooling since we take the maximum or average value in a local neighbourhood).

The fully connected layer used in our network is a traditional Multi Layer Perceptron that uses a softmax activation function in the output layer. The output from the convolutional and pooling layers represent high-level features of the input image.

The purpose of the fully connected layer is to use these features for classifying the input image into various classes based on the training dataset.

CNN can contain one or more pairs of convolutional and max-pooling layers, where higher layers use broader filters to process more complex parts of the input. The top layers in CNN are stacked by one or more fully connected normal neural networks. These fully connected neural networks are expected to combine different local structures in lower layers for final classification purpose. In the training stage, CNN parameters are estimated by standard forward and backward propagation algorithms to minimise the objective function.

4.4 Activity Recognition and Abnormal Behaviour Detection

To assess our models in activity recognition and abnormal activity detection, we propose the following steps:

1. The activity data is segmented into temporal slices by using a sliding window approach as described in [83].

2. Sensor-based features are extracted from the slices. These features are *binary*, *change-point* and *last-fired* representations as in [83].

3. RNNs (Vanilla, GRU and LSTM variants) and CNNs are trained to recognise daily activities.

4. The trained models are used to detect anomalies deviating from the normal sequences.

The order of sensor readings is important to take the spatial information into account. We use a sliding window approach to segment data into activity instances.

We applied the same sliding window approach as in [83] to extract the sensor reading chunks which are used as input to RNNs and CNNs. The window size is based on time rather than the number of sensor events (one sensor reading tuple is named as sensor event). Different time windows can be used such as 1-minute slices, 60-second slices, etc. If 1-minute windows are used, all sensor readings which are in that minute are extracted in a window. These windows are mapped into specific representations as described in the following.

4.4.1 Sensor Data Representation

Raw sensor reading are mapped into raw (binary), change-point and last-fired representations using a sliding window. These representations are depicted in Figure 4.5. Figure is retrieved from [83].

- *Binary*: This representation gives 1 when the sensor is triggered and 0 when that sensor is not triggered.
- *Change-point*: This representation gives information when a sensor changes value. More specifically, it gives 1 when a sensor changes its current state (either from state 1 to state 0 or vice versa) and 0 when its value remains the same.
- *Last-fired*: This representation indicates which sensor is fired last. The sensor that changed state last continues to give 1 and changes to 0 when another sensor changes state.

4.4.2 Abnormal Behaviour Detection

In order to recognise daily activities, training instances of the datasets and their corresponding labels are fed into the RNNs and CNNs and these models are trained. The models assign a class label to each instance with a confidence value. Firstly, the

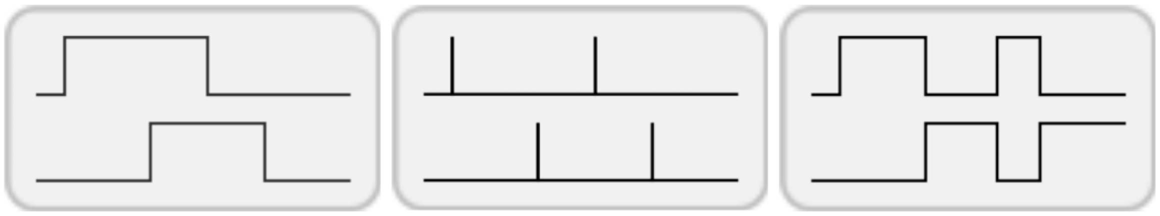

(a) Raw (b) Changepoint (c) Last-fired

Figure 4.5: Sensor representations

mean of confidence values of training instances for each class is calculated as follows.

$$m_j = 1/N \sum_{t=1}^{N} p_t \tag{4.14}$$

where m_j is the mean confidence value of class j and p_t is the confidence value for training instance t of that class and N is the total number of instances in that class. Then when a new test instance is introduced, if the model assigns it to a class with a confidence value which is bigger than the mean of that class (m_j), that instance is considered as a normal activity, otherwise it is flagged as an abnormal activity.

Now, we will continue with the experimental settings, evaluation metrics and results with RNNs and CNNs, followed by discussion.

4.5 Experiments

We used Keras deep learning library's [29] and Theano's [78] implementations of the RNNs (GRU, LSTM, Vanilla RNN) and CNNs in this study. Adam optimiser is used and the instances are fed into the system with a batch size of 20 samples. Moreover for the sake of comparison, we also used the One-class SVM from WEKA with default parameters, Naïve Bayes (NB), Hidden Markov Models (HMM), Hidden Semi-Markov Models (HSMM) and Conditional Random Fields (CRF) which are based on the implementation provided in [83].

4.5.1 Evaluation Metrics

In order to assess the recognition success, we use the following metrics: Precision, Recall, F-measure and Accuracy, where TP is true positive, TT is total true labels, TI is total of inferred labels, N is the number of instances in a specific class of the dataset and $Total$ is the total number of instances of all classes in the dataset. As seen in Formulas 4.15 and 4.16, final precision and recall values are calculated by taking average over classes. Note that precision and recall measures are used since these metrics give some idea on the models' performance on imbalanced datasets like the ones in this study. On the other hand, the accuracy represents the percentage of correctly classified time slices, therefore more frequently occurring classes have a larger weight in this measure.

$$\text{Precision} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TI_i} \tag{4.15}$$

$$\text{Recall} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TT_i} \tag{4.16}$$

$$\text{F-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.17}$$

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} TP_i}{\text{Total}} \tag{4.18}$$

Abnormal behaviour detection success rate is evaluated by sensitivity and specificity metrics. Sensitivity or True Positive Rate (TPR) refers to the method's ability to correctly detect instances which are abnormal. Specificity or True Negative Rate (TNR) gives the percentage of correctly recognized normal instances, thus reflects the method's ability to differentiate between normal and abnormal.

$$\text{Sensitivity} = \frac{TP}{(TP + FN)} \tag{4.19}$$

$$\text{Specificity} = \frac{TN}{(TN + FP)} \tag{4.20}$$

4.5.2 Results with Recurrent Neural Networks

We split Kasteren datasets into a test and training set using the leave-one-day-out cross-validation approach. One full day of sensor readings is used for testing and the remaining days are used for training. Then we cycle over all days and report the average performance. We left out 10% of the training data for validation. We also set the batch size to 10 instances and the epoch to 500 iterations. The internal architecture of RNNs (2 layers consisting of 30 and 50 nodes respectively) and time step of the sequences (25 activity slices) were empirically set.

Note that the results obtained by the models HMM, HSMM, CRF and NB (see Tables 4.1 - 4.3) are taken from the study by [83].

Table 4.1 refers to the results obtained on dataset A and shows that there is no clear winner among the three different feature representations. Considering the accuracy, the results indicate that LSTM is the best method (with the accuracy of 96.7%) when *last-fired* feature is used, while HMM performs the worst. Using *change-point* feature, HMM outperforms all other methods. Using *binary* feature on the other hand shows that CRF (accuracy of 89.8%) is the best. Also all RNNs, NB and SVM do not perform well when adopting *change-point* feature. HMM and HSMM are not good when using *binary* feature representation. In a nutshell, for the majority of the methods, except HMM and HSMM, *last-fired* representation is the best one. In terms of recall which reflects better on performance in the presence of imbalanced data, the highest value is obtained by GRU (80.6%). This potentially indicate that RNNs are good to detect relevant class instances. CRF, for instance, score higher on precision,

Table 4.1: Activity recognition results for household A of Van Kasteren dataset.

Model	Feature	Precision	Recall	F-Measure	Accuracy
NB	Binary	48.3 ± 17.7	42.6 ± 16.6	45.1 ± 16.9	77.1 ± 20.8
	Change-point	52.7 ± 17.5	43.2 ± 18.0	47.1 ± 17.2	55.9 ± 18.8
	Last-fired	67.3 ± 17.2	64.8 ± 14.6	65.8 ± 15.5	95.3 ± 2.8
HMM	Binary	37.9 ± 19.8	45.5 ± 19.5	41.0 ± 19.5	59.1 ± 28.7
	Change-point	70.3 ± 16.0	74.3 ± 13.3	72.0 ± 14.2	92.3 ± 5.8
	Last-fired	54.6 ± 17.0	69.5 ± 12.7	60.8 ± 14.9	89.5 ± 8.4
HSMM	Binary	39.5 ± 18.9	48.5 ± 19.5	43.2 ± 19.1	59.5 ± 29.0
	Change-point	70.5 ± 16.0	75.0 ± 12.1	72.4 ± 13.7	91.8 ± 5.9
	Last-fired	60.2 ± 15.4	73.8 ± 12.5	66.0 ± 13.7	91.0 ± 7.2
CRF	Binary	59.2 ± 18.3	56.1 ± 17.3	57.2 ± 17.3	89.8 ± 8.5
	Change-point	73.5 ± 16.6	68.0 ± 16.0	70.4 ± 15.9	91.4 ± 5.6
	Last-fired	66.2 ± 15.8	65.8 ± 14.0	65.9 ± 14.6	96.4 ± 2.4
Vanilla	Binary	46.5 ± 17.7	64.8 ± 16.2	53.5 ± 16.3	86.8 ± 10.6
	Change-point	46.3 ± 19.5	63.8 ± 16.4	53.2 ± 17.9	61.4 ± 16.4
	Last-fired	61.9 ± 19.1	74.3 ± 12.8	67.2 ± 16.4	95.5 ± 3.4
LSTM	Binary	50.8 ± 18.4	63.9 ± 16.5	56.2 ± 17.1	86.7 ± 10.5
	Change-point	46.8 ± 18.7	63.6 ± 14	53.5 ± 16.7	61.4 ± 16.4
	Last-fired	63.7 ± 19.9	73.9 ± 16.8	68.1 ± 18.2	96.7 ± 2.6
GRU	Binary	47.3 ± 18.7	69.1 ± 14.9	55.4 ± 16.5	86.6 ± 10.7
	Change-point	42.9 ± 19	65.0 ± 15.3	51.0 ± 17.1	61.4 ± 16.4
	Last-fired	61.8 ± 16.3	80.6 ± 11.5	69.5 ± 14.0	96.1 ± 2.5
SVM	Binary	45.6 ± 17.9	69.1 ± 15.9	54.2 ± 15.9	85.4 ± 10.4
	Change-point	40.3 ± 19.1	63.4 ± 14.6	48.6 ± 17.0	55.9 ± 18.7
	Last-fired	58.6 ± 16.2	77.2 ± 14.0	66.3 ± 14.9	96.1 ± 2.4

because the most frequent-class instances are favoured, but then it is not so good at when it comes to the infrequent classes. Overall, there is a clear hint that that recurrent architectures perform better than HMM, NB and HSMM for most of the cases, while CRF is slightly better than these recurrent architectures on dataset A.

Table 4.2: Activity recognition results for household B of Van Kasteren dataset.

Model	Feature	Precision	Recall	F-Measure	Accuracy
NB	Binary	33.6 ± 10.9	32.5 ± 8.4	32.4 ± 8.9	80.4 ± 18.9
	Change-point	40.9 ± 7.2	38.9 ± 5.7	39.5 ± 5.9	67.8 ± 18.6
	Last-fired	43.7 ± 8.7	44.6 ± 7.2	43.3 ± 4.8	86.2 ± 13.8
HMM	Binary	38.8 ± 14.7	44.7 ± 13.4	40.7 ± 12.4	63.2 ± 24.7
	Change-point	48.2 ± 17.2	63.1 ± 14.1	53.6 ± 16.5	81.0 ± 14.2
	Last-fired	38.5 ± 15.8	46.6 ± 19.5	41.8 ± 17.1	48.4 ± 26.9
HSMM	Binary	37.4 ± 16.9	44.6 ± 14.3	39.9 ± 14.3	63.8 ± 24.2
	Change-point	49.8 ± 15.8	65.2 ± 13.4	55.7 ± 14.6	82.3 ± 13.5
	Last-fired	40.8 ± 11.6	53.3 ± 10.9	45.8 ± 11.2	67.1 ± 24.8
CRF	Binary	35.7 ± 15.2	40.6 ± 12.0	37.5 ± 13.7	78.0 ± 25.9
	Change-point	48.3 ± 8.3	51.5 ± 8.5	49.7 ± 7.9	92.9 ± 6.2
	Last-fired	46.9 ± 12.5	47.8 ± 12.1	46.6 ± 12.9	89.2 ± 13.9
Vanilla	Binary	26.7 ± 13.5	46.9 ± 24.8	32.5 ± 17.9	65.2 ± 34.7
	Change-point	39.6 ± 8	62.4 ± 15.3	48.3 ± 10.2	76.9 ± 13.9
	Last-fired	41.2 ± 12.3	64.4 ± 17.8	49.7 ± 13.6	87.9 ± 13.1
LSTM	Binary	29.1 ± 12.0	44.0 ± 22.0	33.9 ± 16.2	63.5 ± 32.7
	Change-point	40.0 ± 11.2	59.0 ± 16.4	47.5 ± 12.9	76.8 ± 14.2
	Last-fired	40.8 ± 10.7	60.1 ± 16.3	48.2 ± 12.3	87.2 ± 13.2
GRU	Binary	28.5 ± 15.9	36.3 ± 17.2	31.4 ± 16.2	64.5 ± 32.1
	Change-point	37.7 ± 7.6	53.5 ± 9.2	44.9 ± 7.1	76.4 ± 14.5
	Last-fired	41.7 ± 13.2	56.9 ± 17.9	47.5 ± 14.6	87.0 ± 12.9
SVM	Binary	39.6 ± 10.9	58.5 ± 17.4	46.7 ± 12.9	81.6 ± 18.5
	Change-point	32.3 ± 6.5	53.6 ± 7.5	40.0 ± 6.2	67.9 ± 28.5
	Last-fired	36.4 ± 5.4	54.6 ± 10.4	43.5 ± 6.6	86.2 ± 14.9

Table 4.2 refers to the results obtained on dataset B and shows that SVM is the best method when adopting *binary* representation achieving the accuracy of 81.6%. On the other hand, CRF is the best when using the *change-point* feature and *last-fired* representations with accuracy 92.9% and 89.2% respectively. It can be noted that HMM is not as good as the other methods achieving the best case 81.0% with the *change-point* representation. The closest successful model to CRF is Vanilla RNN

and again overall RNNs deliver high recall rates compared to the other methods. *Change-point* and *last-fired* representations give the highest recall results except for CRF.

Table 4.3: Activity recognition results for household C of Van Kasteren dataset.

Model	Feature	Precision	Recall	F-Measure	Accuracy
NB	Binary	19.6 ± 11.4	16.8 ± 7.5	17.8 ± 9.1	46.5 ± 22.6
	Change-point	39.9 ± 6.9	30.8 ± 4.8	34.5 ± 4.6	57.6 ± 15.4
	Last-fired	40.5 ± 7.4	46.4 ± 14.8	42.3 ± 6.8	87.0 ± 12.2
HMM	Binary	15.2 ± 9.2	17.2 ± 9.3	15.7 ± 8.8	26.5 ± 22.7
	Change-point	41.4 ± 8.8	50.0 ± 11.4	44.9 ± 8.8	77.2 ± 14.6
	Last-fired	40.7 ± 9.7	53.7 ± 16.2	45.9 ± 11.2	83.9 ± 13.9
HSMM	Binary	15.6 ± 9.2	20.4 ± 10.9	17.3 ± 9.6	31.2 ± 24.6
	Change-point	43.8 ± 10.0	52.3 ± 12.8	47.4 ± 10.5	77.5 ± 15.3
	Last-fired	42.5 ± 10.8	56.0 ± 15.4	47.9 ± 11.3	84.5 ± 13.2
CRF	Binary	17.8 ± 22.1	21.8 ± 20.9	19.0 ± 21.8	46.3 ± 25.5 ‘
	Change-point	36.7 ± 18.0	39.6 ± 17.4	38.0 ± 17.6	82.2 ± 13.9
	Last-fired	37.7 ± 17.1	40.4 ± 16.0	38.9 ± 16.5	89.7 ± 8.4
Vanilla	Binary	15.4 ± 5.3	43.1 ± 18.1	22.2 ± 7.3	50.2 ± 22.4
	Change-point	31.3 ± 7.1	54.9 ± 11.3	39.5 ± 8.3	72.2 ± 13.0
	Last-fired	38.3 ± 16.3	59.6 ± 15.1	45.8 ± 14.8	86.7 ± 12.5
LSTM	Binary	16.8 ± 6.2	34.8 ± 12.5	22.1 ± 7.4	45.3 ± 21.2
	Change-point	31.0 ± 5.1	53.3 ± 6.5	38.9 ± 5.0	72.0 ± 13.0
	Last-fired	41.3 ± 17.2	57.3 ± 15.9	47.5 ± 16.1	87.4 ± 12.4
GRU	Binary	18.7 ± 8.3	33.2 ± 12.7	23.9 ± 9.6	46.7 ± 23.4
	Change-point	31.2 ± 8.3	$47. \pm 10.9$	31.2 ± 8.5	71.6 ± 12.6
	Last-fired	40.4 ± 16.5	52.7 ± 16.4	45.4 ± 16.9	86.6 ± 12.3
SVM	Binary	19.4 ± 9.0	35.2 ± 12.7	24.0 ± 9.2	37.4 ± 19.0
	Change-point	25.6 ± 6.2	51.4 ± 9.5	34.0 ± 7.2	57.8 ± 15.5
	Last-fired	37.0 ± 7.9	55.5 ± 11.6	44.1 ± 8.5	87.5 ± 12.1

Table 4.3 reports the results on dataset C showing that CRF performs best for *change-point* and *binary* representations obtaining 82.2% and 89.7% respectively. Overall, none of the methods performs well when adopting *binary* representation. The results are slightly better with *change-point* but clearly better when applying the *last-fired* representation. RNNs again give the highest recall values for all representations. Overall, the results show that RNNs perform better than HMM, NB and HSMM in all cases, while CRF is slightly better than RNNs. But in terms of recall,

RNNs outperform all methods for all feature representations. The reason behind this is that RNNs perform better for imbalanced data compared to CRF. RNNs variants generally perform equally well.

Table 4.4: Abnormal activity detection results for the synthetic dataset.

Model	Sensitivity (TPR)	Specificity (TNR)
NB	40.40%	56.50%
HMM	58.36%	3.80%
HSMM	68.85%	67.8%
CRF	66.22%	59.45%
One-class SVM	72.11%	66%
LSTM	91.43%	59.04%

For abnormal activity detection, we considered LSTM only and compared against NB, HSMM, HMM, SVM and CRF as shown in Table 3.3. We used only *last-fired* feature in this experiment. The results indicate that LSTM is the best to prune false negatives compared to the other methods. Methods like NB, One-class SVM which do not capture the data order performs the worst. The models ignore the frequency of the activity, but apply the temporal and contextual information to make a decision. Results show that LSTM is capable of encoding the order of activities. Hence, when an activity is introduced in a different context or in a different order, LSTM can detect such anomalies.

4.5.3 Results with Convolutional Neural Networks

In order to test the effect of convolutions on different dimensions and different architectures, we tested the following networks (see Figure 4.6) on the Aruba dataset. Here, the input matrix is $N \times M$, where the rows are sensor readings for each time slice and columns are the values of each sensor as time passes.

1D Convolution: In this model, convolution is done on temporal dimension. As depicted in Figure 4.6-a, in the convolutional layer, 100 filters with a length of 10 is used. 1D convolution is followed by a max-pooling layer, which has a stride of

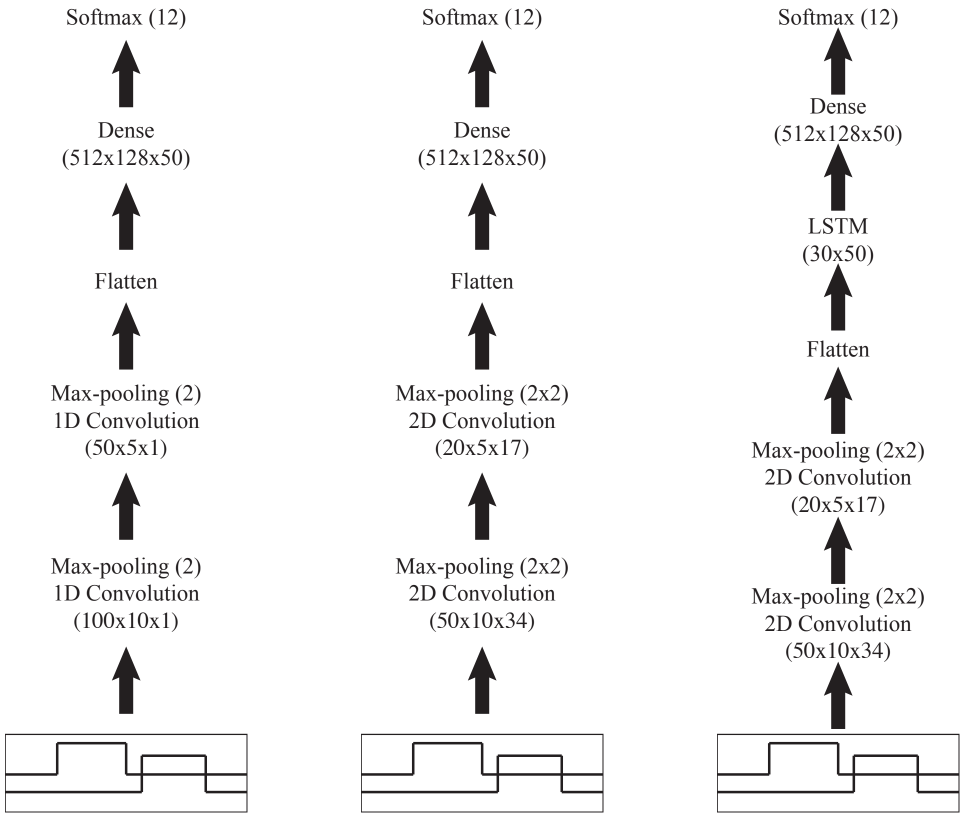

Softmax (12)
Dense (512x128x50)
Flatten
Max-pooling (2) 1D Convolution (50x5x1)
Max-pooling (2) 1D Convolution (100x10x1)
Softmax (12)
Dense (512x128x50)
Flatten
Max-pooling (2x2) 2D Convolution (20x5x17)
Max-pooling (2x2) 2D Convolution (50x10x34)
Softmax (12)
Dense (512x128x50)
LSTM (30x50)
Flatten
Max-pooling (2x2) 2D Convolution (20x5x17)
Max-pooling (2x2) 2D Convolution (50x10x34)

Figure 4.6: Convolutional Neural Network Architectures used.

2. Then another convolutional layer (with 50 filters and a length of 5) and a max-pooling layer are added. After the extracted features are flattened, these features are fed into dense layers (3 hidden layers having 512, 128 and 50 units respectively) and then the final decision is given by a softmax layer producing the confidence values of assigned class labels.

2D Convolution: In this model, convolution is done on both of the dimensions, specifically on feature and temporal dimension. 100 filters with a size of 10×34 are used in the first convolutional layer which is followed by a 2×2 max-pooling. Then another 2D convolution operator is added this time with 20 filters with the size of 5×34. The flattened features are fed into the same dense layer and the softmax layer described above.

CNN and LSTM (2D CNN + LSTM): CNNs can learn spatial relationships on a given $N \times M$ input but they cannot relate a current input to the next one in the occurrence order of the input sequence. To overcome such limitation, we use LSTMs at the end of the CNN network. In this combination, firstly, the 2-layer 2D-CNN described above is used to learn the fruitful feature representation. And then the extracted feature maps are fed into LSTM layers which will be taking further temporal information of the slices into account. LSTM has hidden layers of size 30×50 respectively. The LSTM layer is followed by a dense layer with 128 hidden units and then another dense layer with 50 units. Eventually, softmax layer classifies the input into one of the activity classes with a probability value.

In order to evaluate our methods, we first split the Aruba dataset into train and test sets. However, the split is not done with a traditional split method since dividing daily activity datasets based on a fixed time period such as day is more meaningful [76]. Aruba test bed was collected in 224 days, thus 70 days are used as test, 15 days for validation and the remaining days are used for training. The first WSU set *adlnormal*, representing normal behaviour, are used to train the classifiers, while the second set, *adlerror*, containing the abnormal activity, is used for test set.

Table 4.5: Activity recognition results for Aruba with LSTM

Feature	Precision	Recall	F-measure	Accuracy
Binary	34.21%	28.26%	30.95%	49.69%
Change-point	29.28%	26.41%	27.77%	48.79%
Last-fired	43.18%	43.37%	43.27%	87.72%
Change-point + Last-fired	37.33%	42.42%	39.71%	87.78%

The first experiment was performed on Aruba to choose the best input representation that gives the highest recognition results. This experiment is performed with the LSTM classifier on 1 minute time slices of sensor readings. Extracted slices are mapped into *binary*, *change-point* and *last-fired* representations. As seen in Table 4.5, *last-fired* representation gives the highest precision and recall rates as well as a good accuracy rate (accuracy rates of 87.72%, 49.69%, 48.79% for *last-fired*, *binary* and *change-point* respectively). A combination of the best two features: *change-point* and *last-fired*, gives a slightly better accuracy rate (87.78%) than *last-fired* alone but results in lower precision and recall values. Thus, we continue our following experiments with *last-fired* feature.

Table 4.6: Activity recognition results on the Aruba dataset.

Model	Precision	Recall	F-measure	Accuracy
NB	42.87%	61.04%	50.36%	84.37%
HMM	43.66%	72.03%	54.36%	77.90%
HSMM	43.97%	71.56%	54.47%	77.98%
CRF	50.24%	52.83%	51.50%	88.58%
LSTM	38.65%	41.29%	39.92%	89.00%
CNN-1D	31.42%	36.78%	33.89%	87.50%
CNN-2D	46.84%	41.68%	44.11%	89.67%
CNN-2D + LSTM	51.20%	50.55%	50.87%	89.72%

In Table 4.6, we provide an analysis of the second experiment which is activity recognition. The success rates by both generative and discriminative methods on Aruba set are depicted in Table 4.6. The results indicate that CNNs with 2D convolution (accuracy of 89.67%) and also CNN-2D followed by an LSTM classifier outperforms CRF (accuracy of 89.72%). The reason is CNNs extract their own fruitful features while CRF only relies on the given input. HMM and HSMM give the worst accuracy results (77.90% and 77.98% respectively). NB gives better accuracy result (84.37%) than HMM and HSMM but it results in lower precision (42.87%) and recall (61.04%) rates. Although HMM and HSMM give the best recall rates (72.03% and 71.56%), they fail in giving good precision rates (43.66% and 43.97% respectively). We see that CNN-1D network has an accuracy of 87.50% while it fails in high precision (31.42%) and recall (36.78%) values. CNN-1D extracts features on temporal dimension, it takes temporal information within a time-slice chunk into account but on the other hand it ignores the relationship between sensors since it doesn't do convolution on the feature dimension. Thus, it doesn't learn class specific feature maps to differentiate between different classes resulting in high accuracy (87.50%) but low precision and recall. When 2D convolution is used, both temporal and spatial information are taken into account and the networks learn more informative features. Thus, it gains the ability to learn class specific features, which results in higher precision and recall values (46.84% and 41.68%) and high accuracy results (89.67%). CNNs cannot remember the previous and the next inputs, but feeding the feature maps into an LSTM layer helps us process the temporal dimension further. In result, CNN-2D + LSTM networks achieves a precision rate of 51.20% and a recall rate of 50.55% and an accuracy of 89.72%.

In Figure 4.7, extracted feature maps from first and second layer and the flatten layer are visualised for CNN-2D network as described in Figure 4.6. We see that noise is reduced and more informative features are learnt as the layer level increases. The x-axis represents features while y-axis is time axis and the white pixels are activations.

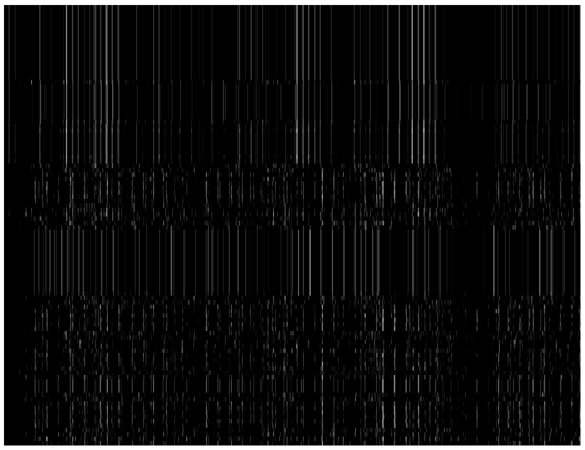
Results on the Aruba dataset show that classifiers mostly successful detect the instances of *leave home* and *enter home* activities since they are the only activities involving door sensors, thus they are not confused with any other activities. Moreover, *meal preparation* activity is confused with *wash dishes* activity most of the time since they involve same kind of sensors and they both take place in the kitchen. Also *house keeping* activity is generally confused with *work* activity since they may take place in the same room and may involve same sensors.

Table 4.7: Abnormal behaviour detection results on modified the Aruba dataset

	Aruba Modified		WSU	
Model	Sensitivity	Specificity	Sensitivity	Specificity
NB	99.33%	33.89%	46.17%	98.42%
HMM	45.54%	27.71%	100%	50.55%
HSMM	100%	35.61%	100%	42.89%
CRF	100%	66.03%	47.87%	72.17%
LSTM	98.67%	75.48%	86.50%	77.89%
CNN -2D	85.33%	33.89%	88.70%	67.46%

The third experiment, abnormal activity detection is performed firstly on modi-

(a) First layer activations.


(b) Second layer activations.

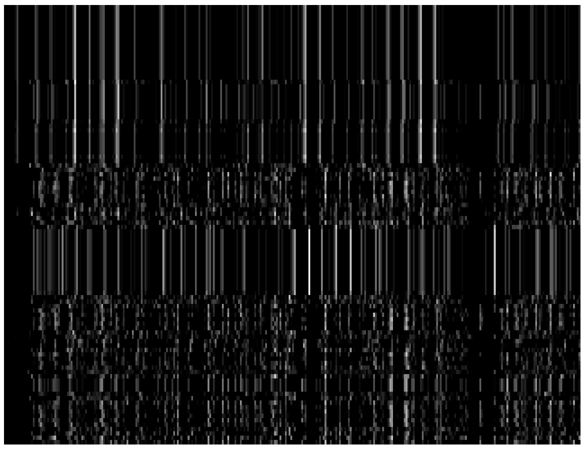

(c) Flatten layer activations.

Figure 4.7: Learned features from 2D CNN.

fied Aruba set. As a representative of CNN networks, we present the result only with the CNN-2D network in order to see the affect of CNNs individually. After training the models with normal behaviour, test set which included abnormal behaviour is introduced to the classifier and activity instances which are assigned a label with low confidence values are flagged as abnormal. In Table 4.7, we see that the highest specificity is achieved by LSTM networks giving an accuracy of 75.48% (and sensitivity rate of 98.67%). Although NB, HMM and HSMM models gives higher sensitivity rates (99.33%, 100%, 100% respectively), the specificity rates are smaller (33.89%, 27.71% and 35.61% respectively). HMM gives the worst results (a sensitivity rate of 45.54% and a specificity rate of 27.72%). CNN-2D gives a sensitivity rate of 85.33% and a specificity of 33.89%. This shows that LSTMs are more suitable to detect repetition and order related abnormal activities since it can relate current input with the upcoming ones what CNN cannot do.

The second part of anomaly detection experiments are performed on WSU testbed. We extracted 30 second time-slice chunks from sensor readings from WSU. This dataset is not collected in a daily life scenario, thus sensor readings are not in a sequential order. Thus the sensor readings are available only for activities labelled in the dataset. *adlnormal* set is used as training set and *adlerror* set is used as test dataset. The aim here is to measure how successful the classifiers are to detect the anomalies given normal dataset. The results on Table 4.7 indicate that the highest sensitivity rate is given by HMM and HSMM (both 100%), while HMM gives a specificity rate of 50.55% and HSMM achieves specificity of 42.89%. The highest sensitivity rate is achieved by CNN-2D classifier (86.70%), but LSTM gives a very close sensitivity rate (86.50%) and a higher specificity rate (77.89%) where CNN-2D achieves a specificity rate of 67.47% only. In [17], the authors present their results as follows. The number of correctly detected activities are 95 for *adlnormal* and 76 for *adlerror*, both out of 100 activity instances. We perform experiments on activity slices, on the other hand they take whole activity and extract features from that activity and then try to decide if it is normal or abnormal. The problem here that is in real life scenario, we cannot know, where an activity starts and ends. Thus using slice-based detection is more meaningful.

LSTM is better to capture repetition related activities, while CNN is better to detect confusion related sub-activities. CNN can detect changes in feature patterns. Even though it is not explicitly defined in daily activity datasets each activity is formed by steps. The steps in this dataset are based on the motion sensors triggered. For example, when we consider the *sleeping* activity, we see that the resident first goes out of bed, then goes to the middle of the room and then goes to the bathroom in the *bed to toilet activity*. CNN doesn't need to extract them, but it exploits them hierarchically automatically. In the end, model can not identify steps but it detects the anomaly. Thus, whenever the orders of sensors or these steps change, input matrix changes which leads different feature maps extracted by CNNs.

4.6 Conclusion

This chapter introduces a method of recognising sensor based activities and detecting abnormal behaviour related to cognitive decline in smart homes. CNNs and RNNs are exploited as well as their combination in order to achieve these tasks. Our results on activity recognition show that these methods are better than their competitors such as NB, SVMs, HMMs, HSMMs and CRFs. The experiments with RNNs showed that the temporal order of activities is an important cue to model daily life activity patterns of an elderly. Once temporal order is modelled, detecting any abnormal behaviour deviating from these patterns will be easier. Furthermore, the empirical experiments showed that the three variants of RNNs generally perform equally well, but LSTM seems to be slightly better across all datasets used in this study. Moreover, in terms of representation, there is no clear preference, but *last-fired* feature seems to be better, at least on the datasets A and C, compared to the *change-point* and *binary* representations. CNN experiments showed that fruitful features extracted from these sensor representations are useful to encode the relationship between single sensor features. The combination of RNNs and CNNs perform the best results since they encode both temporal and spatial information coming from the sensors. However, they fail to understand the intrinsic structure of activities. Thus, we need hierarchical models to understand the sub-activities and their relationships in a given activity and relate them to cognitive decline related abnormal behaviour.

CHAPTER V

Recursive Deep Learning for Abnormal Behaviour Detection

5.1 Introduction

Daily life activities are often composed of several steps [55]. For example; the activity *wash clothes* implies the following actions: *get clothes from basket, fill up washing machine, turn on washing machine, take clothes out.* The anomalies related to dementia may be reflected in the repetition frequency of these steps and their relation with each other. The elderly people with dementia tend to confuse things and repeat or skip some steps during the completion of a specific activity. For example, when an elderly person wants to make a phone call, he/she may check the phonebook many times and perform this step more than once. Building activities from their granular units hierarchically would be helpful to understand the internal dynamics of the activities. Hence, the problem of activity recognition can be viewed as a hierarchical and recursive learning problem which resembles scene parsing or phrase detection [75, 74]. Thus in this chapter, activity recognition problem is emulated as a hierarchical learning problem which resembles to scene parsing or phrase detection. Inspired by solutions to these problems [74, 75], we explore Recursive Auto-Encoders to model upper-level activities from their low-level sub-activity structures recursively.

In this chapter, we firstly describe sensor representations in Section 5.2 and then auto-encoder models are summarised for abnormal behaviour detection in Section 5.3. In Section 5.4, we summarise how Recursive Auto-encoders are used to detect abnormal behaviour related to dementia. In Section 5.5, we describe how transfer learning is applied with Recursive Auto-Encoders to model activities and then detect abnormal behaviour. Lastly, in Section 5.6, we present the experimental settings and results along with a discussion.

5.2 Feature Engineering

In this chapter, raw sensor readings are mapped onto two representations; namely bag-of-sensors (BOS) and raw-sensor-measurement (RSM) representations.

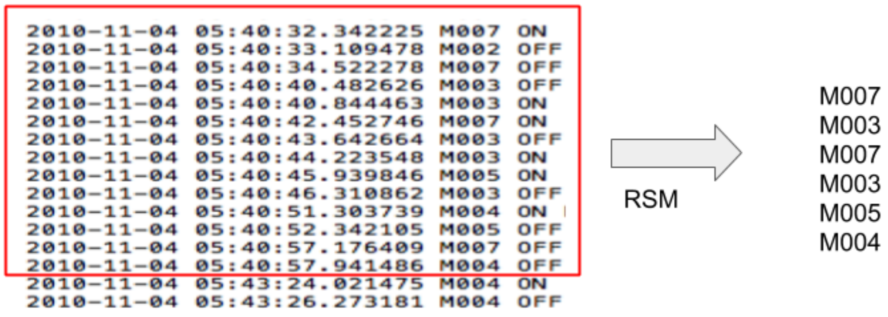
5.2.1 Bag-of-sensors Representation

This representation is the same as binary feature described in [83]. But we name it as Bag-Of-Sensors (BOS) since it resembles a bag-of-words model in document classification literature. Similar to bag-of-words model, this representation ignores the context of sensor events in a given duration. Firstly, time-slice chunks are extracted from raw sensor data via a sliding window approach [83]. A time-slice chunk can be considered as a bag that collects the sensors which are triggered in a given time. A vector of length N, where N is the total number of sensors in the dataset, is initialised to zeros and the sensors triggered at a given time, are set to 1. This representation ignores the frequency and the order of activations.

For example, sensor readings within 1 minute time are shown in Figure 5.1. There are 34 sensors in Aruba test-bed. Thus BOS representation for this chunk will be 0011101000000000000000000000000000, where only the positions at $3, 4, 5, 7$ are set to 1. Although, M_3 is triggered 2 times and M_5 is triggered only once, they have the same effect on the representation. Moreover, first M_7 is activated and then M_3 and so on, but this order is lost.

5.2.2 Raw-sensor-measurement Representation

In this version, the frequency and the correlation between the activations is preserved. For example, given the one minute data in Figure 5.1, RSM representation will be $M_7, M_3, M_7, M_3, M_5, M_4$. This representation is then mapped onto one-hot encoded representation for each sensor activation. The extracted feature will be variable length (6×34), whereas BOS has a fixed length (1×34). BOS representation ignores the relative order and the frequency of sensor activations. However, the order of sensor activations, their correlation with other sensors and their frequency are granular level important details to detect anomalies related to dementia.



```
2010-11-04 05:40:32.342225 M007 ON
2010-11-04 05:40:33.109478 M002 OFF
2010-11-04 05:40:34.522278 M007 OFF
2010-11-04 05:40:40.482626 M003 OFF
2010-11-04 05:40:40.844463 M003 ON
2010-11-04 05:40:42.452746 M007 ON
2010-11-04 05:40:43.642664 M003 OFF
2010-11-04 05:40:44.223548 M003 ON
2010-11-04 05:40:45.939846 M005 ON
2010-11-04 05:40:46.310862 M003 OFF
2010-11-04 05:40:51.303739 M004 ON
2010-11-04 05:40:52.342105 M005 OFF
2010-11-04 05:40:57.176409 M007 OFF
2010-11-04 05:40:57.941486 M004 OFF
2010-11-04 05:43:24.021475 M004 ON
2010-11-04 05:43:26.273181 M004 OFF
```

Figure 5.1: Raw sensor data and its RSM representation.

5.3 Auto-encoder Models for Abnormal Behaviour Detection

An auto-encoder network is an architecture that takes an input and is trained to reproduce that input in its prediction layer. Auto-encoders are unsupervised since they don't need explicit labels in the training phase. On the other hand, they are self-supervised because they use the input instances as labels and use training data to learn the parameters for the model. An auto-encoder has 3 parts: an encoding function, a decoding function, and a loss function. The encoder compresses the input, the decoder then reconstructs the input. Loss function calculates the error between the real input and the reconstructed input.

In a Recursive Auto-Encoder (RAE), which is originated from [66], given two children, the parent is constructed by an encoding function. Then the children are reconstructed by decoding function to calculate the loss. At each level of the tree, the same encoding and decoding function is used recursively. We will be focusing on two types of RAEs; traditional RAEs and greedy RAEs.

5.3.1 Traditional Linear Recursive Auto-Encoders

In a traditional RAE, each instance is merged with its next neighbour to construct the parent. In Figure 5.2 (figure retrieved from [75]), a list of inputs $x = (x_1, x_2, x_3, x_4)$ is given. The first parent vector y_1 is computed from the children $(c1, c2) = (x_3, x_4)$, so that $p = f(W^{(1)}[c1; c2] + b^{(1)})$ where a matrix of weights W is multiplied with the children vector. After adding a bias term, an element-wise activation function such as *tanh* is applied to the resulting vector. Then the parent y_1 is merged with the child x_2 and this goes on in the upper layers. In order to see how well this function is doing, the model reconstructs the children in a reconstruction layer: $[c_1; c_2] = g(W^{(2)}p + b^{(2)})$. During training, the goal is to minimise the reconstruction errors of the input pairs. For each pair, the distance between the original input and its reconstruction is calculated: $E = P_{orig}([c1; c2]) - P_{rec}([c1; c2])$. The process repeats until the full tree is constructed and a reconstruction error is obtained at each non-terminal node. The encoding and decoding weight parameters are learnt by using the train set and applying back-propagation algorithm.

5.3.2 Greedy Recursive Auto-Encoder

In greedy RAE, two children which give the least reconstruction error are merged at each tree level. This greedy approach is described as follows. Assume that a

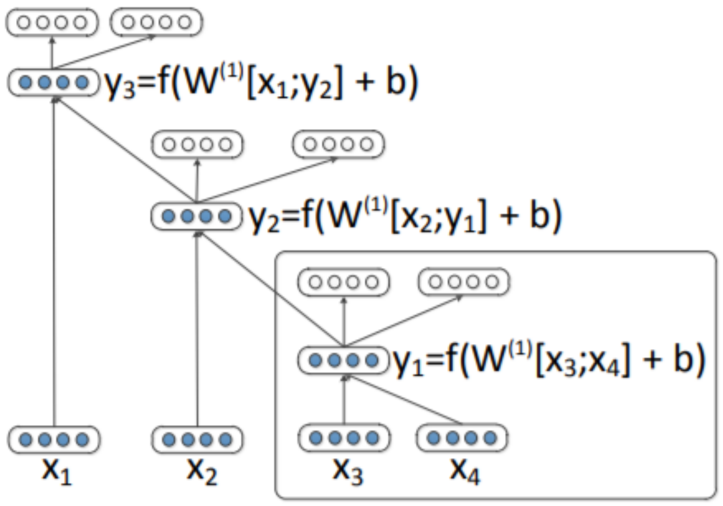


Figure 5.2: Recursive Auto-encoder structure.

sequence of instances x_1, x_2, x_3, x_4, x_5 is given (see Figure 5.3). First, the parent $p = x_1, x_2$ of children $[x_1,x_2]$ is encoded, then the children are reconstructed. The reconstruction error e_1 is calculated and kept in memory. Then, the merging is shifted to right child where the parent of children $[x_2,\ x_3]$ is encoded and the reconstruction error is calculated as e_2. This shifting is done until the last child is used. The minimum error among the errors e_1, e_2, e_3, e_4 is chosen and the corresponding children are merged at that level. Let's assume e_4 is the minimum which is a result of merging of children $[x_4,\ x_5]$. The first merging for the first level of the tree is done as $y_1 = x_4, x_5$ and these children are represented by y_1. Then the merging for the second level is done with x_1, x_2, x_3, y_1 and it continues in the same greedy manner until only one parent (y_4) remains in the last layer.

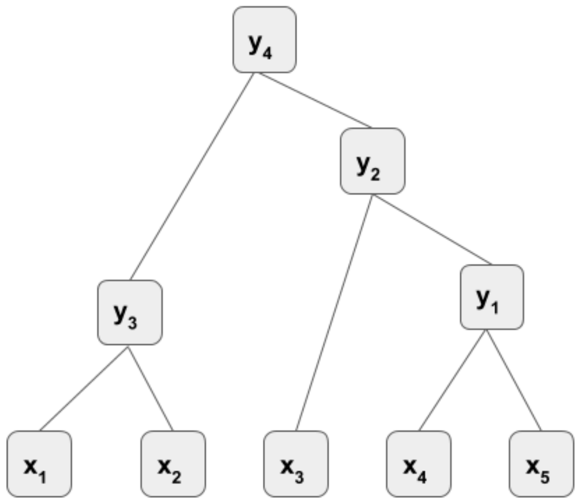


Figure 5.3: Greedy Recursive Auto-Encoder structure

5.4 Abnormal Behaviour Detection using Recursive Auto-Encoders

First, the dataset is split into training and testing sets and the training set is used to learn the parameters ($W^{(1)}$ and $b^{(1)}$ for encoding function, $W^{(2)}$ and $b^{(2)}$ for decoding function) for a RAE model. Then this RAE model is used to evaluate the test instances. Here, the idea is that given a set of training samples containing no anomalies, the goal of RAE is to design and learn a feature representation that captures normal instances. Anomalies are defined as samples that deviate from the expected behaviour. When a new activity is introduced as a test instance, if it is a normal activity, the reconstruction error will be similar to the reconstruction errors of training instances. On the other hand anomalies that represent any deviation will be poorly reconstructed and the error will be high. Reconstruction error will be exploited to decide normal and abnormal instances based on a threshold. Two different methods are used to construct RAE trees as follows.

5.4.1 BOS Representation Merging Method

A sliding window of one minute is applied on the raw data (in both training and testing dataset) and sensor readings in each one window is mapped onto BOS representation (Section 5.2). Then a window size of w is used to extract chunks from these BOS representations. Thus, these chunks have a size $w \times n$, where n is the number of features (= 34). Then each row of a chunk is merged with its next row using traditional RAE until only one parent is constructed in the end.

In Formula 5.1, the error between the original children x_1, x_2 and their reconstructed versions $x_1^{'}$, $x_2^{'}$ is calculated using the mean squared error (MSE). N is the total number of features that each x_i has. Then the error of each parent is used to decide if there is an abnormality in children or not. Here, in a constructed RAE tree for an input, time-slices in a 25 minute chunks are spanned and the relationship between each one minute slice is taken into account during the mergings in RAE.

$$E_{rec}(x_1) = 1/N \sum_{i=1}^{N} (x_{1i}^{'} - x_{1i})^2 \tag{5.1}$$

5.4.2 RSM Representation Merging Method

First, each one minute time-slice is mapped onto RSM representation. Inspired by [75], where words in a sentence are merged by a RAE, we treat each sensor activation as a word and each extracted RSM as a sentence. For example, in the extracted RSM feature $M_7, M_3, M_7, M_3, M_5, M_4$, each sensor activation such as M_7 is treated as a word. Resembling to a sentence, in a RSM representation the order of the words, their neighbours are important to decide the context of a sentence. The sensor activations in RSM representations are merged hierarchically by greedy RAE. Each sensor activation is represented as a one-hot encoding representation during the merging. Here, the error for each RSM tree is used to decide if that time-slice is abnormal or not. This error is decided in two ways. First, the average error of all parents in the tree is used. Second, the error of last parent is used. The experiments with this feature is performed in two modes, unsupervised RAE and semi-supervised RAE following the same procedure in [75].

5.4.3 Unsupervised RAE

In unsupervised RAE, activity labels are not used and RAE is trained as described in Section 5.3.1. Each sensor reading is represented by one-hot encoding and parents are constructed from the children. The error is calculated using MSE in Equation 5.1.

5.4.4 Semi-Supervised RAE

In semi-supervised RAE, the error at each parent node is a combination of unsupervised RAE error (see Section 5.3.1) and supervised error. Supervised error is calculated in the following way. Assume that we have the RSM input x_1, x_2, x_3, x_4, x_5, which is extracted within one minute duration from raw data (Figure 5.3). The activity occurred at that one minute, label l is used as the label for whole parents in the tree while the parents are used as the features. Each parent p can be seen as a feature describing the sub-tree under it. Then a softmax layer is added to each parent as follows.

$$d(p;\theta) = softmax(W^{label}p) \tag{5.2}$$

where $\theta = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}, W^{(label)})$. Assume that there are K labels, $d_k \in \mathbb{R}^K$ is a $K - dimensional$ multinomial distribution and $\sum_{k=1} d_k$. Then softmax layer's outputs are interpreted as conditional probabilities for a parent p as $d_k(p;\theta) = p(k|[c_1;c_2])$. Then, cross-entropy (supervised) error is:

$$E_{sup}(p,t;\theta) = -\sum_{k=1}^{K} t_k \log d_k(p;\theta) \tag{5.3}$$

where t_k is the k^{th} element of the multinomial target label distribution t for parent.

Then a weighted average of supervised error $E_{sup}(x_1, x_2)$ and unsupervised error $E_{unsup}(x_1, x_2)$ (Equation 5.1) is used to calculate the final error (Equation 5.4), where α is decided experimentally as a value between 0 and 1.

$$E_{rec}(x_1, x_2) = \alpha E_{unsup}(x_1, x_2) + (1 - \alpha) E_{sup}(x_1, x_2) \tag{5.4}$$

5.5 Abnormal Behaviour Detection using Transfer Learning

Machine learning based cognitive status assessment studies rely on activity recognition techniques. These methods first learn what is normal from training data and then flag the abnormal activity based on classification confidence values [60, 34, 40, 28, 53, 81, 8]. They require training data to be manually annotated, which is extremely hard and time-consuming to do. Moreover, these techniques assume that the training data is available prior to the training phase. However, we cannot expect elderly people to annotate the necessary training data. Thus, tackling the activity recognition and abnormal behaviour detection as an unsupervised process would be helpful. This monitoring may need to be done over long periods of time, maybe months, and sometimes even years. But collecting sequential data of months or years with time dependency is highly time-consuming and difficult. Thus, in this research, we use existing data from a source household to learn what is normal, and then transfer this knowledge to a target house.

The hierarchical representation of RAE provides an abstraction of activities in a house and then mapping these abstract levels to another house via transfer learning will be more useful. The activities are characterised by their hierarchical organisation, conditioned on global and local structural context. Thus we need a more abstract encoding of the granular level details of each activity. Also, abstraction provides a generalisation of the hierarchical level of information between the houses and it reduces the differences between them, making transfer learning an appealing approach. We use unlabelled data collected for normal activities from a source house to train the RAE model. Then, we transfer this model to a target house to detect abnormal behaviours related to dementia.

The proposed work consists of the following steps: 1) Time-slice chunks are extracted from sequential sensor reading data using a sliding window. 2) *Last-fired* features are extracted from time-slices as in [83]. 3) RAE is trained on a source household dataset to learn the parameters for *normal* behaviours. 4) These parameters are then transferred into a target household to detect abnormal behaviours.

We chose households A and C of Kasteren datasets [42] since they span more days (25 days and 18 days respectively). The activities performed in household A and C are used to reflect normal behaviours. However, some of the data in household C is modified (Section III) to generate samples representing abnormal behaviour of dementia sufferers. Here, household A will be used as the target house while household C will be used as the source house.

5.5.1 Feature Extraction

After the synthesis, the datasets are processed in the following way. Firstly, 1 minute slices are extracted from datasets using a sliding window [83]. This time-slice length is long enough to provide a discriminative sensor pattern and short enough to provide high resolution labelling results. After discretization we have a total of 35486 time-slices for dataset A and 26236 time-slices for dataset C.

Then time-slices are mapped into *last-fired* feature representation [83]. *Last-fired* feature [42] indicates which sensor fired last. The sensor that changed state last continues to give 1 and changes to 0 when another sensor changes state.

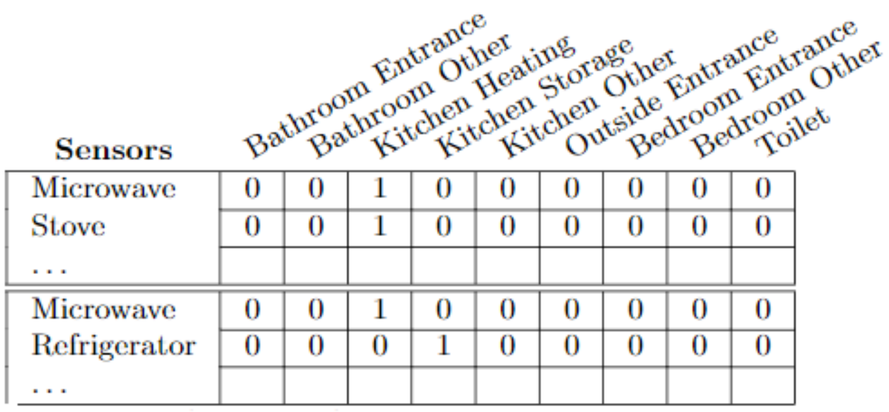
The last representation gives an indication of the location of an inhabitant. The sensors replaced in the house will not be triggered as long as people do not move.

As they start moving, the sensors are triggered based on the location of the movement, which will provide an update of their current location [83]. The updates, in the form of a time-series data, provide fine-grained information about the activity performed. For example, when the person performs *meal preparation* activity, sensors are triggered one after another based on their location. The steps taken between sensor activations form granular level details of the activity performed. We make the key observation that such patterns are hierarchical and they follow grouping rules at multiple levels of abstraction. Findings in [93] support our approach. The authors extract location-based patterns that occurs during daily-life routines. Hence, we employ RAE as a hierarchical model to organise the steps in an activity and record their relative ordering. We exploit *last-fired* feature to model location based granular level information in the activities performed, since such feature allows for capturing execution details of the activities.

Since Kasteren dataset does not include any fine-grained level activities labelled, we treat each time-slice of a sequence as a fine-grained unit that forms the activities. Assume that the person performs *having dinner* activity D in n time-slices such as $D = t_1, \ldots, t_n$, where each time-slice t_i represents a step. Relating these time-slices with each other in a hierarchical way and taking their spatial information into account is important to capture dementia related abnormal behaviours.

5.5.2 Sensor Mapping

There are 14 sensors in dataset A and 21 sensors in dataset C. We map these sensors to each other by using meta features (see Figure 5.4) as described in [42] . For example, a sensor on the microwave might have one meta feature describing the sensor is located in the kitchen, and another that the sensor is attached to a heating device. We use the mapping that combined sensor readings in a single feature based on their function (e.g. sensors used during cooking). In [84], different mapping strategies, such as union, intersection and duplicate are investigated. We use *union* mapping since it gave the best results. Using *union* mapping for each function group, the union of all the sensors in the group is taken, resulting in one sensor output per group per house. For example, the front and back door in the target house are combined into a single sensor and matched with the front door sensor in the source house. This results in 7 sensor groups, which will be treated as features. Moreover, the activities in two datasets are mapped and 9 similar activities are used [42].



Sensors	Bathroom Entrance	Bathroom Other	Kitchen Heating	Kitchen Storage	Kitchen Other	Outside Entrance	Bedroom Entrance	Bedroom Other	Toilet
Microwave	0	0	1	0	0	0	0	0	0
Stove	0	0	1	0	0	0	0	0	0
...									
Microwave	0	0	1	0	0	0	0	0	0
Refrigerator	0	0	0	1	0	0	0	0	0
...									

Figure 5.4: Sensor mapping on Van Kasteren dataset.

5.5.3 Abnormal Behaviour Detection

First, house A normal dataset is used to learn the parameters ($W^{(1)}$, $b^{(1)}$) for encoding function and ($W^{(2)}$, $b^{(2)}$) for decoding function) of RAE. These parameters are then used to construct RAE trees to test instances of house C. In each level of the tree, two children are merged to form a feature vector as their parent, which encodes the information coming from the children. Thus, the feature vector at the root node summarises all the information coming from the children in the tree and their hierarchical orderings are learnt by RAE. The feature vectors at parent nodes can be decomposed into their granular level, hierarchical pieces by using the decoding weights. Anomalies are defined as samples that are deviations from the expected behaviour. Any deviations from this normal can be identified by measuring the reconstruction error E described above. When a new activity is introduced as a test instance, if it is a normal activity, the reconstruction error will be smaller since anomalies that represent any deviations will be poorly reconstructed. If it is an abnormal behaviour, which is not seen in the training data (source household), it will reconstruct that instance with a higher error. As mentioned earlier, at each layer of RAE, a reconstruction error is calculated. We exploit these errors to decide if that activity is normal or abnormal based on a threshold.

The comparison is done with the same supervised methods described in Chapter II. These models first perform activity recognition and then detect anomalies. First, using the training instances and their corresponding labels in dataset A, these models are trained. Then the instances in dataset C are given to the trained classifiers. The models assign a class label to each instance with a confidence value. When a new test instance in house C is introduced, if the model assigns it to a class with a confidence value which is bigger than a threshold, that instance is considered as a normal activity, otherwise it is flagged as an abnormal activity.

5.6 Experiments

5.6.1 Evaluation Metrics

In order to assess the abnormal behaviour detection success, True Positive Rate (TPR) and False Positive Rate (FPR) is used. These values for different thresholds are depicted on a receiver operating characteristic (ROC) curve. Moreover, Area Under Curve (AUC) is calculated for each model to interpret the results in a better way. True Positive Rate (TPR) refers to the method's ability to correctly detect instances which are abnormal. FPR gives the percentage of mislabelled normal instances, thus reflects the method's ability to differentiate between normal and abnormal. Precision, recall, accuracy and F-measures are used to evaluate classifier performance as described in Chapter IV.

5.6.2 Experiments with Recursive Auto-encoders

In this section, we will present the experimental settings and results obtained with Recursive Auto-Encoders.

5.6.2.1 Experimental Setting

In order to evaluate the proposed RAE based method, the dataset is split into training and testing sets, where 70 days are used as test, 15 days for validation and the remaining 139 days are used for training. The test set is modified to include sub-activity and activity related anomalies. The modifications (Section III) are done separately, which result in two different testing sets. We analyse sub-activity and activity related anomalies separately to see the affect of RAEs on both types of anomalies individually.

RAEs are compared with the following state-of-the-art supervised methods; RNNs (Long Short Term variants), CNNs, NB, HMM and CRF. For comparison experiments, BOS representation is used since we need a fixed-length feature representation for these experiments. These models are supervised and they assign a class label to each instance with a confidence value. When a new test instance is introduced, if the model assigns it to a class with a confidence value which is larger than a threshold, that instance is considered as a normal activity. Otherwise it is flagged up as an abnormal activity. Keras Deep Learning library's [29] and Theano's [78] implementations of the CNNs and LSTM are used in this study. Results with NB, HMM, HSMM and CRF are based on the implementation provided in [83]. In the CNN and RNN experiments, Adam optimiser [44] is used and the instances are fed into the system with a batch size of 20. In CNN experiments, time-series window of length 10 seconds is extracted from the raw sensor reading data based on a sliding window approach. The CNN model has the following layers: A 2D convolutional layer (with 20 kernels of size 5×10), a Max Pooling layer (with a pooling size of 2×2), a 2D convolutional layer (with 10 kernels of size 10×15), a Max Pooling layer (with a pooling size of 2×2) a flatten layer, and two dense layers of size 128 and 50, followed by a softmax layer to do the classification. In LSTM two hidden layers of 50 and 100 nodes are used. Then, dense layers of size 100, 128 and 50 are added to the network, followed by a softmax layer. There are drop-out layers with a probability of 0.5 between each two layers in both CNN and LSTM models.

RAE experiments are performed in two ways. First experiment is conducted with BOS representation and it is implemented on Theano and Python, while the second experiment is based on Socher et. al.'s Matlab implementation [75] and performed with RSM representation.

5.6.2.2 Evaluation of Features and Models

The results for anomaly detection are shown in Figures 5.5 and Figure 5.6. The abbreviations in Table 5.1 are used for the results.

Table 5.1: Abbreviations used on ROC curves.

LSTM	Long Short Term Memory variants of RNNs
HMM	Hidden Markov Model
CRF	Conditional Random Field
CNN	Convolutional Neural Network
BOS-L	Unsupervised (U) traditional linear (L) RAE with BOS representation
RSM-GUA	Unsupervised (U) RAE with RSM and greedy (G) merging when the average (A) of all parent errors is used
RSM-GSA	Supervised (S) RAE with RSM feature and greedy merging when the average (A) of all parent errors is used
RSM-GUE	Unsupervised (U) RAE with RSM and greedy (G) merging when the error of root node (E) is used
RSM-GSE	Supervised (S) RAE with RSM and greedy (G) merging when the error of root (E) node is used

Figures 5.5 and 5.6 show the results on activity related anomaly detection. The results show that LSTM is the best method giving the highest AUC ($58, 48\%$), while NB is the worst one (with AUC 41.48%). Activity related anomalies occur at the order of the activities involved and LSTM is good at capturing temporal dependency between inputs, so it detects changes in the order of the activities. CNN comes as the second method (with AUC 57.79%). Instead of relying on given features, CNN extracts its own features taking spatial context into account. After CNN, RSM-GSE produces AUC of 55.83%, which is followed by RSM-GSA (with AUC 54.59%) and RSM-GUE (with AUC 54.10%). Then CRF achieves AUC of 53.80%. The next methods are RSM-GUA (with AUC 52.10%), BOS-L (46.45%) and HMM (AUC of 43.55%).

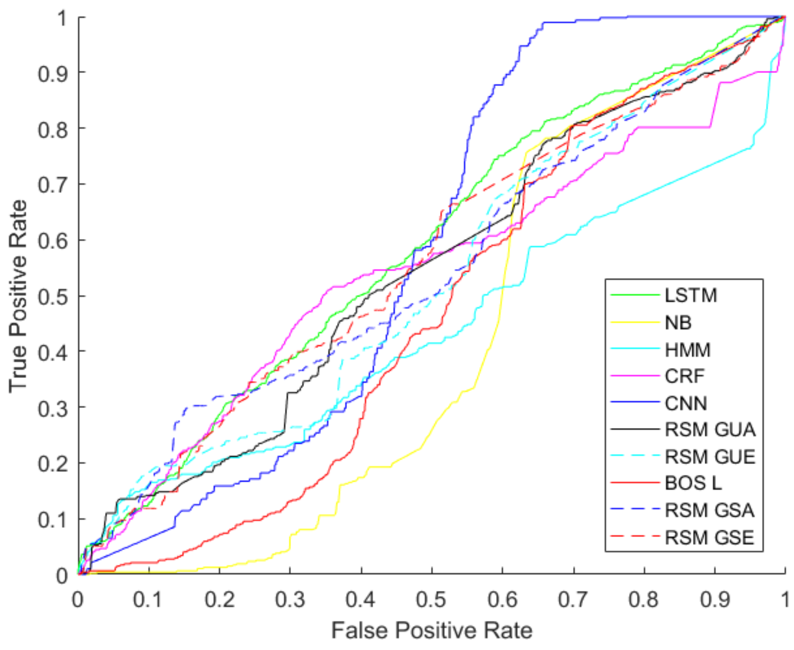
We see that RSM-GUE performs better than supervised methods NB, CRF and HMM. The reasons for this as follows. HMMs are constrained to binary transition and emission feature functions, which force each instance to depend only on the current label and each label to depend only on the previous label. NB does not rely on any temporal dependency and it uses BOS representation, neglecting both temporal context and granular-level details of each feature. Linear-chain CRF has limited memory since it captures linear dependency between the current input and the previous one. RAEs learn hierarchical structures and the learned structures can capture more of the semantic relationships of sensor activations in RSM representation.

Moreover, supervised models, especially deep learning methods such as CNNs and RNNs, require too much training data. Collecting and labelling that much training data is time consuming and a laborious task. Moreover, providing labelled data just once wouldn't be enough since observation of dementia sufferers in a smart home is a task which can be up to years. Thus, a continuous labelling of the data would be necessary. Also, these models need activity classes to be fixed. On the other hand, in a time lapse of years, users may change their behavioural patterns and they may introduce new activity labels. This would require the train set to be updated and labelled again.

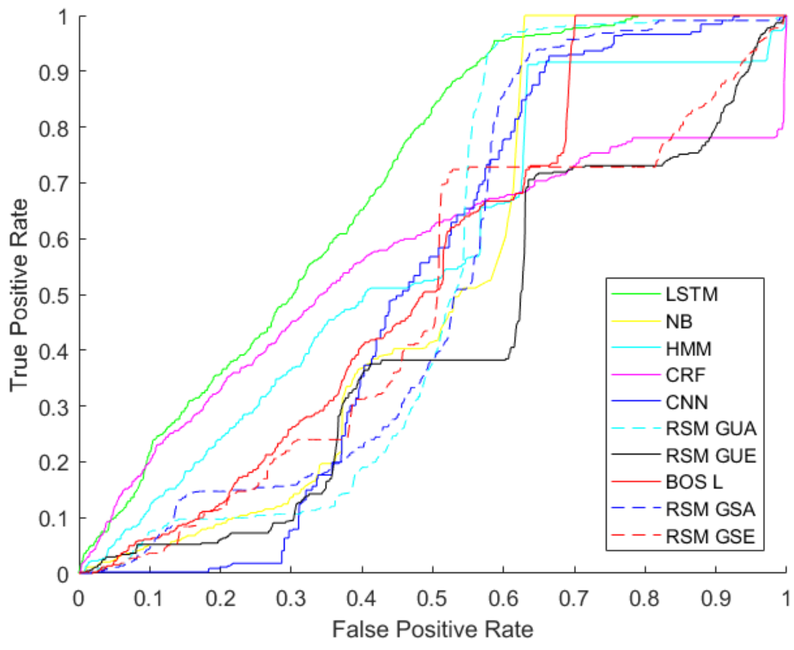
In Figures 5.5 and 5.6, LSTM is the best method to detect anomalies related to sub-activities with an AUC of 69.91%. The second best achieving method is HMM and then CRF with AUC of 56.76% and 55.43%, which is followed by BOS-L achieves AUC of 54.36%. Please remember that BOS-L merges 25 instances of 1 minute time-slices at each tree. Thus it can detect changes within 25 minutes and relate the changes between these time-slice instances. Sub-activity related anomaly causes changes in the feature vector itself and in the neighbour feature vectors. NB gives AUC of 51.20% but it cannot capture temporal context. Greedy RAE with RSM model only takes 1 minute time-slice into account and constructs RAE trees, but it unfortunately cannot relate each RAE tree of 1 minute time-slice to next time-slice since it cannot take temporal information into account.

We see that RAE models don't give the best results when AUC values are compared. However, when an optimum threshold is chosen on the ROC curve, RAE models can perform as well as supervised methods. For example, in Figure 5.5-a, RSM-GSE gives the same TPR (65%) and FPR (55%) with CNN and LSTM at intersection point of their ROC curves. In Figure 5.5-b, we see that RSM-GUA intersects with LSTM at TPR of 95% and FPR of 55%. AUC weights TRP and FPR equally. However, in some scenarios like ours, detection of true positives is more important. For abnormality detection problem in skewed datasets, where the number of anomalies is much less than normal ones, true positives are more important.

Moreover, although supervised methods give better AUC results than RAE models, they require labelling information which is tedious and time consuming task to obtain. In a case where getting a training set is difficult, RAE models can be an alternative to supervised methods. Moreover, detection of dementia indicators is a process spanning months and maybe years. In this time, the habits of residents may



(a) ROC curve for activity related abnormal behaviour test set.



(b) ROC curve for sub-activity related abnormal behaviour test set.

Figure 5.5: ROC curves for abnormal behaviour detection using RAE.

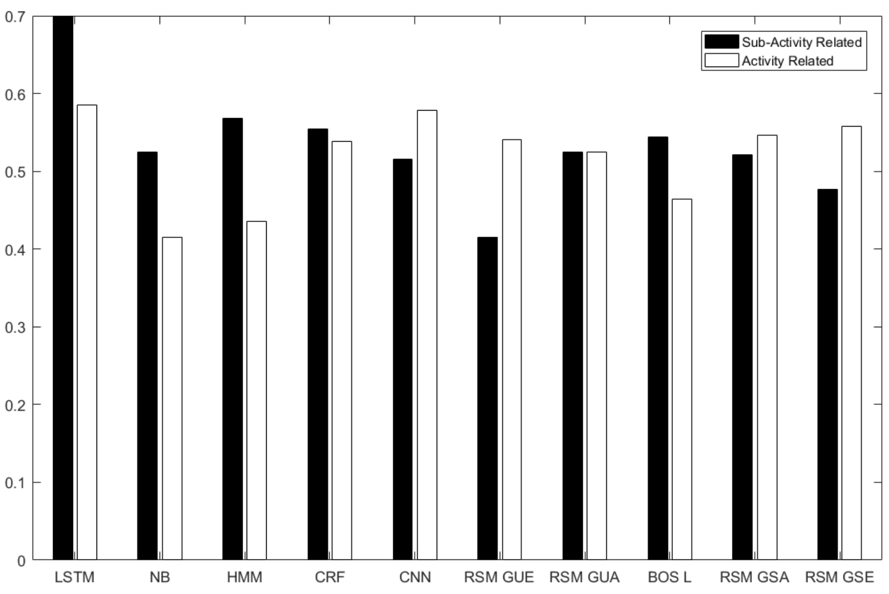


Figure 5.6: AUC histogram for ROC curves using RAE.

change and new activities may emerge. Thus, obtaining training data and labelling it wouldn't be sufficient since this labelling process would be needed to repeated again when the activity labels change. But with unsupervised methods such as RAE, no activity label data is used and the model can be updated at any time. Some of the supervised methods such as CRF, take frequency information of each class instances into account and favours those classes in terms of classification. This would be a problem with imbalanced datasets like daily life activity datasets, where abnormal detection of infrequent classes are important as well. However, RAE models don't learn class based parameters since they don't use class labels. We see that supervised methods used in the experiments tend to detect abnormal instances of frequent classes better than the others.

However, RAE models cannot relate one instance to another and neglect temporal information. Another problem with BOS is that it doesn't reflect the real status of an activity being performed. For example, people don't tend to close the room doors after they enter or leave. Once the door is open, the door sensor continues to emit 1. But, RMS representation only takes the activation of the sensor into account and then neglects the information that the door is left open. For scenarios, where the door sensor is not important, it is good that the ON status is not carried forward, but for abnormality detection scenarios like leaving the door open, RMS representation wouldn't be able to catch this information.

5.6.2.3 Classification Performance

The next set of experiments are conducted to evaluate the modelling ability of RAE and the representation ability of the reconstructed features. Even though RSM representation has a variable length for each input, RAE model outputs a fixed feature vector at the root node. The reconstructed feature of the root (size of 1×34) can be used as a final feature representation for the variable length input and supervised classification methods can be trained with these features for further applications. We choose J48 decision tree (the Weka implementation of the standard C4.5 algorithm) as our classifier due to its simplicity. The classification results are depicted in Table 5.2. Firstly, classifier accuracy rates with BOS representation are presented to provide a baseline for comparison. The classifier accuracy with BOS representation for activity related test set is 81.37%, while it is 81.49% for sub-activity related test set. The recognition accuracy rates with RSM representation are as follows: 78.78% for activity related anomaly test set, and accuracy of 78.49% for sub-activity related anomaly test set when supervised RAE is used, and accuracy of 71.81% for activity related anomaly test set, accuracy of 72.64% for sub-activity related anomaly test set when unsupervised RAE is used.

Although RSM representation gives less classification accuracy compared to BOS representation, it gives better precision and recall rates, which means experiments with RSM is good at providing class specific detailed information and it results in higher precision and recall rates. For example, BOS-Original experiment achieves precision of 42.92%, recall of 42.31% and F-Measure of 41.84% on activity anomaly set,

Table 5.2: Classification performance using reconstructed features.

Model	**Activity Anomaly Test Set**			
	Precision	**Recall**	**F-Measure**	**Accuracy**
BOS - Original	42.92%	42.31%	41.84%	81.37%
RSM - Semi-supervised	46.92%	47.23%	46.06%	78.78%
RSM - Unsupervised	47.33%	38.72%	37.89%	71.81%
Model	**Sub-activity Anomaly Test Set**			
	Precision	**Recall**	**F-Measure**	**Accuracy**
BOS - Original	40.93%	42.08%	38.93%	81.49%
RSM - Semi-supervised	43.77%	42.77%	41.65%	78.49%
RSM - Unsupervised	42.84%	39.27%	37.89%	72.64%

while RSM semi-supervised experiment achieves precision of 46.92%, recall of 47.23% and F-Measure of 46.06%. For imbalanced datasets, RSM representation can be used where not only total accuracy is important, but also precision and recall on the least frequent classes are important. We see that supervised RAE calculates better representation than unsupervised RAE and it gives very close classification accuracy rate with BOS representation, which shows that RSM has a high representation ability.

5.6.2.4 Pattern Extraction

We also provide a quantitative analysis to show that how greedy RAE merges sub-activities in a hierarchical way to model activities. Sub-activities come together and form meaningful structures, which we call patterns. A sample set of constructed trees are shown in Figure 5.7. For example, we see that the sensors M_{19} and M_{15} are grouped in the constructed trees for *meal preparation* activity. In Aruba testbed, these sensors are replaced close to each other and when the resident performs *meal preparation* activity, these sensors are triggered one after another. Thus, they form a sub-activity pattern during the performance of this activity. The pattern constructed by these two sensors are identified as *near the kitchen range and sink* in [93], which supports our finding. Also we see that the sensor M_{16} is added to this sub-activity (M_{15}, M_{19}) which probably represent the cupboard usage during *meal preparation*

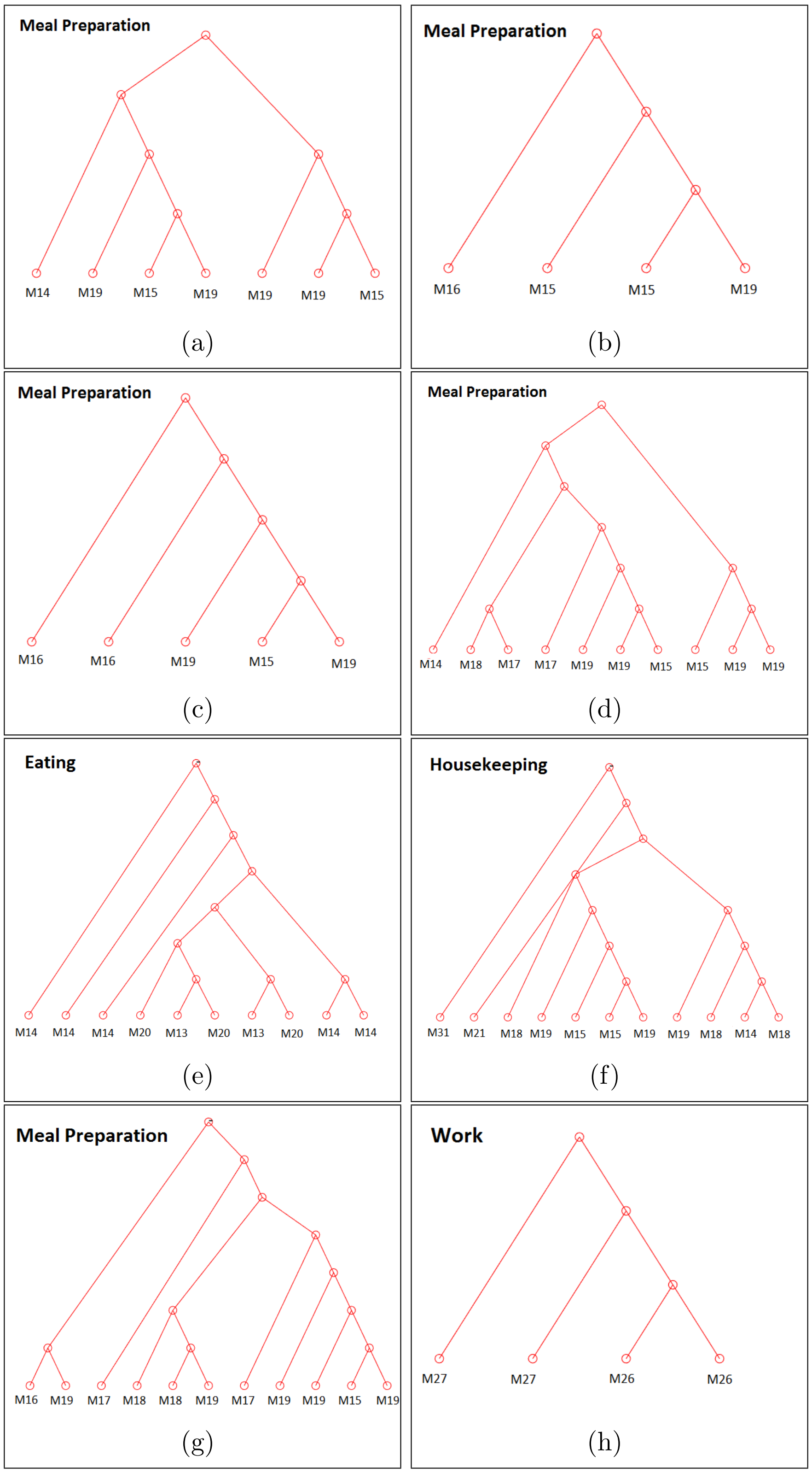

Meal Preparation
M14 M19 M15 M19 M19 M19 M15
(a)
Meal Preparation
M16 M15 M15 M19
(b)
Meal Preparation
M16 M16 M19 M15 M19
(c)
Meal Preparation
M14 M18 M17 M17 M19 M19 M15 M15 M19 M19
(d)
Eating
M14 M14 M14 M20 M13 M20 M13 M20 M14 M14
(e)
Housekeeping
M31 M21 M18 M19 M15 M15 M19 M19 M18 M14 M18
(f)
Meal Preparation
M16 M19 M17 M18 M18 M19 M17 M19 M19 M15 M19
(g)
Work
M27 M27 M26 M26
(h)

Figure 5.7: Constructed RAE trees for the training set.

activity. Another grouping of sensors, namely M_{17}, M_{15}, M_{19} shows another sub-pattern in this activity, which is again (M_{17} and M_{19}) found as a movement pattern in [93]. We see that RAE hierarchically models these relations in the trees. In *eating* activity, we see that M_{13} and M_{15} represent a sub-activity and M_{14}, M_{13}, M_{15} represent another sub-activity which is constructed by the sub-activity M_{13}, M_{15} and the sensor M_{14}.

Table 5.3: N-gram patterns extracted from the training set.

Activity	**2-gram**	**3-gram**
Bed to Toilet	M_4, M_7 M_5, M_7	M_4, M_5, M_7 M_4, M_4, M_7
Meal Preparation	M_{15}, M_{19} M_{18}, M_{19} M_{17}, M_{19}	M_{15}, M_{19}, M_{19} M_{15}, M_{18}, M_{19} M_{15}, M_{16}, M_{19}
Relax	M_6, M_9 M_8, M_9 M_9, M_{13}	M_9, M_9, M_{13} M_9, M_9, M_{10} M_9, M_{13}, M_{20}
Eating	M_8, M_{14} M_6, M_{14}	M_9, M_{14}, M_{14} M_{10}, M_{14}, M_{14}
Work	M_{26}, M_{27} M_8, M_{26}	M_{26}, M_{26}, M_{27} M_{26}, M_{27}, M_{27}
Sleeping	M_2, M_3 M_3, M_3 M_3, M_7	M_2, M_3, M_7 M_2, M_3, M_3 M_3, M_3, M_7
Wash Dishes	M_{15}, M_{19} M_{18}, M_{19} M_{17}, M_{19}	M_{15}, M_{16}, M_{19} M_{15}, M_{19}, M_{19} M_{15}, M_{18}, M_{19}
Housekeeping	M_{14}, M_{20} M_{13}, M_{20}	M_{15}, M_{18}, M_{19} M_{14}, M_{18}, M_{20}
Leave Home	M_{31}, D_3 M_{18}, M_{21}	M_{29}, M_{30}, D_4 M_{10}, M_{22}, m_{29}
Enter Home	M_{31}, D_3 M_{21}, M_{14}	M_{29}, M_{30}, D_4 M_{22}, M_{30}, D_4
Respirate	M_{27}, M_{25}	M_{25}, M_{25}, M_{25}

Moreover, we extract the most common and important patterns for each activity class in the following way. The idea is that the sensor readings which are triggered one after each other frequently, represent a sub-activity (pattern). If they are seen together frequently in the train set, RAE learns to reconstruct them better and then in the test set, it gives less reconstruction error compared to the ones not seen frequently. We firstly sort all reconstruction errors of each node in train set, and take top-500 nodes with the least error. Then n-gram patterns are calculated with these top patterns. We calculate n-grams with only $n = 2$ and $n = 3$, which is already enough to see the patterns in the dataset. The n-grams are extracted from constructed RAE trees by supervised greedy method on the train set. The most frequent 2-grams and 3-grams are shown for each activity class in Table 5.3. For example, for the activity *sleeping*, the most frequent pattern is M_2, M_3, this makes sense because when we look at the sensor locations on Aruba testbed, we see that these sensors are on the bed and they will be triggered one after another during *sleeping* activity, thus they have a correlation. After extracting these frequent patterns (sub-activities), we can look for their errors in the RAE trees. If there is high error at those patterns, we can easily detect specific anomalies related to these patterns. For example, to check if the person is washing the dishes after cooking activity, we can check sub-activity between the sink and the kitchen table and check the error of this sub-activity.

5.6.3 Experiments with Transfer Learning

In this section, we will present the experimental settings and results obtained with Recursive Auto-Encoders when transfer learning is used.

5.6.3.1 Experimental Setting

In RAE trees, each child represents 1 minute time-slice. Thus, each child is a feature vector of size 1×7, where 7 is the number of features. RAE trees are constructed with a time-step of 5, where 5 (time-slice) instances are merged in a RAE tree. This time-step parameter is chosen experimentally.

To run experiments on LSTM, we used drop-out with a value of 0.5. We also set the batch size to 10 instances and the epoch to 500 iterations. The internal architecture of LSTM (2 hidden layers consisting of 30 and 50 nodes respectively) and time-step of the sequences (25 activity slices) were empirically set.

In supervised models, the parameters are learnt by using the instances of dataset A, which is treated as a train set. And then the parameters are transferred into dataset C, whereas the instances of this dataset is treated as a test set. The activities in house C are evaluated based on the model learnt from the house A.

5.6.3.2 Classification Performance

The first experiment is conducted to evaluate the classification performance of the supervised methods when transfer learning is used. These methods are trained on house A and then tested on house C. Activity recognition accuracy rates are depicted in Table 5.4. These results are very close to activity recognition rates with leave-one-out cross validation presented on the same datasets in [83], where one day of the dataset C is used as testing set, while the remaining days are used training set. However, our results are obtained via transfer learning, where the training is done on dataset A and the testing is done dataset C. In [83], the leave-one-out classification accuracy with NB, HMM and CRF are given as 87.0%, 83.9% and 89.7% as respectively. In our case, the classification accuracy rates are 87.47%, 47.88% and 84.55% with NB, HMM and CRF respectively, while it is 87.02% with LSTM. The similar results show that applying transfer learning is successful to recognise activities in dataset C. Unfortunately, we cannot test the classification accuracy of RAE model since it is an unsupervised method.

Moreover, the results in Table 5.4 show that the highest accuracy is achieved by NB, the reason is that NB favours the most frequent class. Analysing precision, recall and F-measures (36.71%, 33.37% and 34.96% respectively), we see that class-based success is not high since these measures are averaged over different classes. Accuracy is calculated for all testing instances. However, for the methods which take temporal information into consideration, such as LSTM, HMM and CRF, precision, recall and F-measure are relatively better. The highest precision is achieved by LSTM (48.58%) while the highest recall is achieved by HMM (44.18%). The highest F-measure is achieved by LSTM with a success rate of 42.95%. HMM and CRF takes temporal information into account but their capability to encode temporal information is not as good as LSTM. Now, the aim is to use these well-trained classifiers to detect abnormal behaviours in dataset C.

Table 5.4: Activity recognition results with transfer learning.

Model	Precision	Recall	F-Measure	Accuracy
NB	36.71%	33.37%	34.96%	87.47%
HMM	37.32%	44.18%	40.46%	84.88%
CRF	42.80%	37.81%	40.15%	84.55%
LSTM	48.58%	38.49%	42.95%	87.02%

The ability of RAE to reconstruct the features is evaluated by k-means clustering. After clustering the reconstructed features into 9 clusters, the dimensions of the features are reduced to 2D by Principal Component Analysis (PCA). As depicted in Figure 5.8, RAE is successful to reconstruct and differentiate the features from different classes. In this figure, each colour indicates a different class, while X and Y coordinates are 2D features (2 principal components).

5.6.3.3 Abnormal Behaviour Detection

The purpose of second experiment is to compare the methods in terms of abnormal behaviour detection. The results are depicted as ROC in Figure 5.9 and AUC calculations in Figure 5.10. The results show that the proposed unsupervised RAE based abnormal behaviour detection is competitive with the supervised methods. There is not a significant difference in the success of the supervised methods. The proposed RAE based method produces slightly worse TPR and FPR results. However

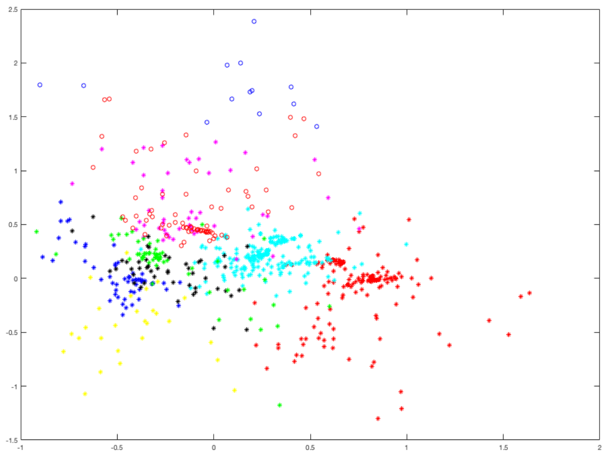


Figure 5.8: Clustering of RAE re-constructed features.

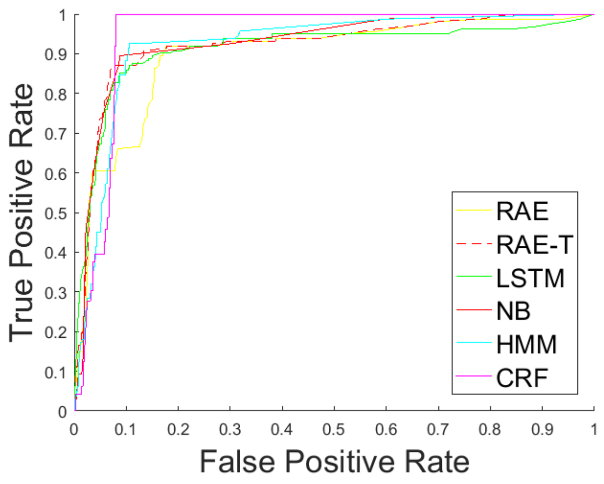


Figure 5.9: ROC curve for abnormal behaviour detection using transfer learning.

its superiority comes from the fact that it doesn't use any labels during the parameter learning process. All methods are good at detecting the abnormal behaviour instances and pruning the false alarms.

Moreover, some of the data of target household is used to train the learnt RAE model in the source household. In this way, we can tailor the RAE model for the resident by re-tuning the parameters of previously trained RAE using user-specific training examples. This strategy is similar to inductive transfer learning or self-taught learning [61] when none or few data labels are available in the target domain. However, in our case, which is unsupervised, we use only some of the unlabelled data coming

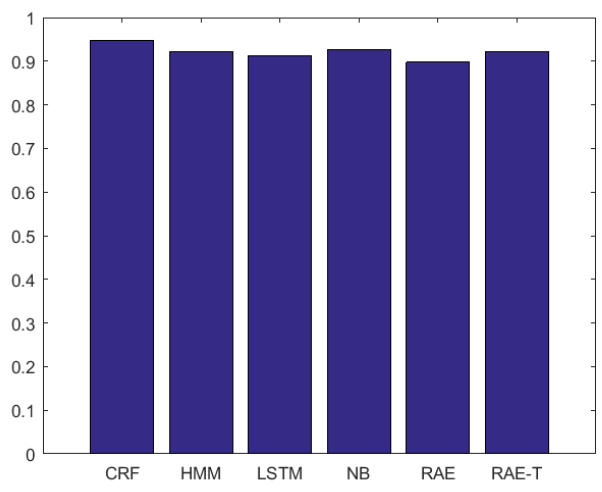


Figure 5.10: AUC bar for abnormal behaviour detection using transfer learning.

from the target dataset. This domain adaptation will be helpful to consider house-specific behaviour in the target household. Although we need some data from the target house, this still allows us to reduce several weeks or months of data collection and annotation in the target space to only a few days. For this purpose, RAE learnt on instances of source household dataset is re-trained over 10-days data from house C. The results are shown in the ROC curve (Figure 5.9 with the abbreviation RAE-T). These results indicate that re-tuning the parameters and considering the house specific behaviour improve the results.

Moreover, we calculate Cohen's Kappa statistics to show the robustness of RAE to detect abnormal behaviours in the target household. Kappa statistics is a measure that handles both multi-class and imbalanced class problems. It tells how good the classifier is performing over the performance of a classifier that simply guesses at random according to the frequency of each class. It is thought to be a more robust measure than simple percent agreement calculation, since Kappa takes into account the possibility of the agreement occurring by chance [47]. However, we prefer to use weighted Kappa statistics, since detecting abnormal behaviour is more important than pruning normal ones in our case. In health-care problems, missing a true positive may cause more serious problems than retrieving a high number of false positives. Thus, we assign a higher weight to true positive than false positive in the weight matrix of Kappa. The calculated Kappa statistics for RAE is 0.531, which is a moderate agreement according to [47].

Supervised models, especially deep learning methods such as LSTMs, require too much training data. Collecting and labelling that much training data is time consuming and a laborious task. Moreover, providing labelled data just once would not be enough since observation of dementia sufferers in a smart home is a task which can be up to years. Also, these models need activity classes to be fixed. On the other hand, in a time lapse of years, users may change their behavioural patterns and they may introduce new activity labels. Thus, labels of the train set would needed to be updated. On the other hand, when transfer learning is used, just changing the mapping of sensors and activities would be enough to adapt the model to the new data.

Moreover, supervised methods such as NB, HMM, CRF, LSTM don't encode time-slices in a hierarchical representation. RAE takes hierarchical representation of time-slices into account where it merges them in a bottom-up tree structure. The use of hierarchical models might be a better fit for transfer learning because the different levels of the hierarchy allow a better abstraction between houses.

Although we re-train the learnt RAE model on partial data steming from the source house, when there is no source data available prior, domain adaptation would not be possible. Then, there will be a problem to detect resident specific abnormal behaviours. Even though for each house, the same activity labels are used, there may still be differences in how activities are performed. Transfer learning generalises the behaviours of inhabitants between different houses and does not take resident specific behaviours into account. For example, going to toilet during sleep might be normal for a person, while it is abnormal for another person. If the person in a source house does not go to toilet during sleeping activity, then going to toilet during sleep will be detected as abnormal in the target household. Such differences in behaviour might not be transferable across different houses. A prior distribution is learnt from the source house and used to provide a sensible initial value for the model parameters of the target house. The activities in a target house can be used to detect the abnormal behaviours of a resident in a source house under the condition that the resident profiles such as age, gender and lifestyle are similar in both households. Moreover, the behaviour across different houses is transferable if the different sensors and activities in these houses can be mapped into each other. If this mapping cannot be done, then the knowledge will not be transferable. However, please note that our aim in this study is not to replace medical doctors or caregivers in the process of cognitive status assessment. The proposed method can be used as a decision supporting system rather than a decision making system.

The proposed system would improve life experience of dementia sufferers in the following way. The system detects possible candidates for abnormal behaviour to inform the caregiver or the medical doctor. The decision maker will analyse the abnormal behaviours detected by the proposed system to decide by considering the person's profile and personal life style. Detecting high amount of false positives will not introduce any risk related to the health of the person. Detecting true positives in an early stage would trigger further analysis and would be helpful for an early treatment. The important advantage of the proposed system would be to provide a cognitive status assessment in the natural flow of daily living without annoying elderly people.

5.7 Conclusion

This chapter introduces a RAE based method to detect early indicators of dementia before it gets worse. The abnormal behaviour of dementia sufferers are detected by exploiting the granular level sub-activities in activity instances. The proposed method builds activities based on their sub-activities in a recursive and hierarchical tree structure. The results show that this method is promising to model activities from their sub-activities and detect anomalies. However, this method cannot relate one instance to another and neglects temporal information.

Moreover, this chapter proposes a transfer learning and RAE-based method to detect abnormal behaviour of elderly people with dementia. Transfer learning can be an interesting option to cope with scarcity of data. The empirical results show that the proposed method is promising when supervised methods cannot be exploited because of the lack of (labelled) training data. However, the proposed method fails to detect the person's specific abnormal behaviour such as sleep patterns.

CHAPTER VI

Hierarchical Activity Recognition and Abnormal Behaviour Detection

6.1 Introduction

Convolutional Neural Networks (CNNs) have the capability to learn their own features directly from the raw data. CNNs are currently the state-of-the-art method for many problems in the literature. However, irregular data such as graphs cannot be handled by CNNs since CNNs require a fixed dimension of input. Recently, there has been a growing interest in Graph Convolutional Networks (GCNs) to apply the same convolution idea on graph-structured data. The convolution is done on the spatial neighbourhood of a graph network. Inspired by the solutions offered by GCNs, in this research, we use the GCN model to recognise activities and detect abnormal behaviour related to dementia.

In this chapter, we firstly describe Graph Convolutional Networks (GCNs) in Section 6.2, then we describe how we adapt GCNs to detect abnormal behaviour in Section 6.4 along with experiments in Section 6.5.

6.2 Graph Convolutional Networks

In this research, we use the GCN model proposed by [45], which makes use of a convolutional architecture via a localised first-order approximation in Fourier-domain to obtain an efficient linear-time graph-CNNs of spectral graph convolutions. In GCNs, the hidden layers serve as a kernel. They encode graph structure and features coming from nodes and their neighbours to extract fruitful features. In a graph-based neural network model $f(X, A)$, $f(.)$ is a neural network-like differentiable function, X is a matrix of feature vectors and A is an adjacency matrix and L is one-hot encoding labels of instances and $A \in \mathbb{R}^{NxN}$ where N is the number of nodes in the graph, while $X \in \mathbb{R}^{NxF}$ where F is the number of features. Then, the following layer-wise propagation rule is applied where a degree matrix $D_{ii} = \sum_j A_{ij}$ is calculated beforehand.

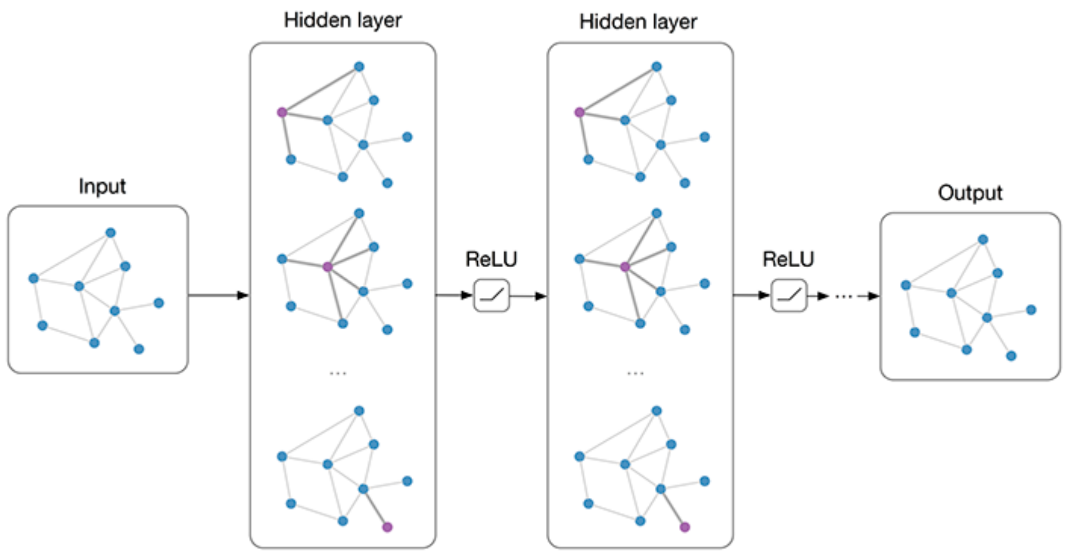


Figure 6.1: Convolutions are applied on graph network.

Let $\tilde{A} = A + I_N$ be the adjacency matrix of the undirected graph $\mathcal{G}$ with added self-connections, I_N is the identity matrix, $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ and $W^{(l)}$ is a layer-specific trainable weight matrix. $\sigma(.)$ denotes an activation function, e.g., Rectified Liner Unit (ReLu), where $ReLU(x) = max(0, x)$. Hidden layer $H^{(l)} \in \mathbb{R}^{N \times D}$ is the matrix

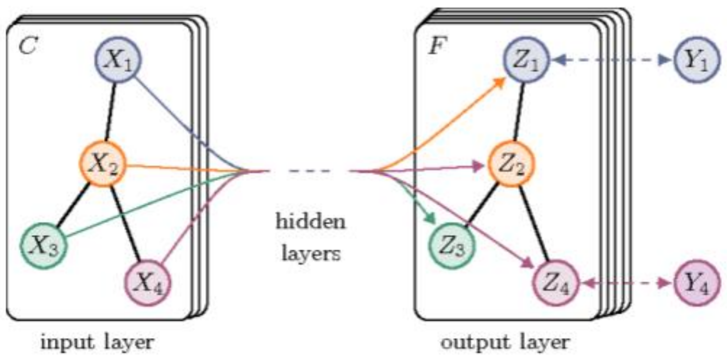


Figure 6.2: Feature maps are extracted to encode node neighbourhood.

of activations of the l^{th} layer. $H^{(0)} = X$, since the first layer is the input layer. Then $H^{(l+1)}$ is given as in Equation 6.1 (see Figures 6.1 and 6.2 retrieved from [45]).

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}) \tag{6.1}$$

In Equation 6.2, the forward model for a two-layer GCN is considered. Here, $W^{(0)} \in \mathbb{R}^{C\times H}$ is an input-to-hidden weight matrix for a hidden layer with H feature maps (hidden neurons) and C input channels. $W^{(1)} \in \mathbb{R}^{HxF}$ is a hidden-to-output weight matrix. After calculating $\hat{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$, then a forward propagation is calculated as in Equation 6.2. The softmax activation function, defined in Equation 6.3, is applied row-wise.

$$Z = f(X, A) = softmax(\hat{A}\ ReLu(\hat{A}XW^{(0)})\ W^{(1)}) \tag{6.2}$$

$$softmax(x_i) = 1/Z\ exp(x_i) \tag{6.3a}$$

$$Z = \sum_i exp(x_i) \tag{6.3b}$$

The cross-entropy error is calculated over all labelled examples as in Equation 6.4, where Y_L is the set of node indices that have labels.

$$E = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \, lnZ_{\,lf} \tag{6.4}$$

The neural network weights $W^{(0)}$ and $W^{(1)}$ are trained using the gradient descent method. In this work, batch gradient descent using full training set for every training iteration is applied.

6.3 Sensor Representation

In this chapter, each sensor activation coming from raw data will be represented as a node in the graph. Firstly, a graph is constructed as in Figure 6.3. There are two node types in our graph, namely *inner* nodes and *outer* nodes. Each *inner* node in the graph represents a sensor activation from input data. These nodes have one-hot encoding of their sensor activations. For example, let's consider the one-minute piece of raw sensor data in Table 3.2. It is mapped into raw sensor data of M_4, M_7, M_5, M_3, if we only consider ON status of each activation. Then there will be 4 inner nodes in this graph, where each inner node represents a sensor activation. For example, the first inner node M_4 will have one-hot encoding feature of 1×34, where the index at 4 is 1 while others are 0. These inner nodes will have dummy labels as they don't have any activity labels on their own. Considering that there are 11 activities in our dataset and 1 activity for *nil* activity, labels are represented by a vector of size 1×12. For example, the nodes having label 2 will have label vector 010000000000. For inner nodes, all values of label vector will be zero, meaning they don't have any labels. Inner nodes are connected to their subsequent ones as shown in Figure 6.3.

Outer nodes are the nodes to represent time-slice activity labels of sensor activities. *Inner* nodes which fall within a certain t-minutes time-slice (e.g. 1 minute) are merged to their *outer* node. Thus, there is only one *outer* node for each t-minutes sen-

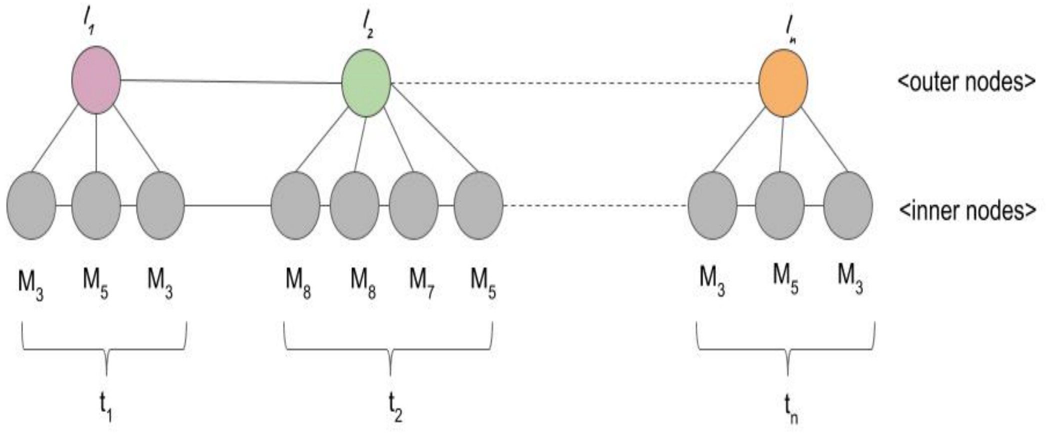


Figure 6.3: Graph structure constructed from sensor activations.

sor activations. Here, please note that the number of sensor activations (the number of *inner* nodes) is arbitrary. An *outer* node will have its corresponding time-slice's activity label. However, an *outer* node has an empty feature vector of the same size $1 \times N$. *Outer* nodes are connected to their subsequent ones and to their corresponding *inner* nodes. If whole data is split into n time-slices, then there are n *outer* nodes in the constructed graph. The data (*outer* nodes) is split into 3 subsets: training, testing and validation. We group *inner* nodes within one-minute chunks which is chosen experimentally.

6.4 Activity Recognition and Abnormal Behaviour Detection

To recognise activities for *outer* nodes, firstly, a graph is constructed as shown in Figure 6.3. The sensor activations are represented as *inner* nodes, while the labels are represented as *outer* nodes (Section 6.3). The adjacency matrix A and feature matrix F are constructed based on the graph structure to be used to train the GCN on *normal* activities. To detect abnormal activities, we use the confidence probability values of assigned labels and then by using a threshold value, we decide if they are abnormal or not (see Algorithm 3).

```
Input: Outer nodes in test set $< n_1, n_2, \ldots, n_j >$
       GCN Classifier $G$
       Threshold $th$
foreach node $n_i$ do
    Classifiy $n_i$ with $G$;
    Obtain classifier confidence probability $p_i$;
    if $p_i \geq th$ then
        $n_i$ is normal;
    else
        $n_i$ is abnormal;
    end
end
```

Algorithm 3: Abnormal behaviour detection

6.5 Experiments

6.5.1 Experimental Set-up

In order to evaluate the proposed GCN model, the Aruba dataset (Section 3.2) is split into training and testing sets. The training dataset consists of first 139 days. Next 15 days are used for validation and remaining 70 days are used as testing set.

Moreover, GCNs are compared with the following state-of-the-art supervised methods: LSTMs (Long Short Term variants of RNNs), Naïve Bayes (NB) classifier, Hidden Markov Models (HMMs) and Conditional Random Fields (CRFs). In these classifiers, BOS representation is adopted since these methods require fixed-length input vectors ($1 \times N$), for N being the number of sensors in the dataset. Each sensor activated in 1 minute is represented as 1, while others are given 0.

In the LSTM and CNN related experiments, Adam optimiser [44] is used. The models and the parameters are decided experimentally as follows. In the LSTM related experiments, the instances are fed into the system with a batch size of 20. In LSTM, two hidden layers of 50 and 100 nodes are used. Then, dense layers of size 100, 128 and 50 are added to the network, followed by a softmax layer. There are drop-out layers with a probability of 0.5 between each two layers in LSTM models. As for CNN, time-series window of length 10 seconds is extracted from the raw sensor readings. The CNN model has the following layers: A 2D convolutional layer (with 20 kernels of size 5×10), a Max Pooling layer (with a pooling size of 2×2), a 2D convolutional layer (with 10 kernels of size 10×15), a Max Pooling layer (with a pooling size of 2×2), a flatten layer, and two dense layers of size 128 and 50, followed by a softmax layer to do the classification.

In activity recognition experiment, nodes are represented by two different representations. In the first one, we only consider the ON status of activations, while in the second one we also consider the OFF status. The feature vector for ON representation is size of 1×34, where 34 is the total number of sensors in the dataset. The OFF status is represented by adding another 34 values to the feature vector, which results in a feature vector of size 1×68. For example, if the sensor M_3 has status ON, the one-hot encoding feature vector of 1×34 will have 1 at position 3, while if the same sensor has status OFF, then feature vector of 1×68 will have 1 at position 37 ($= 34 + 3$). Moreover, activations of GCN hidden neurons are fed into an LSTM network to carry temporal information further with the ability of LSTMs. Here, the same LSTM network is used as described above. Thus activity recognition experiments with GCN have 3 variants: 1) when only ON status of sensor activations are used, 2) when both ON and OFF are used and 3) when activations of kernels (hidden layers) of GCN are fed into an LSTM network.

GCN experiments are performed based on Kipf et. al.'s Python implementation [45] with raw data. Learning rate of 0.01, drop-out value of 0.5, weight decay of 0.00005 and 64 hidden neurons are used. The training is done in 100 epochs with an early stopping of 10. The experiments are conducted in two main parts: 1) Activity recognition and 2) Abnormal behaviour detection.

6.5.2 Activity Recognition Results

In these experiments, we only used test set with *activity* related abnormal behaviours since our focus is abnormal behaviour detection rather than activity recognition. Moreover, activity recognition results are very similar for both test sets.

Table 6.1 shows the activity recognition results. In particular, the best accuracy is retrieved by LSTM (85.95%), while GCN-LSTM comes after with a slight difference (85.67%) when ON and OFF activations are used. However, GCN model using ON and OFF without LSTM gives higher precision (52.25%) and recall values (50.86%). In terms of F-measure, LSTM-BOS achieves 43.29%, while GCN-LSTM achieves 43.11% and GCN with ON and OFF achieves 51.55%. This shows that GCN with ON and OFF status is good at differentiating classes. This comes from the ability of GCN to model activity slices by taking sensor activation relationships into account. Moreover, LSTM experiment is performed with BOS representation which ignores the relationship between sensor activations. Thus, BOS representation might affect the performance of LSTM as well. On the other hand, feeding GCN activations to LSTM cause a decrease in precision and recall (42.61% and 43.62%). The reason for this might be LSTM learns the temporal information of the most frequent classes and gives more importance to them.

Although GCN with ON and OFF achieves slightly better accuracy (84.06%) than its ON version (84.01%), it ends up with better F-measure (51.55% compared to 48.16%). Since F-measure is averaged on each class, this means that adding OFF

Table 6.1: Activity recognition results (%) for *activity* related abnormal behaviour.

Model	**Precision**	**Recall**	**F-Measure**	**Accuracy**
GCN (ON)	46.65	49.77	48.16	84.01
GCN (ON+OFF)	**52.25**	50.86	51.55	84.06
GCN-LSTM (ON+OFF)	42.61	43.62	43.11	85.67
NB-BOS	44.45	69.09	54.10	74.59
HMM-BOS	40.21	**77.47**	52.94	72.97
CRF-BOS	44.46	47.42	45.90	80.39
CNN-BOS	38.81	43.18	40.88	80.68
LSTM-BOS	41.93	44.73	43.29	**85.95**

status helps to differentiate the classes better. When OFF status is used, the model can understand the ordering of the sensor activations better. This ordering provides an update for the location of the person since sensors are activated one after another based on the location of sensors replaced. This gives more insight to the activity being performed. For example, results with only ON status show that *leave home* activity is confused with *enter home* activity, while *eating* activity is confused with *meal preparation* since same sensor types are involved in these activities. However, adding OFF status improves the accuracy for these activities since it reduces the confusion. No method can identify *Wash dishes* well and most of the test instances are recognised as the *Meal preparation* activity. This due to the fact that the two activities run in the same place (kitchen) and use very similar objects.

Despite the high ability of CNNs to capture spatial context, CNN model achieves relatively less accuracy (80.68%) because of the BOS representation. Moreover, CRF model outperforms NB and HMM in terms of accuracy (80.39%, 74.59% and 72.97% respectively) but NB and HMM perform better in terms of recall (69.09% and 77.47% respectively). The reason behind is that CRF favours the most frequent class in the dataset.

6.5.2.1 Sensor Pattern Extraction

A quantitative analysis is provided for each activity class to show which sensors are involved the most during the classification decision of GCN. For this purpose, the average of all GCN hidden layer activations are calculated for each feature of every activity. The normalised sensor activations are visualised in Figure 6.4. The lighter cell means that that activity nodes get more excited by that sensor. The following abbreviations are used for the activities. *N: Nil, S: Sleeping, M: Meal preparation, E: Eating, R: Respirate, W: Working, B: Bed to toilet.* For example the *outer* nodes of *meal preparation* activity correlate more with sensor M_{19}, while nodes of *eat* activity are affected by M_{14}, and *respirate* nodes are affected by M_{25}. *Wash dishes* nodes are affected by M_{15}, M_{18}, M_{19}, *bed to toilet* nodes by sensors M_3, M_4, M_5, M_6 and *leave home* and *enter home* by M_{30} and D_4 while *work* nodes gets excited by M_{26} and M_{27}. These are the sensors which are involved at most in these activities when we look at the sensor lay-out and raw sensor measurements.

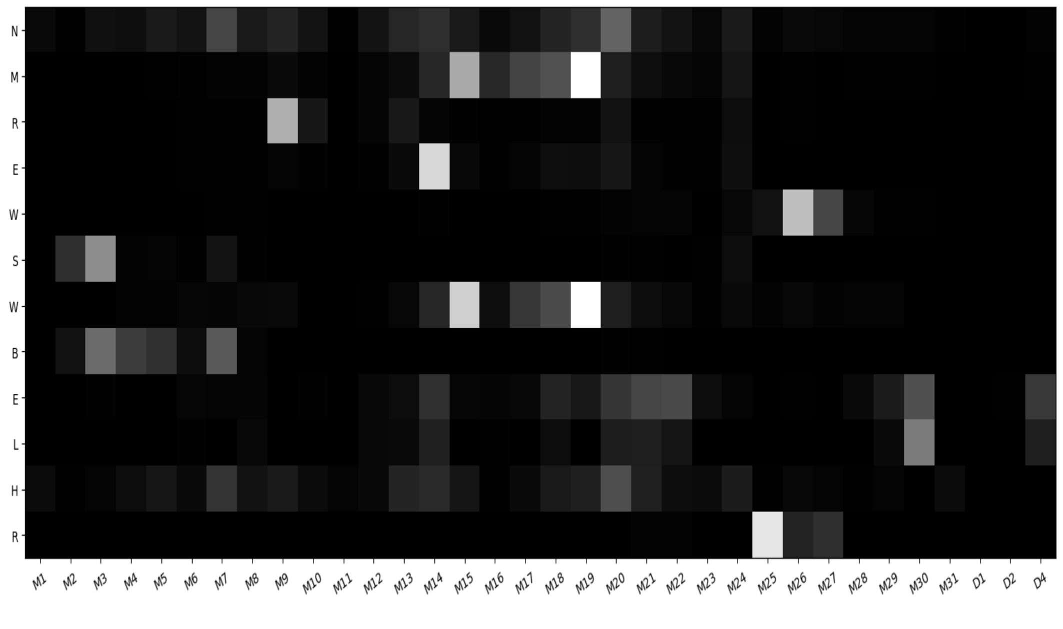


Figure 6.4: Sensor activation map for each activity.

We also present a set of example sub-graphs from each activity category in Fig-

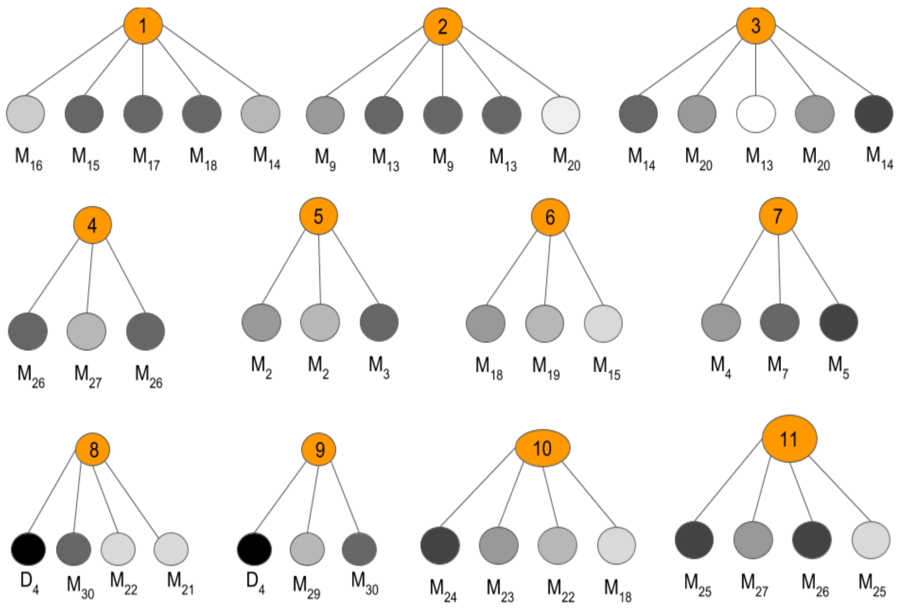


Figure 6.5: Sub-graph from each class showing the level of sensor activations.

ure 6.5. The darker colour means more excitement. The numbers on *outer* nodes show the class label of that time-slice (*outer* node). The classes are numbered in the following order: *Meal Preparation, Relax, Eat, Work, Sleep, Wash Dishes, Bed to toilet, Enter Home, Leave Home, Housekeeping, Respirate.* For example, *meal preparation outer* node is affected by sensors M_{15}, M_{17}, M_{18}. Moreover, we extract the most common and important patterns for each activity class in the graph. This is similar to frequent sub-graph mining, which is about discovering interesting patterns in graphs. In our case, sensor readings which are triggered frequently in an activity represent a pattern. If activation scores of a group of sensors are high, then it means an *outer* node gets excited by that combination of sensors. For this purpose, all n-grams are calculated for *inner* nodes and their average sensor activation scores are calculated. Then all activations are sorted, and $top-500$ nodes, which is decided empirically, are taken for each activity class to calculate n-gram patterns. We calculate n-grams with only $n = 2$ and $n = 3$, which is enough to see the patterns in the dataset. A sample set of extracted patterns are shown in Table 6.2. For example, the sensors M_{19} and M_{15} are grouped in *meal preparation* activity. In the sensor layout, these sensors are

Table 6.2: N-gram patterns learned by GCN.

Activity	**2-gram**	**3-gram**
Bed to Toilet	M_3, M_3 M_4, M_7 M_7, M_5	M_4, M_7, M_5 M_7, M_5, M_3
Meal Preparation	M_{19}, M_{15} M_{18}, M_{19} M_{19}, M_{17}	M_{15}, M_{19}, M_{19} M_{18}, M_{19}, M_{15} M_{17}, M_{19}, M_{15}
Relax	M_9, M_{20} M_9, M_9 M_9, M_{13}	M_9, M_9, M_{13} M_{10}, M_{10}, M_{10} M_9, M_{13}, M_{20}
Eating	M_{14}, M_{18} M_{14}, M_{14} M_{14}, M_{20}	M_{14}, M_{14}, M_{20} M_{24}, M_{14}, M_{14} M_{14}, M_{14}, M_{18}
Work	M_{26}, M_{26} M_{26}, M_{27} M_{22}, M_{28}	M_{26}, M_{27}, M_{26} M_{27}, M_{27}, M_{26}
Sleeping	M_3, M_2 M_3, M_3 M_2, M_3	M_3, M_2, M_3 M_7, M_3, M_7
Wash Dishes	M_{19}, M_{15} M_{18}, M_{19} M_{19}, M_{17}	M_{19}, M_{15}, M_{19} M_{15}, M_{15}, M_{15} M_{18}, M_{19}, M_{15}
Housekeeping	M_{20}, M_{20} M_7, M_5 M_{20}, M_8	M_{24}, M_{24}, M_{24} M_7, M_7, M_7
Leave Home	D_{24}, M_{30} M_{30}, M_{30} M_{22}, M_{30}	D_4, M_{30}, M_{30} M_{21}, M_{22}, M_{30}
Enter Home	D_4, M_{30} M_{22}, M_{21} M_{30}, M_{22}	D_4, M_{30}, M_{29} M_{30}, M_{22}, M_{21}
Respirate	M_{27}, M_{25} M_{25}, M_{26}	M_{25}, M_{25}, M_{26} M_{25}, M_{25}, M_{25}

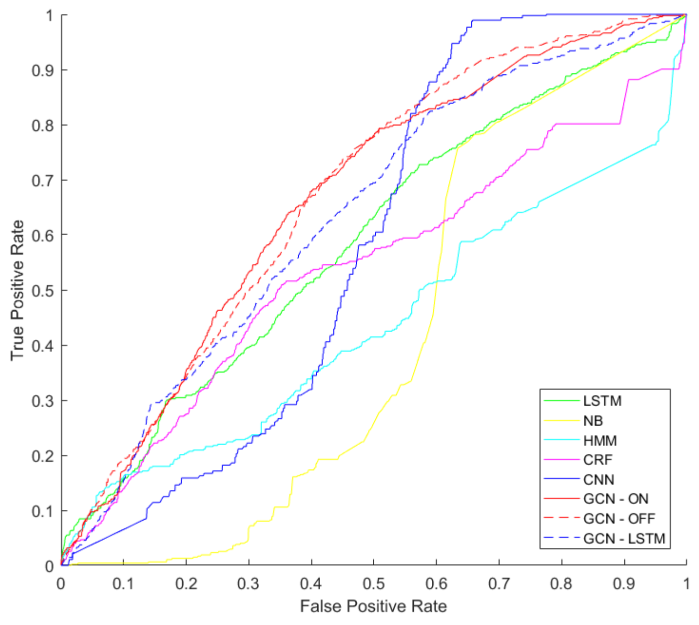
placed close to each other and when the resident performs *meal preparation* activity, these sensors are triggered one after another. The pattern constructed by these two sensors are identified in [93] as *near the kitchen range and sink.* Another grouping of sensors, namely M_{18}, M_{19}, M_{15} shows another pattern in this activity, which is again (M_{18} and M_{19}) found as a movement pattern in [93]. In *eating* activity, we see that M_{14} and M_{18} represent a pattern and M_{14}, M_{13}, M_{15} represent another pattern which is constructed by the sub-pattern M_{13}, M_{15} and the sensor M_{14}. For the activity *sleeping*, the most frequent pattern is M_2, M_3. This makes sense because these sensors are on the bed and they will be triggered one after another during *sleeping* activity.

6.5.3 Abnormal Behaviour Detection Results

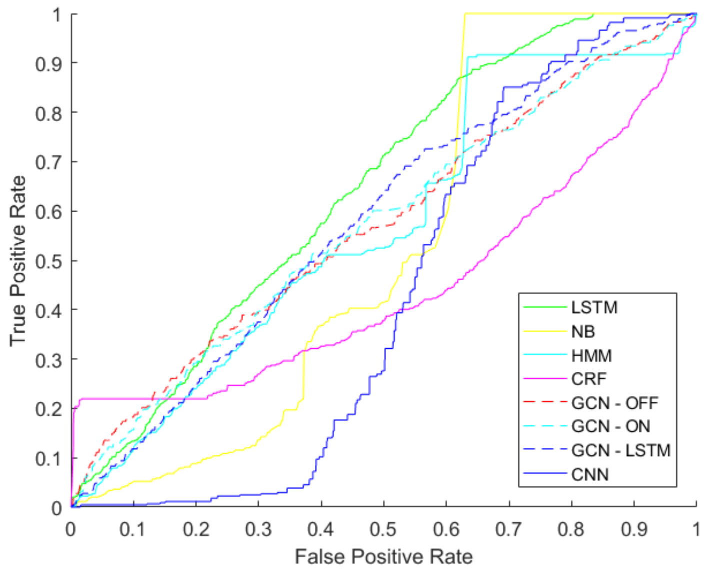
The abnormal behaviour detection results are visualised for *activity* related and *sub-activity* related modified test sets separately on ROC curves in Figure 6.6a. Moreover, AUC histograms of ROC curves are shown in Figure 6.7a. For *activity* related abnormal behaviours, the results show that GCN with only ON and with ON and OFF achieves the best with AUC of 66% and 67% respectively. Adding LSTM layer to the activations of GCN (with ON and OFF) reduces the AUC rate slightly to 63%. The reason for this might be that LSTM encodes temporal information further and tolerates small variations in the sequence. Adding OFF status to GCN model doesn't improve the result too much (around 1%), since *activity* related abnormality doesn't occur at sensor activation level.

Although CNN and LSTM are powerful models, they perform slightly worse than GCN models (with AUC of 57% and 59% respectively), since they rely on BOS representation. However, CNN catches GCN models at TPR (82%) and FPR (56%) on ROC curve. NB performs the worst because of its simple modelling capabilities. HMM and CRF, with AUC of 42% and 54%, doesn't perform well because of BOS representation.

Results for anomaly detection with *sub-activity* related test set are shown in Figures 6.6b and 6.7b. LSTM performs the best AUC (63%), since inserting additional sensor activations (see Chapter III), changes BOS representations. Thus, the activations mistakenly appear and LSTM can relate these changes by encoding temporal



(a) ROC for anomaly detection when *activity related* test is used.

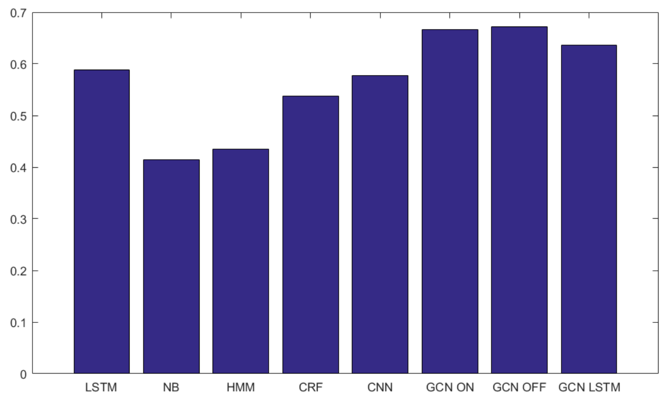


(b) ROC for anomaly detection when *sub-activity related* test is used.

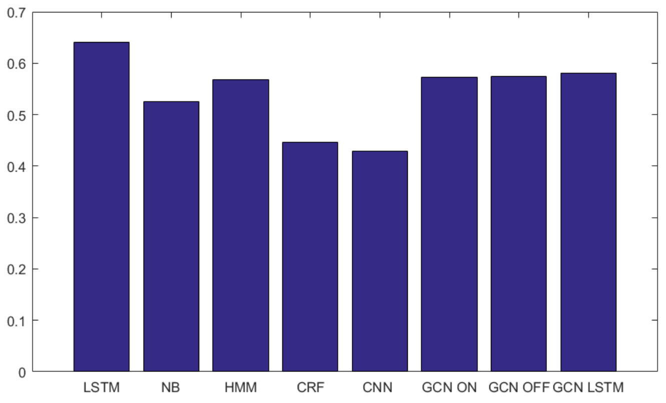
Figure 6.6: ROC curves for abnormal behaviour detection using GCN.

information. GCN-LSTM with ON and OFF produce AUC rate of 57% while GCN with only ON and GCN with both ON and OFF have similar AUC rate 56%. Thus, adding LSTM to GCN with ON and OFF improves the results reaching an FPR of

73% and TPR of 58% on ROC. Although, adding OFF helps differentiating between classes in activity recognition, it also increases the noise to detect abnormal activities. NB, with AUC of 52%, doesn't perform well since it is a simple model and can relate neither temporal nor spatial information. Although HMM can relate previous input to the current one, it achieves an AUC of 54%. As a discriminate model CRF performs only an AUC of 54%.



(a) AUC for abnormal behaviour detection when *activity related* test is used.



(b) AUC for abnormal behaviour detection when *sub-activity related* test is used.

Figure 6.7: AUC histogram for abnormal behaviour detection using GCN.

6.6 Conclusion

This chapter introduces a method to detect abnormal behaviours reflecting cognitive status of elderly people, in the natural flow of daily life in a smart home. Instead of relying on fixed length feature vectors, we work on raw sensor measurements to encode information coming from sensor activations such as frequency and order of the activations. We represent raw sensor activations in a graph and use Graph Convolutional Networks (GCN) to learn activity labels of nodes and the detect abnormal activities. The results show that the proposed GCN based method can encode the low-level information coming from activations and flag abnormal behaviours in the context of dementia. However, this method cannot relate one instance to another and neglects temporal information between time-slices.

CHAPTER VII

Conclusions and Future Work

This chapter summarises the contributions and key findings of this dissertation and discusses the future directions.

7.1 Conclusion

This research aimed to detect abnormal behaviour of dementia sufferers in the daily flow of life in a specially designed smart home. Elderly people suffering from dementia tend to confuse things, or repeat and skip certain activities in their daily life patterns. Firstly, a method was proposed to generate abnormal activities related to dementia. For this purpose, two types of abnormal behaviour were generated, which are activity and sub-activity related anomalies. Secondly, deep learning methods such as Recurrent Neural Networks, Convolutional Networks, Recursive Auto-encoders and Graph Convolutional Networks were exploited to detect abnormal behaviour. Thirdly, we compared these methods with the traditional machine learning methods such as Hidden Markov Models, Conditional Random Fields, Support Vector Machines and Naïve Bayes methods. Our results showed that the proposed methods were competitive with the machine learning methods.

First of all, given the scarcity of real-world data available, we proposed a method to manually synthesise abnormal activities to mimic the daily life patterns of elderly people suffering from dementia (Chapter III). We modified some instances of datasets already available in the literature. The modification was done through injection of abnormal activities in normal daily life activity sequences. We specifically aimed to simulate repetition, sleep disorder and confusion types of abnormal behaviour. Repetition and sleep disorder related abnormal behaviour occur at activity level while confusion related abnormal behaviour occur at sub-activity level of activities. However, generating abnormal behaviour in this method may fail to reflect user specific abnormal behaviour since people show different habits in different times.

Firstly, activity recognition problem was emulated as a sequence labelling problem where time-series data of sensor measurements was treated as a sequence (Chapter IV). Then, daily life patterns were modelled exploiting Recurrent Neural Networks (RNNs) and its variants, namely Long Short Term Memory, Gated Recurrent Units and Vanilla versions, and Convolutional Neural Networks (CNNs). Results with RNNs and CNNs showed that capturing the activity instances and their temporal and spatial relationship was helpful to understand the daily life patterns and their relationships with other activities. With RNNs, we were able to encode the temporal information in daily life in order to model personal environments for dementia support. With CNNs, we tried to extract meaningful and fruitful patterns in the activities and with the help of these patterns, activities were recognized and anomalies were detected. These models were good at detecting anomalies related to repetition of activities and sleep disturbances. However, they are not capable enough to detect anomalies related to confusion since they cannot model sub-activities in an activity. RNNs and CNNs cannot model daily activities at granular-level, so they cannot understand which sub-activities in these activities are forgotten, repeated and confused. Modelling these sub-activities would give a better understanding of activity recognition.

Moreover, a new representation, namely Raw Sensor Measurement (RSM) that captures the intrinsic structures of activities such as the frequency and the order of sensor activations was proposed (Chapter V). Then, the abnormal behaviour of dementia sufferers was detected by exploiting the granular level sub-activities in activity instances. For this purpose, Recursive Auto-encoders (RAE) and their linear and greedy variants were adopted to model activities from their sub-activities hierarchically. Abnormal activities were then detected using RAE's reconstruction error. The results showed that this method was promising to model activities from their sub-activities and detect anomalies. However, this method couldn't relate one instance to another and neglected temporal information. Combining RAEs with a RNN-based model would include temporal information and detect abnormal behaviour occurring not only at hierarchical level but also at temporal level.

Also, we proposed to investigate Recursive Auto-Encoders (RAE)-based transfer learning to cope with the problem of scarcity of data in the context of abnormal behaviour detection (Chapter V). In the absence of training data, it would be helpful to learn the normal behaviour and daily life patterns of a (cognitively) healthy person and use them as a basis for tracking other patients. The empirical results showed that the proposed method is promising when supervised methods cannot be exploited because of the lack of (labelled) training data. The proposed method failed to detect the person's specific abnormal behaviour. Moreover, the behaviour across different houses is transferable under the condition that the resident profiles such as age, gender and lifestyle are similar in both households. Also the different sensors and activities in these houses should be mapped into each other. When there is no source data available prior, domain adaptation would not be possible. Transfer learning generalises the behaviour of inhabitants between different houses and doesn't take resident specific behaviour into account. For example, going to toilet during sleep might be normal for a person, while it is abnormal for another person. Then, there would be a problem to detect resident specific abnormal behaviour.

Lastly, instead of relying on fixed length features as considered in the literature, we exploited raw sensor activations to encode sub-activity related granular level information since it allows us to investigate more into the activation frequency and ordering of sensors (Chapter VI). This representation is helpful to understand the intrinsic sub-structures of activities. Then, we considered the problem of activity recognition as a graph labelling problem and exploit Graph Convolutional Networks (GCNs) to model activities based on their fine-grained sensor activations. Modelling activities in a graph gives the opportunity to encode the intrinsic sub-structures of activities. Then abnormal behaviour related to dementia was detected exploiting the nodes and their relationships in the graph. The results showed that the proposed GCN based method can encode the low-level information coming from activations and flag abnormal behaviour in the context of dementia. However, this method cannot relate one instance to another and neglects temporal information between time-slices.

7.2 Future Work

One possible extension of this research would be to take OFF status of the sensors into account and investigate their impact to detect anomalies. Also specific features or rules can be considered to understand if a sensor (e.g. item sensors representing medicine, or a fridge door sensor to check if it closed or not) is left OFF or not. New features to represent sensor events could be proposed to cover more different types of abnormal behaviour reflecting cognitive status of elderly. For example, incorrectly measuring the oatmeal, not using soap when cleaning, washing hands multiple times, confusing the location of items, and using too much soap or leaving kitchen utilities on, could be important in the context of dementia.

Also, our current approach may fail to detect abnormalities when there is gradual deterioration regarding the health of an elderly. Thus, a real-world data can be collected in which gradual deterioration is observed. Moreover, different types of abnormal behaviour reflecting dementia specific habits can be considered. If no real-world data can be collected, simulation methods to artificially generate these kinds of behaviour can be proposed. Moreover, in terms of transfer learning, personal habits can be taken into account by learning a prior distribution from the source houses to adapt the model to the target house.

One drawback of this thesis is that GCNs and RAEs cannot relate one instance to another and neglects temporal information between time-slices. Thus, in future, these models can be merged with Recurrent Neural Networks to add temporal information into account.

APPENDICES

APPENDIX A

Noname manuscript No.
(will be inserted by the editor)

Detection of Abnormal Behaviour for Dementia Sufferers using Convolutional Neural Networks

Damla Arifoglu, Abdelhamid Bouchachia

Received: date / Accepted: date

Abstract In recent years, there is a rapid increase in the population of elderly people. However, elderly people may suffer from the consequences of cognitive decline, which is a mental health disorder that primarily affects cognitive abilities such as learning, memory, etc. As a result, the elderly people may get dependent on caregivers to complete daily life tasks. Detecting the early indicators of dementia before it gets worsen and warning the caregivers and medical doctors would be helpful for further diagnosis. In this paper, the problem of activity recognition and abnormal behaviour detection is investigated for elderly people with dementia. First of all, the paper presents a methodology for generating synthetic data reflecting on some behavioural difficulties of people with dementia given the difficulty of obtaining real-world data. Secondly, the paper explores Convolutional Neural Networks (CNNs) to model patterns in activity sequences and detect abnormal behaviour related to dementia. Activity recognition is considered as a sequence labelling problem, while abnormal behaviour is flagged based on the deviation from normal patterns. Moreover, the performance of CNNs is compared against the state-of-art methods such as Naïve Bayes (NB), Hidden Markov Models (HMMs), Hidden Semi-Markov Models (HSMM), Conditional Random Fields (CRFs). The results obtained indicate that CNNs are competitive with those state-of-art methods.

Keywords Smart Homes · Sensor based Activity Recognition · Convolutional Neural Networks · Long Short Term Memory Recurrent Neural Networks · Dementia · Abnormal Behaviour Detection

Department of Computing and Informatics
Faculty of Science and Technology
Bournemouth University, UK
E-mail: {darifoglu, abouchachia}@bournemouth.ac.uk

1 Introduction

Studies indicate that by year 2030, the number of people aged 65 to 74 will be about 3% of the total population [1]. Elderly people may suffer from the consequences of dementia, which is a condition that causes problems with mobility, physical and mental abilities such as memory and thinking [2]. It may also cause decrease in the ability of speaking, writing, distinguishing objects, performing motor activities and performing complex functional tasks (paying bills, preparing a meal, etc.) [3]. An elderly person having such cognitive decline loses independence in daily life and requires care and support from caregivers. On the other hand, the use of assisted living technologies such as smart homes can substantially help a person with dementia to live independently. Unfortunately, currently there are no dementia friendly smart homes addressing elderly people's special needs.

Cognitive diseases, like dementia, need to be detected at an early stage so that early treatment will be possible. However, research shows that 75% of dementia cases go unnoticed [4] and many cases are diagnosed only when the impairment reaches moderate or advanced stage. The best markers of cognitive decline may not necessarily be detected based on a person's performance at any single point in time, but rather by monitoring the trend over time and the variability of change in a duration [5]. Most common types of dementia (Alzheimer, Parkinsons disease) can be identified by behavioural changes like sleep disturbances, difficulty of walking and inability to complete tasks. Thus, such changes can provide key information about memory, mobility and cognition of a person. For instance, an old person suffering from Alzheimer may forget to have his lunch, take multiple lunches instead, wake up in the middle of the night, go to the toilet frequently, or have dehydration problems because of forgetting to drink daily amount of water. In particular, the daily home activity involving basic functions like preparing food, showering, walking, sleeping, etc. can be used to assess the well-being of elderly people.

The development of ambient home assessment environments has begun to provide the opportunity to assess behaviour change unobtrusively in real-time [6,4,5]. Prevention or delay of dementia onset is contingent upon the ability to detect early, meaningful, cognitive change during the life course [6,4]. The identification of early onsets of dementia using non-medical diagnosis methods requires the development of new diagnostic tools. Although a few promising methods have been experimentally validated [6,7,8,9,10], the translation of the current knowledge into smart homes still requires more dedication and work. Current assessment methods mostly rely on queries from questionnaires or in-person examinations, which depend on recall of events or brief snap-shots of function that may poorly represent a person's typical state of function. Also the clinical methods have some limitations such as their episodic nature, and possible biased reporting. The main motivation for our work is that cognitive decline can be observed in daily activities and routines of an elderly person. Real-time monitoring of activities performed by an elderly person in a smart home would be beneficial for early detection of such decline.

In machine learning, a convolutional neural network (CNN) is a class of deep, feed-forward artificial neural networks. Recently, CNNs are popular due to their ability to learn fruitful representations and capture local dependency and spatial information of granular-level patterns. For example, in image recognition, CNNs firstly detect pixels, then edges and shapes, then parts of objects as the layer level increases. Similar to images, there are granular-level patterns in daily life activities. For example, when the activity *preparing coffee* is considered, it is seen that this activity is constructed by many steps such as getting closer to the sink, turning the water on, filling the coffee machine with water and turning the machine on, etc. In [11], granular-level activity patterns, which they call as movement vectors, are extracted by using a decomposition based unsupervised approach. It is shown that the movement vector can distinguish different high-level activities. The occupant tends to have the same routine of performing the same activities, but has different movement patterns in different activities. For example, the occupant may mainly move around the kitchen sink in *Wash Dishes* activity, and stay around the bedroom area during *Sleeping* activity. A combination of some motion sensors are mostly seen in the instances of *relax* activity while usage of some other motion sensors indicates the movement between kitchen range and the sink and this movement pattern is seen in the instances of *wash dishes* activity. CNNs are good at modelling these granular-level patterns and defining their relationship with each other by using spatial information. Thus, in the present study, CNNs are exploited to model sensors and their relationship with each other in daily life activity recognition.

Unfortunately, there exists no publicly available dataset on abnormal behaviour of people with dementia. Producing such a dataset requires time and adequate experimental environment. When there is no real-world dataset available, data simulation can be a solution [12,13,14,15]. Given the scarcity of such data, simulating daily life abnormal behaviours of elderly people suffering from dementia would be helpful for providing automatic assessment methods. Thus, in this paper, a method is proposed to artificially produce abnormal activities reflecting on typical behaviour of elderly people with dementia.

In a nutshell, the present paper introduces the following contributions.

1. A method is proposed to generate synthetic data that simulates the abnormal behaviours of people with dementia.
2. To the best of our knowledge, our study is the first to apply CNNs, thanks to their ability to model granular-level patterns, for daily life activity recognition and dementia related anomaly detection task.

The rest of the paper is organised as follows. Section 2 provides an overview of literature work. Section 3 presents the details of the proposed methodology together with the datasets and models used. Section 4 describes the experimental set-up and results of the experiments followed by a discussion. Finally, Section 5 concludes the paper.

2 Literature Review

In-home automatic assessment of cognitive decline has been the subject of many studies [16,17,6,7,18,19,20]. Many machine learning approaches such as SVMs and Naïve Bayes methods [21,22], Restricted Boltzmann Machines (RBMs) [19] and Markov Logic Networks [7,18,20], Hidden Markov Models (HMMs) [14], Random Forest methods [15], and Hidden Conditional Random Fields [23] have been exploited.

In some studies [16,24], assessment of cognitive status is done by providing patients some instructions during the completion of pre-defined tasks (e.g., sweeping the kitchen). In the end, the patients receive scores which are calculated based on the time spent, the frequency of the sensor triggered, etc. These scores are used to assess the cognitive status of elderly people. In [16], cognitive decline assessment is done by asking elderly people to complete a sequence of scripted instrumental activities of daily living (IADLs). The participants are monitored via a camera while they perform tasks such as cooking oatmeal on the stove and in the end, they receive scores by trained experts. In [24], the authors first extract sensor based features (the duration the activity and the number of sensors triggered) and then use SVMs, NB and neural networks to assess the activity quality and cognitive status of elderly people in smart homes. However, participants are provided with a brief description of each sub-task that they should refer to during the simulated activities such as planning a bus route, finding a recipe in the recipe book. These studies fail to provide an unobtrusive way of assessment since they are not done in the natural flow of daily living and in real life scenarios. Moreover, using rule-based systems, an expert is needed to manually integrate specific rules to the system since every person has own daily life routines. For example, waking up and drinking water in the middle of the night might be normal for a person, while abnormal for some other person. However, our approach does not require any expert knowledge, since it learns what is normal and abnormal from the training data automatically. Specifically, we aim in this study to detect anomalies in the natural flow of daily living without giving any instruction and considering not only some time interval, but everyday living scenario.

Some studies [25,26] focus on anomalies related to the duration and the timing of performed activities and other type of anomalies related to dementia such as repetition of activities are not taken into account. In [25], the authors introduce *activity curves* which models an individual's generalised daily activity routines based on automatically recognized activities. Deviations in behavioural routines are detected by comparing *activity curves* in order to do health assessment. In [26], the authors use a probabilistic model based on the location and outing interference of each activity. Then cross-entropy measure is used to detect anomalies such as staying in bed for a long time or not using the bedroom for sleeping during the night. In [14], the authors exploit HMM and fuzzy rules to detect duration, time and frequency related anomalies.

In the literature, there is some work dedicated to the synthesis of activity related data [12,14,15]. In [14], the authors modified real-world dataset in order to synthesise health related abnormal behaviours for their experiments. 8 daily activities such as sleeping, waking up, walking, eating are chosen and health related abnormal behaviours like frequent toilet visit, no exercise, slept without dinner are synthesised. In [15], more data is synthesised using HMMs based on a small set of real data collected. To increase the realism of data simulation, the sensor events were modelled by a combination of Markov chains and the Poisson distribution. However, in both [14,15], it is not mentioned in detail how the data synthesis was done. In [12], the authors modified a real-life dataset of an older adult converting basically the rooms into activities. The authors focused on walking and eating in conjunction with the sleeping activity and samples of these activities are manually inserted in the XML data set.

In [23], the authors exploit Hidden State Conditional Random Field (HCRF) method to detect abnormal activities that often occur in homes of elderly people by considering sub-activity relations. First, HCRF is used to recognise activities by producing a recognition confidence value for each activity. Then, a threshold based method is used to decide the activities as normal and abnormal. In [7], the authors detect anomalies of mild cognitive impairment by exploiting Markov Logic network. They use a hybrid technique including supervised learning, rule-based reasoning and probabilistic reasoning. However, they construct their model prior by defining each steps of each action. Those rules strongly depend on the specific home environment, on the used sensors, and on the particular habits of the elderly people; hence, their definition is time-expensive, and rules are not portable to different environments. In order to address this issue, the same authors propose a method to automatically learn the rule-based definitions of behavioural anomalies [18]. They exploit formal rule induction methods and a training set of normal and abnormal behaviours. However, the authors claim that their proposed rule learning method infers deterministic rules, which are prone to generate anomaly mispredictions in the presence of noise from the sensor infrastructure. In our study, normal daily life patterns are learnt for each individual from training data automatically and without the integration of any rules. Similar to [23,7], in our proposed work, anomaly is defined not activities alone but defined in the context of sequences, with other activities happened before and after.

Recently, there has been growing interest in CNNs [27,28,29,30,31,32,33, 34,35,36], Deep Belief Networks (DBN) [37], Restricked Boltzman Machines (RBMs) [38,19,37,39] and Recurrent Neural Networks (RNNs) [29,30,40,41]. In [38], RBMs are used for feature extraction and selection from sequential data. In [39], results with RBM on CASAS dataset outperformed HMMs and NB in most of the cases. In [29], the authors use a combination of CNNs and Long Short Term (LSTM) RNNs to do multi-modal wearable activity recognition. In [30], the authors explore deep, convolutional and recurrent approaches on movement data captured with wearable sensors. Moreover, they describe how to train recurrent models in this setting and introduce a novel regularisation. In [34], the authors utilised convolutional networks to classify activities using time-series data collected from smart phone sensors. In [35], in a real world setting, an automatic stereotypical motor movement in Autism detection systems is developed exploiting CNNs. The discriminating features from multi-sensor accelerometer signals are learnt via CNNs and this knowledge is transferred to a new dataset. In [36], CNNs are exploited to learn features from raw physiological signals in an unsupervised manner analysis and then using multivariate Gaussian distribution, anomalies are detected to identify latent risks. In our previous work [41], RNNs are exploited to detect anomalies related to dementia in a daily living scenario.

CNNs have been exploited for activity recognition using movement datasets that are generated by wearable sensors [38,27,28,29,31,34,37]. Except the work by Fang et al. [39,41], none of these studies focus on daily activity datasets collected by sensors placed at home. Previous work on activity recognition based on wearable sensor datasets shows that CNNs and RNNs are useful to recognise activities, but leaves a lot of room for improvement. In this work, CNNs and their combination with LSTMs are investigated on daily activities datasets, namely Aruba [42] and WSU testbeds of CASAS smart home datasets [22] since the activities in these datasets are good examples to reflect daily life patterns of elderly person and to synthesise anomalies related to dementia.

3 Methodology

To assess CNNs in daily life activity recognition and abnormal behaviour detection tasks, the following steps are proposed: Firstly, a real-world dataset is modified in order to simulate abnormal behaviours related to dementia. Secondly, this dataset is segmented into time-slices by using a sliding window approach as described in [43]. Thirdly, sensor-based raw data is mapped into *last-fired* representations as described in [43]. Fourthly, CNNs are trained to recognise daily activities and encode daily-life behaviour routines. Lastly, the trained model is used to detect anomalies deviating from the normal daily-life sequences. In the following, the datasets are described as well as the methodology used to generate artificial dataset that reflects on the typical behaviour of a person with dementia.

3.1 Dataset

In this study, two datasets are used to evaluate activity recognition and abnormal behaviour detection. These datasets are namely Aruba [42] and WSU testbeds of CASAS smart home project [22].

In Aruba testbed, motion, door and temperature sensors are used. However, temperature sensors are excluded in this study and other 34 sensors (3 door and 31 motion sensors) are used. The data is provided as a list of (sensor, time-stamp) sensor measurements. In this dataset, there are 11 daily activities performed by a single user and it spans 224 days. These activities are *Meal*

Preparation (1606 instances), *Relax* (2910 instances), *Eating* (257 instances), *Work* (171 instances), *Sleeping* (401 instances), *Wash dishes* (65 instances), *Bed to toilet* (157 instances), *Enter home* (431 instances), *Leave home* (431 instances), *Housekeeping* (33 instances) and *Respirate* (6 instances). The activities performed in this dataset are totally normal and some of these normal activities will be modified for anomaly detection.

In WSU testbed [22], there are 5 activities, which are *Make a phone call*, *Hand washing*, *Meal Preparation*, *Eating*, *Cleaning*. There are 20 instances of each activity performed by 20 students in both *adlerror* and *adlnormal* versions. The *adlnormal* version consists of totally normal behaviours while in the *adlerror* version, there are specific errors in the task completion of these activities. Errors were selected to reflect common difficulties that can compromise everyday functional independence. The participants are told to include these errors during their performance. These errors can be seen in daily life activities and activity patterns of elderly people who are suffering from the consequences of cognitive decline.

3.2 Synthesis of Abnormal Activities Related to Dementia

This study aims to detect the following 3 different kinds of anomalies that can be seen in daily-life routines of elderly people with dementia: 1) Repeating activities, 2) Disruption in sleep, and 3) Confusion (getting confused during the activities).

1) Repeating activities: Elderly people suffering from dementia may forget whether they performed a particular daily activity or not, so they may repeat that activity. Frequency sensitive activities such as having a snack or drink, brushing teeth, taking medicine multiple times, etc. are the ones only the number of occurrences matters in terms of medical assessment. For instance, an elderly person suffering from Alzheimer may forget to have lunch, take multiple lunches instead [44], may forget to have dinner and start to prepare it in the middle of the night.

To reflect on this cognitive problem, we generate this kind of abnormal activities by manually inserting a specific set of actions within a random area of the normal activity sequence. This will result in multiple occurrences of that activity, which will occur in some inadequate time of the day such as having dinner in the middle of the night. We inject the instances of the following activities: *brushing teeth, preparing dinner, eating, getting snack* into the normal activity sequences to generate abnormal activities related to the frequency. For example; let's assume that S is a sequence of activities occurring in a day such as $S = d_1, d_2, d_3, ..., d_x, b_1, b_2, ..., b_t, d_{x+1}, ..., d_n$ where each d_i is a time-slice of some activities and each b_j is a time-slice of *brushing teeth* activity. Here, there are t time-slices of *brushing teeth* activity which consecutively results in only one instance of *brushing teeth* activity in the whole day sequence. Then, time-slice instances of *brushing teeth* activity are injected into the sequence S to have the abnormal version. Then modified S becomes

$S = d_1, d_2, d_3, ..., d_m, b_1, b_2, ..., b_t, d_{m+1}, ..., d_a, b_1, b_2, b_3, ..., b_k, d_{a+1}, ..., d_n$. As a result, we have two occurrences of *brushing teeth* activity in the sequence.

2) Disruption in sleep: Degeneration of the sleep-waking cycle and night time wandering are among the most severe behavioural symptoms of dementia. For example, elderly people may wake up many times in the night to use the toilet and go back to sleep or may forget to take daily amount of water [44, 45].

We simulate these anomalies by inserting some specific synthetic activities in the normal night-time activity sequences of a person. More specifically, we inject *Eating, Bed to Toilet* into a random area of *sleeping* activity in the normal daily activity sequence. This will emulate the activities of getting drink and going to the toilet frequently in the middle of the night. For example; given a sequence of *sleeping* activity such as $S = s_1, s_2, s_3, ..., s_n$ where each s_i is a time-slice; time-slice instances of *getting drink* are injected into a random area of S. Then modified S becomes $S = s_1, s_2, s_3, ..., s_m, d_1, d_2, d_3...d_k, s_{m+1}, ..., s_n$ where each time-slice d_j is from *getting drink* activity. As a result, we simulate disruption in sleep anomaly where the person wakes up in the middle of the night and gets drink.

3) Confusion: Older adults suffering from cognitive decline tend to confuse things and perform some steps of activities more than once during the completion of activities. For example, they may fail to remember how to turn a CD player on, or may forget to turn off the television, air conditioning or house utilities such as kettle, oven, or they may leave the refrigerator door, the main door open. In order to test our methods on these kind of anomalies, the *adlerror* set of WSU dataset is used since confusion and forgetting anomalies are reflected in this dataset. For example, leaving the water running after washing hands, leaving the burner on after cooking the oatmeal, forgetting to take medication with the meal, wiping off the dishes without using running water to clean them are some examples to these kind of anomalies in the WSU *adlerror* set.

The first two types of anomalies are simulated by modifying Aruba testbed. Here, there is only one subject in the dataset. The lifestyle in the training data is taken as a norm and then we synthesise the abnormalities deviating from this norm and introduce these abnormalities in the test data. These activities are totally normal on their own but they become abnormal when they occur at a wrong time of the day and after or before a specific activity. Hence, capturing these abnormalities within the context is important. In all, 150 abnormal activity slices are generated manually. The third anomaly type is already reflected in WSU dataset; thus it is used directly without modifying any sensor reading.

3.3 Sensor Reading Representation

Firstly, time-slice chunks are extracted from raw sensor readings via a sliding window approach [43]. Data is discretised using the time-slice length of 60 seconds. A time-series chunk is a matrix of $t \times f$ size, where t is the length of time-slices and f is the number of sensor features. Then raw sensor readings are mapped into *last-fired* representation. Last-fired representation indicates which sensor is fired last. The sensor that changed state last continues to give 1 and changes to 0 when another sensor changes state. Previous work [41] shows that this representation gives better activity recognition accuracy rates than other representations as proposed in [43].

In the following, a description of CNNs used in this work is given.

3.4 Convolutional Neural Networks (CNNs)

CNN takes inputs of dimensions $h \times w \times d$, where h is the height of the input matrix, w is the width of the input matrix and d is the number of different channels of the input matrix. In our study, d is 1 since time-slice input matrices has only one channel.

A local filter (kernel) with a size of $n \times m \times q$ is used to extract fruitful feature patterns and capture local dependencies on the given input. Here, n is the number height of the filter, m is the width of the filter, while q is the number of filters used. These values are given as a parameter during the network construction process. The weight of these filters are initialised randomly in the beginning and then CNN learns these weights on its own during the training process by optimising the values. In this study, random uniform initialisation is used to initialise the filters and Stochastic gradient descent is used to optimise the values during the training. An additional operation called activation function has been used after every convolution operation. In this study, Rectified Linear Unit (ReLU) is used as the activation function. Then a max-pooling layer, which is followed by a fully connected layer is added to the network. The fully connected layer used in our network is a traditional Multi Layer Perceptron that uses a softmax activation function in the output layer. The purpose of this layer is to use these features for classifying the input image into various classes based on the training dataset.

CNNs can contain one or more pairs of convolutional and max-pooling layers, where higher layers use broader filters to process more complex parts of the input. The top layers in CNNs are stacked by one or more fully connected normal neural networks. These fully connected neural network are expected to combine different local structures in lower layers for final classification purpose. In the training stage, CNN parameters are estimated by standard forward and backward propagation algorithms to minimise objective function.

3.4.1 Activity Recognition and Abnormal Behaviour Detection

In order to recognise daily activities, training instances of the datasets and their corresponding labels are fed into CNNs to be trained. The models assign a class label to each instance with a confidence value. Firstly, the mean of confidence values of training instances for each class is calculated as follows.

$$m_j = 1/N \sum_{t=1}^{N} p_t \tag{1}$$

where m_j is the mean confidence value of class j and p_t is the confidence value for training instance t of that class and N is the total number of instances in that class.

Then when a new test instance is introduced, if the model assigns it to a class with a confidence value which is bigger than the mean of that class (m_j), that instance is considered as a normal activity, otherwise it is flagged as an abnormal activity.

In order to test the affect of convolutions on different dimensions and different architectures, the following networks are tested on Aruba dataset (see Figure 1). Here, the input matrix is $N \times M$, where the rows are sensor readings for each time-slice and columns are the values of each sensor as time passes.

1D Convolution: In this model, convolution is done on temporal dimension. As depicted in Figure 1a, in the convolutional layer, 100 filters with a length of 10 is used. 1D convolution is followed by a max-pooling layer, which has a stride of 2. Then another convolutional layer (with 50 filters and a length of 5) and a max-pooling layer are added. After the extracted features are flattened, these features are fed into dense layers (3 hidden layers having 512, 128 and 50 units respectively) and then the final decision is given by a softmax layer producing the confidence values of assigned class labels.

2D Convolution: In this model, convolution is done on both of the dimensions, specifically on feature and temporal dimension. 100 filters with a size of 10×34 are used in the first convolutional layer which is followed by a 2×2 max-pooling. Then another 2D convolution operator is added this time with 20 filters with the size of 5×34. The flattened features are fed into the same dense layer and the softmax layer described above.

CNN and LSTM (2D CNN + LSTM): CNNs can learn spatial relationships on a given $N \times M$ input but they cannot relate a current input to the next one in the occurrence order of the input sequence. To overcome such limitation, LSTMs are used at the end of the CNN network. In this combination, firstly, the 2-layer 2D-CNN described above is used to learn the fruitful feature representation. And then the extracted feature maps are fed into LSTM layers which will be taking further temporal information of the slices into account. LSTM has hidden layers of size 30×50 respectively. LSTM layer is followed by a dense layer with 128 hidden units and then another dense layer with 50 units. Eventually, softmax layer classifies the input into one of the activity classes with a probability value.

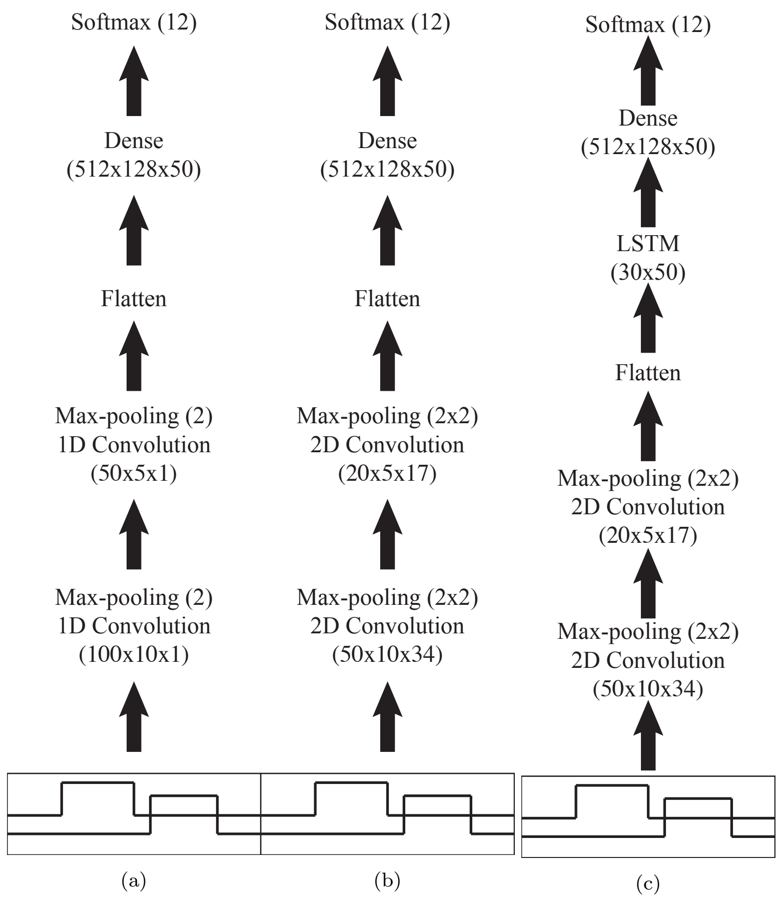


Fig. 1: Convolutional neural network architectures used. (a) 1D convolutional along temporal dimension (b) 2D convolutional both along temporal dimension and feature dimension (c) 2D convolution followed by an LSTM layer.

4 Experiments

In order to evaluate our methods, first the datasets are splitted (see Sec. 3.1) into train and test sets. However, the split is not done with a traditional split method since dividing daily activity datasets based on a fixed time period such as day is more meaningful [17]. Aruba testbed was collected in 224 days, thus 70 days are used as test, 15 days for validation and the remaining days are used for training. The first WSU set *adlnormal*, representing normal behaviours, are used to train the classifiers, while the second set, *adlerror*, containing the abnormal activity, is used for test set.

Keras Deep Learning library's [46] and Theano's [47] implementations of the CNNs and LSTM are used in this study. Moreover for the sake of comparison, results with NB, HMM, HSMM and CRF are provided which are based on the implementation provided in [43]. In the CNN experiments below, Adam optimiser [48] is used and the instances are fed into the system with a batch size of 20. In the following, the evaluation metrics are explained further.

4.1 Evaluation Metrics

In order to assess the activity recognition success, the following metrics are used: Precision, Recall, F-measure and Accuracy. As seen in Formula 2 and 3, final precision and recall values are calculated by taking average over classes. Precision and recall measures are used in order to show how well the models perform on imbalanced datasets like the one in this study. On the other hand, the accuracy represents the percentage of correctly classified time slices, therefore more frequently occurring classes have a larger weight in this measure. Here, TP is true positive, TT is total number of instances, TP is total true labels, TI is total of inferred labels, N is the number of classes in a specific class of the dataset and $Total$ is the total number of instances of all classes in the dataset

$$\text{Precision} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TI_i} \tag{2}$$

$$\text{Recall} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TT_i} \tag{3}$$

$$\text{F-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} TP_i}{\text{Total}} \tag{5}$$

Abnormal behaviour detection success rate is evaluated by sensitivity and specificity metrics. Sensitivity or True Positive Rate (TPR) refers to the method's ability to correctly detect instances which are abnormal. Specificity or True Negative Rate (TNR) gives the percentage of correctly recognized normal instances, thus reflects the method's ability to differentiate between normal and abnormal.

$$\text{Sensitivity (TPR)} = TP/(TP + FN) \tag{6}$$

$$\text{Specificity (TNR)} = TN/(TN + FP) \tag{7}$$

4.2 Results

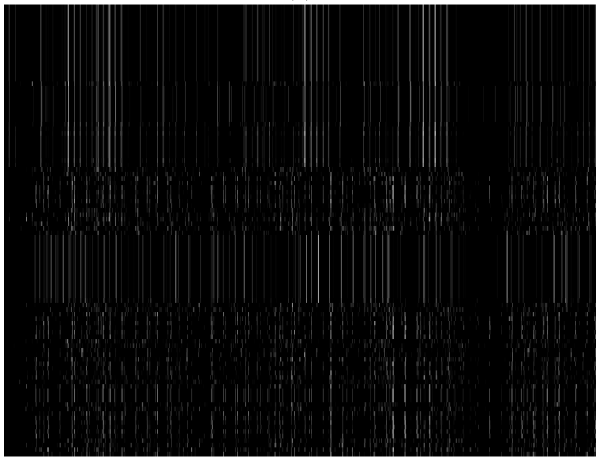
Two types of experiments are performed: 1) Activity recognition, and 2) Abnormal activity detection. Activity recognition success rates by both generative and discriminative methods on Aruba set are depicted in Table 1. The results indicate that CNNs with 2D convolution (accuracy of 89.67%) and also CNN-2D followed by an LSTM classifier (accuracy of 89.72%) outperforms CRF (accuracy of 88.58%). The reason is CNNs extract their own fruitful features while CRF only relies on the given input. HMM and HSMM give the worst accuracy results (77.90% and 77.98% respectively). NB gives better accuracy result (84.37%) than HMM and HSMM but it results in lower precision (42.87%) and recall (61.04%) rates. Although HMM and HSMM give the best recall rates (72.03% and 71.56%), they fail in giving good precision rates (43.66% and 43.97% respectively). It is seen that CNN-1D network has an accuracy of 87.50% while it fails in high precision (31.42%) and recall (36.78%) values. CNN-1D extracts features on temporal dimension, so it takes temporal information within a time-slice chunk into account but on the other hand, it ignores the relationship between sensors since it doesn't do convolution on the feature dimension. Thus, it doesn't learn class specific feature maps to differentiate between different classes resulting in low precision and recall. When 2D convolution is used, both temporal and spatial information are taken into account and the networks learn more informative features. Thus, it gains the ability to learn class specific features, which results in higher precision and recall values (46.84% and 41.68%) and high accuracy results (89.67%). CNNs cannot remember the previous and the next inputs, but feeding the feature maps into an LSTM layer helps us process the temporal dimension further. In result, CNN with 2D convolution followed by LSTM (CNN-2D + LSTM) achieves a precision rate of 51.20% and a recall rate of 50.55% and an accuracy of 89.72%.

Table 1: Activity recognition results with *last-fired* representation on Aruba dataset

Model	**Precision**	**Recall**	**F-measure**	**Accuracy**
NB	42.87%	61.04%	50.36%	84.37%
HMM	43.66%	72.03%	54.36%	77.90%
HSMM	43.97%	71.56%	54.47%	77.98%
CRF	50.24%	52.83%	51.50%	88.58%
LSTM	38.65%	41.29%	39.92%	89.00%
CNN-1D	31.42%	36.78%	33.89%	87.50%
CNN-2D	46.84%	41.68%	44.11%	89.67%
CNN-2D + LSTM	51.20%	50.55%	50.87%	89.72%

In Figure 2, extracted feature maps from first and second layer and the flatten layer are visualised for CNN-2D network as described in Figure 1. It is seen that noise is reduced and more informative features are learnt as the layer level increases. The x-axis represents features while y-axis is time axis

(a)


(b)

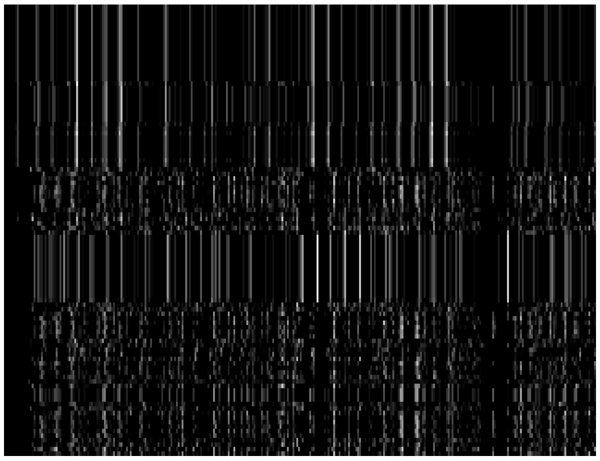

(c)

Fig. 2: a) Extracted features from the first layer. b) Extracted features from the second layer. c) Extracted features from the flatten layer. The x-axis shows time while y-axis represents features. Successive model layers learn deeper intermediate representations. The features get more discriminative and visible in the last layer.

and the white pixels are activations of neurons. It is seen that as the times passes, different activities give different features.

Moreover, we calculate Cohen's Kappa statistics in order to show the robustness of the proposed method, CNN-2D classifier. Kappa statistics is measure that can handle well on both multi-class and imbalanced class problems. It tells how much better the classifier is performing over the performance of a classifier that simply guesses at random according to the frequency of each class. It is thought to be a more robust measure than simple percent agreement calculation, since Kappa takes into account the possibility of the agreement occurring by chance [49]. The calculated Kappa statistics for CNN-2D classifier is 0.64431, which is a substantial agreement according to [49].

Results on Aruba dataset show that classifiers are mostly successful in detecting the instances of *leave home* and *enter home* activities since they are the only activities involving door sensors, thus they are not confused with any other activities. Moreover, *meal preparation* activity is confused with *wash dishes* activity most of the time since they involve same kind of sensors and they both take place in the kitchen. Also *house keeping* activity is generally confused with *work* activity since they may take place in the same room and may involve same sensors.

Table 2: Abnormal behaviour detection results on Aruba Modified dataset

	Aruba Modified		**WSU**	
Model	**Sensitivity**	**Specificity**	**Sensitivity**	**Specificity**
NB	99.33%	33.89%	46.17%	98.42%
HMM	45.54%	27.71%	100%	50.55%
HSMM	100%	35.61%	100%	42.89%
CRF	100%	66.03%	47.87%	72.17%
LSTM	98.67%	75.48%	86.50%	77.89%
CNN -2D	85.33%	33.89%	88.70%	67.46%

The second experiment, abnormal activity detection is performed firstly on modified Aruba set. As a representative of CNN networks, the results are presented with the CNN-2D network. After training the models with normal behaviours, test set which includes the abnormal behaviours is introduced to the classifier and activity instances which are assigned a label with low confidence values are flagged as abnormal. In Table 2, it is seen that the highest specificity is achieved by LSTM networks giving an accuracy of 75.48% (and sensitivity rate of 98.67%). Although NB, HMM and HSMM models gives higher sensitivity rates (99.33%, 100%, 100% respectively), the specificity rates are smaller (33.89%, 27.71% and 35.61% respectively). HMM gives the worst results (a sensitivity rate of 45.54% and a specificity rate of 27.71%). CNN-2D gives a sensitivity rate of 85.33% and a specificity of 33.89%. This shows that LSTMs are more suitable to detect repetition and order related abnormal activities since it can relate current input with the upcoming ones what CNN cannot do.

The second part of anomaly detection experiments are performed on WSU testbed. 30 second time-slice chunks are extracted from sensor readings from WSU. This dataset is not collected in a daily life scenario, thus sensor readings are not in a sequential order. Thus the sensor readings are available only for activities labelled in the dataset. The *adlnormal* set is used as training set and the *adlerror* set is used as test dataset. The aim here is to see how successful the classifiers are to detect the anomalies, given normal behaviours. The results in Table 2 indicate that the highest sensitivity rate is given by HMM and HSMM (both 100%), while HMM gives a specificity rate of 50.55% and HSMM achieves specificity of 42.89%. The highest sensitivity rate is achieved by CNN-2D classifier (86.70%), but LSTM gives a very close sensitivity rate (86.50%) and a higher specificity rate (77.89%) where CNN-2D achieves a specificity rate of 67.47% only.

As a comparison, in [22], the authors present their results as follows. The number of correctly detected activities are 95 for *adlnormal* and 76 for *adlerror*, both out of 100. Experiments in our study are performed on activity slices, on the other hand. in [22] they take whole activity and extract features from that activity and then try to decide if it is normal or abnormal. The problem here that is in real-life scenario, it cannot be known, where an activity starts and ends. Thus using slice-based detection is more meaningful.

LSTM is better to capture repetition related activities, while CNN is better to detect “confusion related activities”. CNN can detect changes in feature patterns. Even though it is not explicitly defined in daily activity datasets each activity is formed by steps. The steps in this dataset are based on the motion sensors triggered. For example, when the *sleeping* activity is considered, it is seen that the resident first goes out of bed, then goes to the middle of the room and then goes to the bathroom in the *bed to toilet activity.* CNN doesn't need to extract them, but it exploits them hierarchically in each layer. In the end, model cannot identify the steps involved, but it detects the anomaly in the higher level. Thus, whenever the orders of sensors or these steps change, input matrix changes which leads different feature maps extracted by CNNs.

Our method cannot detect anomalies such as incorrectly measuring the oatmeal, not using soap when cleaning, washing hands multiple times, confusing the location of items, and using too much soap or leaving kitchen utilities on. Because there were no specialised sensors for the items involved in these steps, the algorithm could not detect these errors and future research will be needed to deal with these errors. Moreover, our current approach may fail to detect abnormalities, when there is gradual deterioration regarding the health of an elderly person. This issue will be taken into consideration in future while collecting real-world data in which gradual deterioration can be observed. Moreover, it is planned to extract sub-activities involved in daily life activities and model their relations hierarchically. Then, this information can be used in order to provide more robust and accurate cognitive assessment tools.

5 Conclusion

This paper introduces a method of recognising sensor based activities and detecting anomalies related to dementia in smart homes. CNNs are exploited as well as their combination with LSTM in order to achieve these tasks. Our results on activity recognition shows that these methods are better than their competitors such as NB, HMMs, HSMMs and CRFs. Moreover, results on anomaly detection gives promising results to detect most of the abnormal behaviours simulating the daily life behaviour of elderly people suffering from dementia.

Acknowledgments

We thank to Tim Van Kasteren for providing the code for NB, HMM, HSMM and CRF. We also thank to Diana Cook and her team for making Aruba and WSU datasets publicly available.

References

1. S. Wild, G. Roglic, A. Green, R. Sicree, and H. King. Global prevalence of diabetes estimates for the year 2000 and projections for 2030. pages 1047–1053, 2004.
2. M. S. Albert and et al. The diagnosis of mild cognitive impairment due to Alzheimer's disease: Recommendations from the National Institute on Aging. *Alzheimer's & Dementia*, Volume 7:270–279, 2011.
3. W. Thies and L. Bleiler. 2013 Alzheimer's disease facts and figures. *Alzheimer's & dementia: the journal of the Alzheimer's Association vol. 9, no. 2*, pages 208–245, 2013.
4. M. R. Hodges, K. Kirsch, M. W. Newman, and M. E. Pollack. Automatic assessment of cognitive impairment through electronic observation of object usage. In *Pervasive*, pages 192–209, 2010.
5. K. Wild. Aging changes. In *Geraotechnology, Vol. 9 No 2*, pages 121–125, 2010.
6. P. Dawadi, D. Cook, and M. Schmitter-Edgecombe. Smart home-based longitudinal functional assessment. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, pages 1217–1224, 2014.
7. D. Riboni, C. Bettini, G. Civitarese, Z. H. Janjua, and R. Helaoui. Fine-grained recognition of abnormal behaviours for early detection of mild cognitive impairment. In *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 149–154, 2015.
8. D. Lara and M. A. Labrador. A mobile platform for real-time human activity recognition. In *2012 IEEE Consumer Communications and Networking Conference (CCNC)*, pages 667–671, 2012.
9. T. Kirste, A. Hoffmeyer, P. Koldrack, A. Bauer, S. Schubert, S. Schrder, and S. Teipel. Detecting the effect of Alzheimer's disease on everyday motion behaviour. *Journal of Alzheimer's Disease*, pages 121–132, 2014.
10. M. Ermes, J. Parkka, and L. Cluitmans. Advancing from offline to online activity recognition with wearable sensors. *30thAnnual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4451–4454, 2008.
11. T. Zhang, W. Fu, J. Ye, and M. Fischer. Learning movement patterns of the occupant in smart home environments: an unsupervised learning approach. *Journal of Ambient Intelligence and Humanized Computing*, 8(1):133–146, 2017.
12. Gilles Virone. Assessing everyday life behavioural rhythms for the older generation. *Pervasive Mob. Comput.*, 5(5):606–622, 2009.

13. K. S. Park J. H. Shin, B. Lee. Detection of abnormal living patterns for elderly living alone using support vector data description. *IEEE Transactions on Information Technology in Biomedicine 15(3):*, page 438448, 2011.
14. Abdur Rahim Mohammad Forkan, Ibrahim Khalil, Zahir Tari, Sebti Foufou, and Abdelaziz Bouras. A context-aware approach for long-term behavioural change detection and abnormality prediction in ambient assisted living. *Pattern Recognition*, 48(3):628 – 641, 2015.
15. Jens Lundstrm, Eric Jrpe, and Antanas Verikas. Detecting and exploring deviating behaviour of smart home residents. *Expert Systems with Applications*, 55:429–440, 2016.
16. D. J. Cook A. Crandall Adriana M. Seelye, M. Schmitter-Edgecombe. Naturalistic assessment of everyday activities and prompting technologies in mild cognitive impairment. *Journal International Neuropsychologly Soc. 2013 (4):*, pages 442–52, 2013.
17. N.K. Suryadevara, S.C. Mukhopadhyay, R. Wang, and R.K. Rayudu. Forecasting the behavior of an elderly using wireless sensors data in a smart home. *Engineering Applications of Artificial Intelligence*, 26(10):2641–2652, 2013.
18. Zaffar Haider Janjua, Daniele Riboni, and Claudio Bettini. Towards automatic induction of abnormal behavioral patterns for recognizing mild cognitive impairment. In *Proceedings of the* 31st *Annual ACM Symposium on Applied Computing*, pages 143–148, 2016.
19. N. Hammerla, J. Fisher, P. Andras, L. Rochester, R. Walker, and T. Plotz. PD disease state assessment in naturalistic environments using deep learning. pages 1742–1748, 2015.
20. K. S. Gayathri, Susan Elias, and Balaraman Ravindran. Hierarchical activity recognition for dementia care using markov logic network. *Personal and Ubiquitous Computing*, 19(2):271–285, 2015.
21. Fco Javier Ordóñez, Paula de Toledo, and Araceli Sanchis. Sensor-based bayesian detection of anomalous living patterns in a home setting. *Personal and Ubiquitous Computing*, 19(2):259–270, 2015.
22. Diana Cook and M. Schmitter-Edgecombe. Assessing the quality of activities in a smart environment. *Methods Inf. Med.*, 48:480–485, 2009.
23. Y. Tong, R. Chen, and J. Gao. Hidden state conditional random field for abnormal activity recognition in smart homes. *Entropy*, 17(3):1358, 2015.
24. P. N. Dawadi, D. J. Cook, and M. Schmitter-Edgecombe. Automated cognitive health assessment using smart home monitoring of complex tasks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 43(6):1302–1313, 2013.
25. Prafulla N. Dawadi, Diane J. Cook, and Maureen Schmitter-Edgecombe. Modeling patterns of activities using activity curves. *Pervasive Mob. Comput.*, 28(C):51–68, June 2016.
26. Oya Aran, Dairazalia Sanchez-Cortes, Minh-Tri Do, and Daniel Gatica-Perez. Anomaly detection in elderly daily behaviour in ambient sensing environments. *Human Behaviour Understanding:* 7th *International Workshop*, pages 51–67, 2016.
27. J. Yang, M. Nguyen, P. San, X. Li Li, and S. Krishnaswamy. Deep convolutional neural networks on multichannel time series for human activity recognition. pages 3995–4001, 2015.
28. M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In *6th International Conference on Mobile Computing, Applications and Services*, pages 197–205, 2014.
29. F. J. Ordonez and D. Roggen. Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016.
30. Nils Y. Hammerla, Shane Halloran, and Thomas Plötz. Deep, convolutional, and recurrent models for human activity recognition using wearables. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 1533–1540, 2016.
31. S. Ha and S. Choi. Convolutional neural networks for human activity recognition using multiple accelerometer and gyroscope sensors. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 381–388, 2016.

32. Charissa Ann Ronao and Sung-Bae Cho. Human activity recognition with smartphone sensors using deep learning neural networks. *Expert Systems with Applications*, 59:235 – 244, 2016.
33. Rui Yao, Guosheng Lin, Qinfeng Shi, and Damith Chinthana Ranasinghe. Efficient dense labeling of human activity sequences from wearables using fully convolutional networks. *CoRR*, abs/1702.06212, 2017.
34. A. Charissa and C. Sung-Bae. Evaluation of deep convolutional neural network architectures for human activity recognition with smartphone sensors. *In Proceedings of the KIISE Korea Computer Congress*, pages 858–860, 2015.
35. Nastaran Mohammadian Rad, Andrea Bizzego, Seyed Mostafa Kia, Giuseppe Jurman, Paola Venuti, and Cesare Furlanello. Convolutional neural network for stereotypical motor movement detection in autism. *CoRR*, abs/1511.01865, 2015.
36. Kai Wang, Youjin Zhao, Qingyu Xiong, and et al. Research on healthy anomaly detection model based on deep learning from multiple time-series physiological signals. *Scientific Programming*, 2016.
37. S. Choi, E. Kim, and S. Oh. Human behavior prediction for smart homes using deep learning. In *2013 IEEE RO-MAN*, pages 173–179, 2013.
38. T. Plötz, N. Hammerla, and P. Olivier. Feature learning for activity recognition in ubiquitous computing. volume 2, pages 1729–1734, 2011.
39. H. Fang and C. Hu. Recognizing human activity in smart home using deep learning algorithm. In *33rd Chinese Control Conference*, pages 4716–4720, 2014.
40. H. Fang, H. Si, and L. Chen. Recurrent neural network for human activity recognition in smart home. *In Proceedings of 2013 Chinese Intelligent Automation Conference*, pages 341–348, 2013.
41. Damla Arifoglu and Abdelhamid Bouchachia. Activity recognition and abnormal behaviour detection with recurrent neural networks. In 14th *International Conference on Mobile Systems and Pervasive Computing*, pages 86–93, 2017.
42. D. J. Cook, A. S. Crandall, B. L. Thomas, and N. C. Krishnan. CASAS: A smart home in a box. *Computer*, 46(7):62–69, 2013.
43. T. Van Kasteren, G. Englebienne, and B. J. A. Kröse. Human activity recognition from wireless sensor network data: Benchmark and software. *Activity Recognition in Pervasive Intelligent Environments*, pages 165–186, 2011.
44. J. Saives, C. Pianon, and G. Faraut. Activity discovery and detection of behavioral deviations of an inhabitant from binary sensors. *IEEE Transactions on Automation Science and Engineering*, 12(4):1211–1224, 2015.
45. M. Amiribesheli and A. Bouchachia. Smart homes design for people with dementia. In *2015 International Conference on Intelligent Environments*, pages 156–159, 2015.
46. C. François. Keras. https://github.com/fchollet/keras, 2015.
47. Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, may 2016.
48. Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.
49. G.G. Landis, J.R.; Koch. The measurement of observer agreement for categorical data. *Biometrics 33 (1)*, pages 159–174, 1977.

APPENDIX B

Abnormal Behaviour Detection for Dementia Sufferers via Transfer Learning and Recursive Auto-Encoders

Damla Arifoglu and Abdelhamid Bouchachia
Bournemouth University
Fern Barrow, Poole BH12 5BB
Dorset, UK
{darifoglu,abouchachia}@bournemouth.ac.uk

***Abstract*—Cognitive impairment is one of the crucial problems elderly people face. Tracking their daily life activities and detecting early indicators of cognitive decline would be necessary for further diagnosis. Depending on the decline magnitude, monitoring may need to be done over long periods of time to detect abnormal behaviour. In the absence of training data, it would be helpful to learn the normal behaviour and daily life patterns of a (cognitively) healthy person and use them as a basis for tracking other patients. In this paper, we propose to investigate Recursive Auto-Encoders (RAE)-based transfer learning to cope with the problem of scarcity of data in the context of abnormal behaviour detection. We present a method for generating synthetic data to reflect on some behavior of people with dementia. An RAE model is trained on data of a healthy person in a source household. Then, the resulting RAE is used to detect abnormal behavior in a target house. To evaluate the proposed approach, we compare the results with the-state-of-the-art supervised methods. The results indicate that transfer learning is promising when there is lack of training data.**

***Index Terms*—Dementia, Transfer Learning, Recursive Auto-encoders, Abnormal Behaviour Detection**

I. Introduction

Studies indicate that by year 2030, the number of people aged 65 to 74 will be about 3% of the total population [1]. Elderly people may suffer from the consequences of cognitive decline, which is a condition that causes problems with physical and mental abilities such as memory and thinking [2]. An elderly person having such cognitive decline requires care and support from caregivers. A continuous monitoring of the daily routine of the elderly can be helpful for clinicians to diagnose the early onset of cognitive decline. The best markers of cognitive decline may not necessarily be detected based on a person's performance at any single point in time, but rather by monitoring the trends over time [3]. Most common types of dementia (Alzheimer) can be identified by behavioural changes like sleep disturbances and inability to complete tasks. For instance, an old person suffering from Alzheimer may forget to have his lunch, take multiple lunches instead, wake up in the middle of the night, go to the toilet frequently. In particular, the daily home activity involving basic functions like preparing food, showering, sleeping, etc. can be used to assess the well-being of elderly people. Thus, it is beneficial to track elderly people's life over time in a smart home to detect the indicators of dementia at an early stage.

The development of ambient home assessment environments has begun to provide the opportunity to assess behaviour change unobtrusively in real-time. Although a few promising methods have been experimentally validated [3]–[5], the translation of the current knowledge into smart homes still requires more dedication and work. Current assessment methods mostly rely on queries from questionnaires or in-person examinations that may poorly represent a person's cognitive status. Also the clinical methods have some limitations such as their episodic nature, and possible biased reporting. The main motivation for our work is that cognitive decline can be observed in daily activities of an elderly person. Real-time monitoring of activities performed by an elderly person in a smart home would be beneficial for early detection of such decline.

Machine learning based cognitive status assessment studies rely on activity recognition techniques. These methods first learn what is normal from training data and then flag the abnormal activity based on classification confidence values [6]–[10]. They require training data to be manually annotated, which is extremely hard and a time-consuming task. Moreover, these techniques assume that the training data is available prior to the training phase. However, we cannot expect elderly people to annotate the necessary training data. Thus, tackling the activity recognition and abnormal behaviour detection as an unsupervised process would be helpful. Moreover, this monitoring may need to be done over long periods of time. But collecting sequential data of months or years with time dependency is highly time-consuming. Thus, using an existing data from a source household to learn what is normal, and then transferring this knowledge to a target house would be beneficial.

Daily life activities are often composed of several steps [11]. For example; the activity *wash clothes* implies the following actions: *get clothes from basket, fill up washing machine, turn on washing machine*. The anomalies related to dementia may be reflected in the repetition frequency of these steps and their relation with each other. The elderly people with dementia tend to confuse things and repeat or skip some steps during the completion of a specific activity. Building activities from their granular units hierarchically would be helpful to understand the internal dynamics of the activities. Hence, the problem of activity recognition can be viewed as a hierarchical learning problem which resembles scene parsing or phrase detection [12]. In this paper, we aim to construct activity instances hierarchically from their low-level units to detect abnormal ones reflecting on indicators of dementia. The hierarchical representation of RAE provides an abstraction of activities in a house and then mapping these abstract levels to another house via transfer learning will be useful. Abstraction provides a generalisation of the hierarchical level of information between the houses and it reduces the differences between them, making transfer learning an appealing approach. In this paper, we use unlabelled data collected for normal activities from a source house to train the RAE model. Then, we transfer this model to a target house to detect abnormal behavior related to dementia. In a nutshell, the present paper introduces two contributions. Firstly, a method is proposed to generate synthetic data that simulates the abnormal behavior of people with dementia. Secondly, RAEs are exploited to model activities based on their low-level structures and detect abnormal behavior related to dementia. The rest of the paper is organised as follows. Section II provides an overview of the related literature. Section III presents the details of the proposed methodology together. Section IV describes the experimental set-up and results of the experiments followed by a discussion. Finally, Section V concludes the paper.

II. Related Work

In-home automatic assessment of cognitive decline has been the subject of many machine learning approaches such as Support Vector Machines (SVMs) and Naïve Bayes (NB) [6], Restricted Boltzmann Machines (RBMs) [7], Hidden Markov Models (HMMs) [8], Random Forests [9] and Recurrent Neural Networks (RNNs) [10]. In [10], the authors exploit RNNs to detect abnormal behavior of dementia sufferers in a daily living scenario. The abnormal behavior are flagged based on their classification confidence values. Unsupervised methods such as auto-encoders are also being used for anomaly detection in time-series literature [13], [14]. Many studies [4], [7] have relied on rule-based systems to assess the cognitive status of elderly people. In [15], the assessment is done by asking elderly people to complete a sequence of scripted actions. The participants are monitored via Web camera while they perform tasks and they receive scores by trained experts. In [4], the authors detect anomalies by exploiting Markov Logic network. They use a hybrid technique including supervised learning, rule-based reasoning and probabilistic reasoning. These studies fail to provide an unobtrusive way of assessment since they are not done in the natural flow of daily living and in real life scenarios. Specifically, we aim in this study to detect anomalies without giving any instruction and considering not only some time interval, but everyday living scenario. Moreover, in rule-based systems, an expert is needed to manually integrate specific rules to the system. The proposed approach does not require any expert knowledge.

In transfer learning literature, most of the activity recognition models are supervised models that require labelled data to learn the model parameters [16]–[18]. Good results are obtained using generative models such as HMM [16], [18] and discriminative models such as CRF [17], [18]. In [19], a method is proposed to learn the parameters of a HMM using labelled data from the source domain, and unlabelled data from the target domain. The study ignores the activities' important features such as the activity structure and related temporal features. They also assume that the structure of HMMs is given and pre-defined. Later they extend this work to learn hyperparameter priors for HMM instead of learning the parameters directly [20].

In transfer learning, sensors and activities in different households are needed to be mapped. In [19], a comparison of feature mappings was done. The mapping that combined sensor readings in a single feature based on their function (e.g. sensors used during cooking) gave the best results. In some cases, meta-features are first manually introduced into the feature space and then the feature space is automatically mapped from the source domain to the target domain [20]. In [21], the authors first assign a location label to each sensor indicating in which room or functional area the sensor is located. Then activity templates are constructed from the data for both the source and target data, finally a mapping is learnt between the source and target datasets based upon the similarity of activities and sensors.

III. Proposed Work

The proposed work consists of the following steps: 1) Time-slice chunks are extracted from sequential sensor reading data using a sliding window. 2) *Last-fired* features are extracted from time-slices as in [22]. 3) RAE is trained on a source household dataset to learn the parameters for *normal* behavior. 4) These parameters are then transferred into a target household to detect abnormal behavior.

A. Dataset

The evaluation of the proposed method is done on households A and C of Kasteren datasets [20]. We chose these two households since they span more days (25 days and 18 days respectively). The activities performed in household A and C are used to reflect normal behavior. However, some of the data in household C is modified (Section III-B) to generate samples representing abnormal behaviour of dementia sufferers. Here, household A will be used as the source house while household C will be used as the target house.

B. Synthesis of Abnormal Behaviour

Given the scarcity of data reflecting abnormal behavior of dementia sufferers, we need to synthesise some activities that can be observed in daily-life routines of elderly people with dementia. We focus on the generation of two kinds of anomalies: 1) Repeating activities and 2) Disruption in sleep

1) Repeating activities: Elderly people with dementia may forget whether they performed a particular activity or not, so they may repeat that activity (having multiple lunches, e.g.) [23]. To reflect on this scenario, we generate synthetic abnormal activities by manually inserting a specific set of actions within the normal activity sequence. This will result in multiple occurrences of that activity, which will occur in some inadequate time of the day such as having dinner in the middle of the night.

2) Disruption in sleep: Degeneration of the sleep-waking cycle and night time wandering are among the most severe behavioural symptoms of dementia. For example, elderly people may wake up many times in the night to use the toilet and go back to sleep [23], [24]. We simulate these anomalies by inserting some synthetic activities in the normal night-time activity sequences.

For example; assume that S is a sequence of activities occurring in a day such as $S = d_1, \ldots, d_x, e_1, \ldots, e_t, d_{x+1}, \ldots, d_n$ where each d_i is a time-slice of some activity and each e_j is a time-slice of *eating* activity. Then, time-slices of e are injected into the sequence S to have the abnormal version. Then S becomes $d_1, \ldots, d_m, e_1, \ldots, e_t, d_{m+1}, \ldots, d_a, e_1, \ldots, e_k, d_{a+1}, \ldots, d_n$. Many instances of *getting drink*, *taking shower*, *use toilet* activities are injected. In result, 162 abnormal time-slices are synthesised in dataset C.

C. Feature Extraction and Sensor Mapping

After the synthesis, the datasets are processed in the following way. Firstly, 1 minute slices are extracted from datasets using a sliding window [22]. Then time-slices are mapped into *last-fired* feature representation [20]. *Last-fired* feature [20] indicates which sensor is fired last. The sensor that changed state last continues to give 1 and changes to 0 when another sensor changes state. The last representation gives an indication of the location of an inhabitant. As people start moving, the sensors are triggered based on the location of the movement, which provides an update of their current location [22]. The updates, in the form of a time-series data, provide fine-grained patterns about the activity performed. Such patterns are hierarchical and they follow grouping rules at multiple levels of abstraction. Findings in [25] support our approach. The authors extract location-based patterns in daily-life routines. Hence, we employ RAE as a hierarchical model to organise the steps in an activity and record their relative ordering. We exploit *last-fired* feature to model location based granular level information, since such feature allows for capturing execution details of the activities. Next, RAEs will be exploited to model all this low-level information in a hierarchical representation.

There are 14 sensors in dataset A and 21 sensors in dataset C. We map these sensors to each other by using meta features as described in [20]. We use the mapping that combined sensor readings in a single feature based on their function (e.g. sensors used during cooking). In [19], different mapping strategies, such as union, intersection and duplicate are investigated. We use *union* mapping since it gave the best results. Using *union* mapping for each function group, the union of all the sensors in the group is taken, resulting in one sensor output per group per house. For example, the front and back door in the target house are combined into a single sensor and matched with the front door sensor in the source house. This results in 7 sensor groups, which will be treated as features. Moreover, the activities in two datasets are mapped and 9 similar activities are used [20].

D. Recursive Auto-Encoders

Auto-encoders are unsupervised artificial neural networks that compress the input into a representation and then reconstructs the output from it. They are self-supervised because they use the input instances as labels and use training data to learn the parameters for the model. An auto-encoder has 3 parts: an encoding function, a decoding function, and a loss function. The encoder compresses the input and produces a representation, the decoder then reconstructs the input from this representation. Loss function calculates the error between the actual input and the reconstructed input.

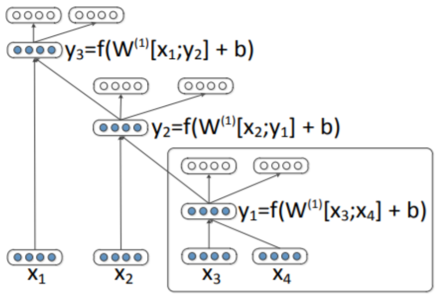


Fig. 1. A recursive auto-encoder

As depicted in Figure 1 (retrieved from [26]), RAEs merge each instance with its next neighbour to construct the parent node [27]. For example, the children x_3 and x_4 are merged and the parent y_1 is constructed by an encoding function f. Then the parent y_1 is merged with another child x_2 and this goes on in the upper layers in a bottom-up fashion. The process yields a hierarchical organisational structure for children. The final code representing the entire tree is decoded to recover the children and the entire hierarchy by the following inverse process. The first parent vector y_1 is computed from the children x_3, x_4, so that $y_1 = f(W^{(1)}[x_3; x_4] + b^{(1)})$ where a matrix of parameters weights W is multiplied with the children vectors. After adding a bias term, an element wise activation function such as $tanh$ is applied to the resulting vector. To see how well this function is doing, the model reconstructs the children in a reconstruction layer: $[x_3; x_4] = g(W^{(2)}y_1 + b^{(2)})$. At each level of the tree, the same encoding and decoding function is used recursively. During training, the goal is to minimise the reconstruction errors of the input pairs. For each pair, the Euclidean distance between the original input and its reconstruction is calculated: $E = P_{rec}([x_3; x_4])$. The process repeats until the full tree is constructed and a reconstruction error is obtained at each non-terminal node. The encoding and decoding weight parameters are learnt by using the training set and applying back-propagation through structure [28] to update the network weights.

E. Abnormal Behaviour Detection

First, house A is used to learn the parameters ($W^{(1)}$, $b^{(1)}$) for encoding function and ($W^{(2)}$, $b^{(2)}$) for decoding function of RAE. These parameters are then used to construct RAE trees to test instances of house C. In each level of the tree, two children are merged to form a feature vector as their parent, which encodes the information coming from the children. Thus, the feature vector at the root node summarises all the information coming from the children in the tree and their hierarchical orderings are learnt by RAE. The feature vectors at parent nodes can be decomposed into their granular level, hierarchical pieces by using the decoding weights. Anomalies are defined as samples that are deviations from the expected behaviour. Any deviations from this normal can be identified by measuring the reconstruction error E described above. When a new activity is introduced as a test instance, if it is a normal activity, the reconstruction error will be smaller since anomalies that represent any deviations will be poorly reconstructed. If it is an abnormal behaviour, which is not seen in the source household, RAE will reconstruct that instance with a higher error. We exploit these errors to decide if that activity is normal or abnormal based on a threshold.

The proposed method is compared with the following state-of-the-art supervised methods; Long Short Term (LSTM) RNNs, NB, HMM and CRF. First, using the training instances and their corresponding labels in dataset A, these models are trained. Then the instances in dataset C are given to the trained classifiers. The models assign a class label to each instance with a confidence value. When a new test instance in house C is introduced, if the model assigns it to a class with a confidence value which is bigger than a threshold, that instance is considered as a normal activity, otherwise it is flagged as an abnormal activity.

IV. Experiments

The experiments with NB, HMM and CRF are performed based on the implementation [19], while LSTM and RAE experiments are performed on Python and Keras. In RAE trees, each child represents 1 minute time-slice of a feature vector (size of 1×7). RAE trees are constructed with a time-step of 5 (chosen experimentally), where 5 time-slices are merged in a RAE tree. To run experiments on LSTM, we used drop-out with a value of 0.5. We also set the batch size to 10 and the epoch to 500 iterations. The internal architecture of LSTM (2 hidden layers consisting of 30 and 50 nodes respectively) and time-step of the sequences (25 activity slices) were empirically set. In supervised models, the activities in house C are evaluated based on the model learnt from the house A.

To assess the abnormal behaviour detection success, True Positive Rate (TPR) and False Positive Rate (FPR) are used. These values for different thresholds are showed on a Receiver Operating Characteristic (ROC) curve. Moreover, Area Under Curve (AUC) is calculated for each model to interpret the results in a better way. TPR refers to the method's ability to correctly detect instances which are abnormal. FPR gives the percentage of mislabelled normal instances, thus reflects the method's ability to differentiate between normal and abnormal.

The performance of the supervised methods to classify the activity instances is measured by precision, recall, F-measure and accuracy. Final precision and recall values are calculated by taking average over classes. Precision and recall give better idea about the performance on imbalanced datasets like the ones in this study. On the other hand, the accuracy represents the percentage of correctly classified time-slices, therefore more frequently occurring classes have a larger weight in this measure.

A. Results

The first experiment is conducted to evaluate the classification performance of the supervised methods when using transfer learning. These methods are trained on house A and then tested on house C. Activity recognition accuracy rates are depicted in Table I. These results are very close to activity recognition rates with leave-one-out cross validation presented on the same datasets in [22], where one day of the dataset C is used as testing set, while the remaining days are used training set. However, our results are obtained via transfer learning, where the training is done on dataset A and the testing is done dataset C. In [22], the leave-one-out classification accuracy with NB, HMM and CRF are given as 87.0%, 83.9% and 89.7% as respectively. In our case, the classification accuracy rates are 87.47%, 84.88%, 84.55% and 87.02% with NB, HMM, CRF and LSTM respectively. The similar results show that applying transfer learning is successful to recognise activities. Unfortunately, we cannot test the classification accuracy of RAE model since it is unsupervised.

Moreover, the results in Table I show that the highest accuracy is achieved by NB since NB favours the most frequent class. Analysing precision, recall and F-measures (36.71%, 33.37% and 34.96% respectively), we see that class-based success is not high. However, for the methods which take temporal information into consideration, such as LSTM, HMM and CRF, these metrics are better. The highest precision is achieved by LSTM (48.58%) while the highest recall is achieved by HMM (44.18%). The highest F-measure is achieved by LSTM (42.95%).

The ability of RAE to reconstruct the features is evaluated by k-means clustering. After clustering the reconstructed features into 9 clusters, the dimensions of the features are reduced to 2D by Principal Component Analysis (PCA). As depicted in Figure 2, RAE is successful to reconstruct the features from different classes.

TABLE I
ACTIVITY RECOGNITION RESULTS

Model	Precision	Recall	F-Measure	Accuracy
NB	36.71%	33.37%	34.96%	87.47%
HMM	37.32%	44.18%	40.46%	84.88%
CRF	42.80%	37.81%	40.15%	84.55%
LSTM	48.58%	38.49%	42.95%	87.02%

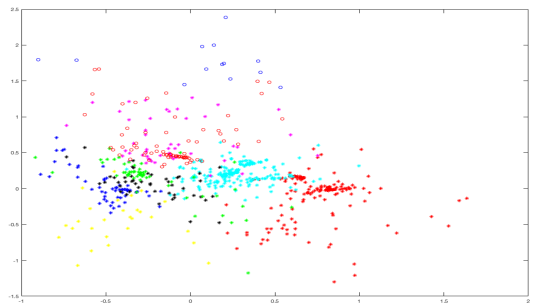


Fig. 2. Clustering of RAE re-constructed features. Each colour indicates a different class, while X and Y coordinates are 2D features (2 principal components).

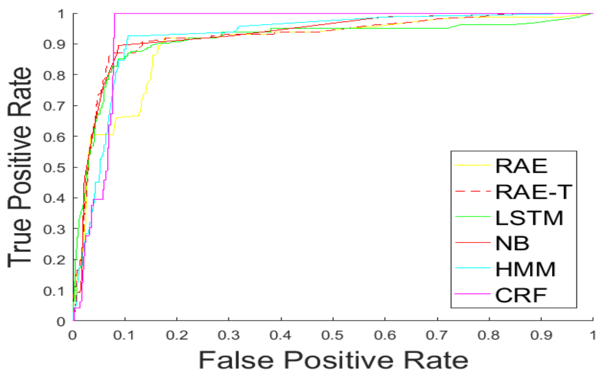


Fig. 3. ROC curves for abnormal behaviour detection.

In the second experiment, the methods are compared in terms of abnormal behaviour detection. The results are depicted as ROC in Figure 3. AUC values for methods CRF, HMM, LSTM, NB, RAE and RAE-T are 94.72%, 92.25%, 91.21%, 92.70%, 89.74%, 92.50% respectively. The results show that RAE based abnormal behaviour detection is competitive with the supervised methods. All methods are good at detecting the abnormal behaviour instances and pruning the false alarms. The proposed RAE based method produces slightly worse TPR and FPR. However its superiority comes from the fact that it doesn't use any labels during the parameter learning process.

Moreover, some of the data of target household is used to re-train the learnt RAE model. In this way, we can tailor the RAE model for the resident by re-tuning the parameters of previously trained RAE using user-specific training examples. This strategy is similar to inductive transfer learning or self-taught learning [29] when none or few data labels are available in the target domain. However, in our case, which is unsupervised, we use only some of the unlabelled data coming from the target dataset. Although we need some data from the target house, this still allows us to reduce several weeks or months of data collection and annotation in the target space to only a few days. For this purpose, RAE learnt on instances of source household dataset is re-trained over 10-days data from house C. The results are shown in the ROC curve (Figure 3 with the abbreviation RAE-T). These results indicate that re-tuning the parameters and considering the house specific behaviour improve the results.

Moreover, we calculate Cohen's Kappa statistics to show the robustness of RAE to detect abnormal behavior. Kappa statistics is a measure that handles both multi-class and imbalanced class problems. It tells how good the classifier is performing over the performance of a classifier that simply guesses at random according to the frequency of each class [30]. However, we use weighted Kappa statistics, since detecting abnormal behaviour is more important than pruning normal ones in our case. In health-care problems, missing a true positive may cause more serious problems than retrieving a high number of false positives. Thus, we assign a higher weight to true positive than false positive in the weight matrix of Kappa. The calculated Kappa for RAE is 0.53, which is a moderate agreement according to [30].

Supervised models, especially deep learning methods such as LSTMs, require too much training data. Moreover, providing labelled data just once would not be enough since observation of dementia sufferers is a task which can be up to years. Also, these models need activity classes to be fixed.However, in a time lapse of years, users may change their behavioural patterns and they may introduce new activity labels. On the other hand, when transfer learning is used, just changing the mapping of sensors and activities would be enough to adapt the model to the new data. Moreover, supervised methods such as NB, HMM, CRF, LSTM doesn't encode time-slices in a hierarchical representation. RAE encodes hierarchy via merging in a bottom-up tree structure. The use of hierarchical models might be a better fit for transfer learning because the different levels of the hierarchy allow a better abstraction between houses.

Although we re-train the learnt RAE model on partial data steming from the source house, when there is no source data available prior, domain adaptation would not be possible. Then, there will be a problem to detect resident specific abnormal behavior. Transfer learning generalises the behavior of inhabitants between different houses and does not take resident specific behavior into account. For example, going to toilet during sleep might be normal for a person, while it is abnormal for another person. A prior distribution can be learnt from the source house and used to provide a sensible initial value for the model parameters of the target house. The behaviour across different houses are transferable under the condition that the resident profiles such as age, gender and lifestyle are similar in both households. Also the different sensors and activities in these houses should be mapped into each other.

The proposed system would improve life experience of dementia sufferers in the following way. The system detects possible candidates for abnormal behaviour to inform the caregiver or the medical doctor. The decision maker will analyse the detected abnormal behavior by considering the person's personal life style. Thus, the proposed method can be used as a decision supporting system rather than a decision making system. Detecting high amount of false positives will not introduce any risk related to the health of the person. Detecting true positives in an early stage would trigger further analysis and would be helpful for an early treatment. The important advantage of the proposed system would be to provide a cognitive status assessment in the natural flow of daily living without annoying elderly people.

V. Conclusion

In this paper, we proposed an RAE-based method to detect abnormal behavior of elderly people with dementia. Transfer learning can be an interesting option to cope with scarcity of data. The empirical results showed that the proposed method is promising when supervised methods cannot be exploited because of the lack of (labelled) training data. However, the proposed method failed to detect the person's specific abnormal behaviour. In future, we will consider personal habits by learning a prior distribution from the source house to adapt the model to the target house.

References

[1] S. Wild, G. Roglic, A. Green, R. Sicree, and H. King, "Global prevalence of diabetes estimates for the year 2000 and projections for 2030," *Diabetes Care, vol. 27, no. 5*, pp. 1047–1053, 2004.

[2] M. S. Albert and et al., "The diagnosis of mild cognitive impairment due to Alzheimer's disease: Recommendations from the National Institute on Aging," *Alzheimer's & Dementia*, vol. Volume 7, pp. 270–279, 2011.

[3] K. Wild, "Aging changes," in *Geraotechnology, Vol. 9 No 2*, 2010, pp. 121–125.

[4] D. Riboni, C. Bettini, G. Civitarese, Z. H. Janjua, and R. Helaoui, "Fine-grained recognition of abnormal behaviors for early detection of mild cognitive impairment," in *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2015, pp. 149–154.

[5] T. Kirste, A. Hoffmeyer, P. Koldrack, A. Bauer, S. Schubert, S. Schrder, and S. Teipel, "Detecting the effect of Alzheimer's disease on everyday motion behaviour," *Journal of Alzheimer's Disease*, pp. 121–132, 2014.

[6] F. J. Ordóñez, P. de Toledo, and A. Sanchis, "Sensor-based bayesian detection of anomalous living patterns in a home setting," *Personal and Ubiquitous Computing*, vol. 19, no. 2, pp. 259–270, 2015.

[7] N. Hammerla, J. Fisher, P. Andras, L. Rochester, R. Walker, and T. Plotz, "PD disease state assessment in naturalistic environments using deep learning," 2015, pp. 1742–1748.

[8] A. R. M. Forkan, I. Khalil, Z. Tari, S. Foufou, and A. Bouras, "A context-aware approach for long-term behavioural change detection and abnormality prediction in ambient assisted living," *Pattern Recognition*, vol. 48, no. 3, pp. 628 – 641, 2015.

[9] J. Lundstrm, E. Jrpe, and A. Verikas, "Detecting and exploring deviating behaviour of smart home residents," *Expert Systems with Applications*, vol. 55, pp. 429–440, 2016.

[11] U. Naeem, R. Bashroush, R. Anthony, M. Azam, A. Tawil, S. Lee, and M. Wong, "Activities of daily life recognition using process representation modelling to support intention analysis," *International Journal of Pervasive Computing and Communications*, vol. 11, no. 3, pp. 347–371, 2015.

[10] D. Arifoglu and A. Bouchachia, "Activity recognition and abnormal behaviour detection with recurrent neural networks," in 14^{th} *International Conference on Mobile Systems and Pervasive Computing*, 2017, pp. 86–93.

[12] R. Socher, C. D. Manning, and A. Y. Ng, "Learning continuous phrase representations and syntactic parsing with recursive neural networks," in *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, 2010, pp. 1–9.

[13] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proceedings of the* 23^{rd} *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '17, 2017, pp. 665–674.

[14] B. Kiran, D. Mathew Thomas, and R. Parakkal, "An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos," vol. 4, 01 2018.

[15] C. D. J. C. A. Adriana M. Seelye, Schmitter-Edgecombe M., "Naturalistic assessment of everyday activities and prompting technologies in mild cognitive impairment," *Journal International Neuropsychologly Soc. 2013 (4):*, pp. 442–52, 2013.

[16] D. J. Patterson, D. Fox, H. Kautz, and M. Philipose, "Fine-grained activity recognition by aggregating abstract object usage," in *Ninth IEEE International Symposium on Wearable Computers (ISWC'05)*, 2005, pp. 44–51.

[17] W. L. H.L Chieu and L. Kaelbling, "Activity recognition from physiological data using conditional random fields." In SMA Symposium. Singapore-MIT Alliance, 2006.

[18] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, "Accurate activity recognition in a home setting," in *Proceedings of the 10th International Conference on Ubiquitous Computing*, ser. UbiComp '08, 2008, pp. 1–9.

[19] T. Van Kasteren, G. Englebienne, and B. Krose, "Recognizing activities in multiple contexts using transfer learning," *Proceedings of the AAAI Fall Symposium on AI in Eldercare: New Solutions to Old Problems.*, 2008.

[20] T. L. M. van Kasteren, G. Englebienne, and B. J. A. Kröse, "Transferring knowledge of activity recognition across sensor networks," in *Proceedings of the 8th International Conference on Pervasive Computing*, ser. Pervasive'10, 2010, pp. 283–300.

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

[22] T. Van Kasteren, G. Englebienne, and B. J. A. Kröse, "Human activity recognition from wireless sensor network data: Benchmark and software," *Activity Recognition in Pervasive Intelligent Environments*, pp. 165–186, 2011.

[23] J. Saives, C. Pianon, and G. Faraut, "Activity discovery and detection of behavioral deviations of an inhabitant from binary sensors," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 4, pp. 1211–1224, 2015.

[24] M. Amiribesheli and A. Bouchachia, "Smart homes design for people with dementia," in *2015 International Conference on Intelligent Environments*, 2015, pp. 156–159.

[25] T. Zhang, W. Fu, J. Ye, and M. Fischer, "Learning movement patterns of the occupant in smart home environments: an unsupervised learning approach," *Journal of Ambient Intelligence and Humanized Computing*, vol. 8, no. 1, pp. 133–146, 2017.

[26] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning, "Semi-supervised recursive autoencoders for predicting sentiment distributions," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '11, 2011, pp. 151–161.

[27] J. B. Pollack, "Recursive distributed representations," *Artif. Intell.*, vol. 46, no. 1-2, pp. 77–105, Nov. 1990.

[28] C. Goller and A. Kuchler, "Learning task-dependent distributed representations by backpropagation through structure," in *Proceedings of International Conference on Neural Networks (ICNN'96)*, vol. 1, 1996, pp. 347–352 vol.1.

[29] S. J. Pan and Q. Yang, "A survey on transfer learning, technical report HKUST-CS08-08, department of computer science and engineering, Hong kong university of science and technology, China."

[30] G. Landis, J.R.; Koch, "The measurement of observer agreement for categorical data," *Biometrics 33 (1)*, pp. 159–174, 1977.

APPENDIX C

To appear in proceedings of the 14th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2017)

Activity Recognition and Abnormal Behaviour Detection with Recurrent Neural Networks

Damla Arifoglu and Abdelhamid Bouchachia
Department of Computing and Informatics, Bournemouth University, UK

September 20, 2017

Abstract

In this paper, we study the problem of activity recognition and abnormal behaviour detection for elderly people with dementia. Very few studies have attempted to address this problem presumably because of the lack of experimental data in the context of dementia care. In particular, the paper investigates three variants of Recurrent Neural Networks (RNNs): Vanilla RNNs (VRNN), Long Short Term RNNs (LSTM) and Gated Recurrent Unit RNNs (GRU). Here activity recognition is considered as a sequence labelling problem, while abnormal behaviour is flagged based on the deviation from normal patterns. To provide an adequate discussion of the performance of RNNs in this context, we compare them against the state-of-art methods such as Support Vector Machines (SVMs), Naïve Bayes (NB), Hidden Markov Models (HMMs), Hidden Semi-Markov Models (HSMM) and Conditional Random Fields (CRFs). The results obtained indicate that RNNs are competitive with those state-of-art methods. Moreover, the paper presents a methodology for generating synthetic data reflecting on some behaviours of people with dementia given the difficulty of obtaining real-world data.

1 Introduction

Studies indicate that by year 2030, 19% of people will be aged 74 to 84 and nearly half of people who are older than 84 will have dementia [27]. Elderly people may suffer from the consequences of dementia, which is a condition that causes problems with mobility, physical and mental abilities such as memory and thinking [8]. It also may cause decrease in the ability of speaking, writing, distinguishing objects, performing motor activities and performing complex functional tasks (paying bills, preparing a meal, shopping, managing medication, etc.) [26]. An elderly person having such cognitive decline loses independence in daily life and requires care and support from caregivers.

Cognitive diseases like dementia need to be detected at an early stage so that early treatment will be possible. However, research shows that 75% of dementia and early dementia cases go unnoticed [15] and many such cases are only diagnosed when such impairment reaches moderate or advanced stage. The detection of early signs of motion and cognitive impairment (MCI) via activity recognition will be useful to track motion and cognitive capabilities of the elderly, thus improving their life quality and financial saving. Unfortunately, currently there are no dementia friendly smart homes addressing these people's special needs.

Most common types of dementia (Alzheimer, Parkinsons disease) can be identified by behavioural changes like sleep disturbances, difficulty of walking and inability to complete tasks. Such changes can provide key information about memory, mobility and cognition of a person. For instance, an inhabitant suffering from Alzheimer may forget his lunch, take multiple lunches instead, wake up in the middle of the night, go to the toilet frequently, or have dehydration problems because of forgetting to drink daily amount of water.

Recent studies suggest that changes in complex daily life tasks can be indicators of early decline [29]. The best markers of cognitive decline may not necessarily be detected based on a person's performance at any single point in time, but rather by monitoring the trend over time and the variability of change in a duration [29]. Thus, tracking an elderly person's life over time in a specially designed smart home, doing in-home health assessment and detecting the indicators of dementia at an early step would be beneficial.

The identification of early onsets of dementia using non-medical diagnosis methods requires the development of new diagnostic tools. Although a few promising methods have been experimentally validated [6, 23, 17, 16, 7], the translation of the current knowledge into smart homes still requires more dedication and work. Current assessment methods mostly rely on queries from questionnaires or in-person examinations, which depend on recall of events or brief snap-shots of function that may poorly represent a person's typical state of function. Moreover, these studies include some pre-defined tasks given to the patients in order to do automatic assessment of cognitive decline by trained experts.

The main motivation for our work is that cognitive decline can be observed in daily activities and routines of an elderly. Real-time monitoring of activities performed by elderly in a smart home would be beneficial for the early detection of such decline. In this study, we firstly recognise activities by variants of RNNs, namely VRNNs, LSTMs and GRUs and model the daily behaviour routines of a person. Whenever a new sequence is introduced, any abnormality deviating from these regular behaviours are detected and could be used for further investigation by formal or informal carer.

Unfortunately, there exists no publicly available dataset on abnormal behaviour of people with dementia. Producing such a dataset require time and adequate experimental environment. Thus we propose in this paper, a way to artificially produce data on abnormal activities reflecting on typical behaviour of elderly people with dementia. We believe that this an important contribution.

The rest of the paper is organised as follows. Section 2 provides a brief overview of the related research to both activity recognition and abnormal behaviour detection. Section 3 presents the details of the proposed methodology together with the datasets and models used. Section 4 describes the experimental set-up and results of the experiments followed by a discussion. Finally, Section 5 concludes the paper.

2 Literature Review

Activity recognition has been addressed using methods such as decision trees, Bayesian methods (Naïve Bayes and Bayesian Networks), k-Nearest Neighbours, Neural Networks (Multilayer perceptron), SVMs, Fuzzy logic, Regression models, Markov models (Hidden Markov Models, Conditional Random Fields) and classifier ensembles (Boosting and bagging) [18]. Recently, there has been growing interest in deep convolutional neural networks [30, 31, 20, 13], Deep Belief Networks [4], Restricked Boltzman Machines (RBMs) [21, 12, 4, 9] and RNNs [20, 13, 10]. Previous work shows that RNNs are useful, but leaves a lot of room for improvement. It is worthwhile to stress that to the best of our knowledge, this study is the first applying RNNs to detect abnormalities related to dementia in the daily life routines of an elderly person.

In [21], RBMs are used for feature extraction and selection from sequential data. In [20], the authors use a combination of deep convolutional networks and LSTM to do multi-modal wearable activity recognition by showing that their approach outperforms some of the previously reported results by up to 9% on OPPORTUNITY dataset. In [1], the authors utilised convolutional networks to classify activities using time-series data collected from smart phone sensors. Experiments show that increasing the number of convolutional layers increases the performance, but the complexity of the derived features decreases with every additional layer. In [13], the authors explore deep, convolutional and recurrent approaches across three representative datasets that contain movement data captured with wearable sensors. Moreover, they describe how to train recurrent approaches in this setting and introduce a novel regularisation approach, showing better results over OPPORTUNITY, PAMAP2 and Daphnet Gait datasets. In [9], results with RBM on CASAS dataset outperformed HMM and Naïve Bayes Classifier (NBC) in most of the cases. In [19], the authors use RNNs to predict the future values (start time, duration) of the activities.

Most of the aforementioned studies use movement data such as OPPORTUNITY, SKODA [21, 30, 31, 20] or UCI HAR smart phone dataset, MIT home dataset [1, 4], which are obtained through body worn sensors. Except the work by Fang et al. [9, 10], none of these studies focus on daily activity datasets collected by sensors placed at home. In this work, we investigate RNNs on daily activities data obtained by van Kasteren [28] using various environment sensors (see Sec. 3.1 for more details).

In-home automatic assessment of cognitive decline has been the subject of some studies dedicated [5, 6, 22, 12]. For instance, in [5], machine learning approaches such as SVMs and Naïve Bayes are used. In [12], Parkinson's Disease state assessment in home is explored by means of RBMs using data from body worn sensors. In [22], the authors use Markov Logic Network, which is a probabilistic logic that unifies statistical and symbolic reasoning to detect anomalies. In [5], some instructions to perform some tasks (e.g., sweeping the kitchen, dusting the floor, etc.) are given to the patients who then receive scores after completing those tasks. These scores are calculated based on the time spent, the frequency of the sensor triggered, etc. One disadvantage of this scenario is that some pre-selected activities are performed and instructions are given to the elderly who might not be able to cope with such tasks at all. Moreover, using rule-based systems, an expert is needed to manually integrate resident-specific rules to the system since every person has her/his own daily life routines. For example waking up and drinking water in the middle of the night might be normal for a person, while abnormal for some other person. However, our approach does not require any expert knowledge, since it learns what is normal and abnormal from the training data automatically. Specifically, we aim in this study to detect anomalies in the natural flow of daily living without giving any instruction and considering not only some time interval, but everyday living scenario. Continuous assessment of the person is more valid, since activities are performed in the person's own home setting.

3 Proposed Method

To assess RNNs in activity recognition and abnormal activity detection, we propose the following steps: Firstly, raw dataset is segmented into slices by using a sliding window approach. The window size is 60 seconds time of sensor readings as described in [28]. Secondly, sensor-based features are extracted from these slices. These features are *binary*, *change-point* and *last-fired* representations which are used also in [28]. Thirdly, RNNs (Vanilla, GRU and LSTM) are trained to recognise daily activities and encode daily-life behaviour routines. Lastly, the trained model is used to detect anomalies deviating from the normal daily-life sequences.

In the following we describe the dataset as well as the methodology used to generate artificial dataset that reflects on the typical behaviour of a person with dementia.

3.1 Dataset and Features

We used the popular dataset collected by Van Kasteren [28] from 3 households which are denoted as dataset A, B and C. The data captures daily-life activities such as sleeping, cooking, leaving home, etc. using sensors placed at the homes in less than a month. Please see [28] for more details. We applied the same sliding window approach as in [28] to extract the sensor reading chunks. We also considered three feature representations: *binary*, *change-point* and *last-fired* which are described as follows:

- *Binary*: This representation gives 1 when the sensor is triggered and 0 when that sensor is not triggered.

- *Change-point*: This representation gives information when a sensor changes value. More specifically, it gives 1 when a sensor changes its current state (either from state 1 to state 0 or vice versa) and a 0 when its value remains the same.
- *Last-fired*: This representation indicates which sensor is fired last. The sensor that changed state last continues to give 1 and changes to 0 when another sensor changes state.

3.2 Generation of Abnormal Activities Related to Dementia

Since we do not have any available dataset related to abnormal behaviour of people with dementia, we artificially create some anomalies in the dataset. In order to show the applicability of the proposed work to detect these anomalies, we focus on two different kinds of anomalies that can be seen in daily-life routines of elderly people with dementia: 1) Forgetting or repeating activities 2) Dehydration and disruption in sleep.

1. **Forgetting and repeating activities:** Elderly people suffering from dementia may forget whether they performed a particular daily activity or not, so they may repeat that activity multiple times or they may skip that activity. For instance, an elderly person suffering from Alzheimer may forget to have lunch, take multiple lunches instead [24], to have dinner and start to prepare it in the middle of the night. To reflect on this, we generate this kind of abnormal activities by manually inserting a specific set of actions within the normal activity sequence. This will result in multiple occurrences of that activity, which will occur in some inadequate time of the day such as having dinner in the middle of the night. We inject the instances of the following activities: *brushing teeth, preparing dinner, eating, getting snack* into the normal activity sequences to generate abnormal activities related to the frequency.

2. **Dehydration and disruption in sleep:** Degeneration of the sleep-waking cycle, sleep disorders and night time wandering are among the most severe behavioural symptoms of dementia. For example, elderly people may wake up many times in the night to use the toilet and go back to sleep and may forget to take daily amount of water [24, 2]. We simulate these anomalies by inserting some synthetic activities in the normal night-time activity sequences of a person. More specifically, we inject *getting drink, going to toilet* into the *sleeping* activity of normal daily activity sequences. This will emulate the activities of getting drink and going to the toilet frequently in the middle of the night.

We generate these abnormal activity instances on dataset A which has the following 9 activities: *Leave house, use toilet, take shower, brush teeth, go to bed, prepare breakfast, prepare dinner, get snack, get drink.* As a result, we have multiple instances of those injected instances in order to simulate the anomalies related to dementia. Here, please note that there is only one subject in the dataset. We take the lifestyle in the training data as a norm and then synthesise the abnormalities deviating from this norm and introduce these abnormalities in the test data. These activities are totally normal on their own but they become abnormal when they occur at a wrong time of the day and after or before a specific activity. Hence, capturing these abnormalities within the context is important. In all, we manually synthesise 135 abnormal activity slices.

3.3 Activity Recognition and Abnormal Behaviour Detection

We believe that the order of activities and their temporal and spatial information is important to encode an elderly person's daily life routines. This kind of information can provide important cues to understand the daily patterns and thus to detect any anomalies in those patterns. Sequence labelling methods such as HMMs and RNNs can capture temporal and spatial relationship between activities, which some generative methods like SVMs can not do. In this work, we investigate the adequacy of RNNs to this task.

In order to recognise daily activities, training instances of the datasets and their corresponding labels are fed into the RNNs. Then when a new test sequence is introduced, the trained model assigns labels to each activity instances of that sequence. Each model gives a confidence value about the assigned label for the new sequence. Firstly, we calculate the mean of confidence values of training instances that are assigned by the model. Then, when a new test sequence is introduced if the model assigns it to a class label with a confidence value which is bigger than the mean, the sequence is considered as a normal activity, otherwise it is abnormal activity.

3.4 RNN Architectures

In the following we give a summary of the RNN architectures used in this work, more specifically Vanilla RNNs, Long Short Term Memory RNNs, and Gated Recurrent Unit RNNs. Then, we describe how they are used in the context of daily activity recognition and abnormal activity detection tasks.

1. **Vanilla Recurrent Neural Networks:** In feed-forward neural network, it is assumed that all inputs and outputs are independent of each other, but RNNs have a recurrent hidden state whose activation at each time is dependent on that of the previous time. This architecture is recurrent as some of the connections within the network form a directed cycle, where the current time-step t considers the states of the network in the previous time-step $t-1$. They share parameters for different time-steps which enables them to be used in sequential data. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being dependent on the previous computations. Another way

to think about RNNs is that they have a memory which captures information about what has been calculated so far. However, there is a drawback of Vanilla RNNs, as shown by Bengio et al. [3], Vanilla RNNs are not capable of capturing long term dependencies on sequences because of the vanishing gradient problem. In theory, RNNs can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps. Thus, the following two RNN architectures are exploited to solve this problem.

2. **Long Short Term Memory (LSTM) Recurrent Neural Networks:** LSTM cells are designed to counter the effect of diminishing gradients when error derivatives are backpropagated through many layers through time in recurrent networks [14]. Each LSTM unit keeps track of an internal state that represents its memory. Over time the cells learn to output, overwrite, or null their internal memory based on their current input and the history of past internal states, leading to a system capable of retaining information across hundreds of time-steps [14]. LSTM blocks have 3 gates to control the flow of information into or out of their memory. For example, an *input gate* controls the extent to which a new value flows into the memory. A *forget gate* controls the extent to which a value remains in memory while an *output gate* is used to compute the output activation of the block (see Figure 1).

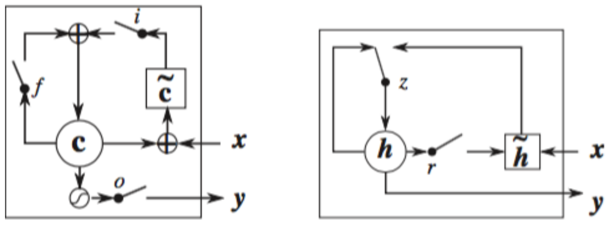


Figure 1: Left: LSTM, Right: GRU. While LSTM can be described as the input signals x_t at time t, the output signals y_t, the forget gate f_t, and the input gate i_t, the output gate o_t,; GRU, on the other hand, can be described in terms of two internal variables, which retain the previous h and current h inner states respectively.

3. **Gated Recurrent Unit:** Cho et al. [3] recently proposed GRU, which is like LSTM but it has fewer parameters than LSTM, as GRUs lack an *output gate*. In GRU, each hidden unit has two gates, which are called *update* and *reset gates* (see Figure 1). GRU also controls the flow of information to prevent vanishing gradient problem, but without having to use a memory unit.

4 Experiments and Results

We used Keras Deep Learning library's [11] and Theano's [25] implementations of the RNNs (GRU, LSTM, Vanilla RNN) in this study. Moreover for the sake of comparison, we also used the One-class SVM from WEKA with default parameters, Naïve Bayes (NB), Hidden Markov Models (HMM), Hidden Semi-Markov Models (HSMM) and Conditional Random Fields (CRF) which are based on the implementation provided in [28].

We split the data (see Sec. 3.1) into a test and training set using the leave-one-day-out cross-validation approach. One full day of sensor readings is used for testing and the remaining days are used for training. Then we cycle over all days and report the average performance.

We evaluate metrics proposed in [28]: precision, recall, F-measure and accuracy. We calculate precision and recall for each class separately and then take the average over all classes. Note that precision and recall measures are used since these metrics give some idea about how well the models perform on imbalanced datasets like the one in this study. On the other hand, the accuracy represents the percentage of correctly classified time slices, therefore more frequently occurring classes have a larger weight in this measure.

To evaluate the performance of abnormal behaviour detection, we use the following evaluation metrics: True Positive Rate (TPR) and False Positive Rate (FPR). TPR is the percentage of correctly detected abnormal activities out of total abnormal activities, FPR is the percentage of normal activities that are detected falsely as abnormal activities by the algorithm (out of total number of normal activities).

To run experiments on RNNs, we left out 10% of the training data for validation and we used drop-out with a value of 0.2. We also set the batch size to 10 instances and the epoch to 500 iterations. The internal architecture of RNNs (2 layers consisting of 30 and 50 nodes respectively) and time step of the sequences (25 activity slices) were empirically set.

Note that the results obtained by the models HMM, HSMM, CRF and NB (see Tab. 1 - 3) are taken from the study by Kasteren et al. [28].

Table 1 refers to the results obtained on dataset A and shows that there is no clear winner among the three different feature representations. Considering the accuracy, the results indicate that LSTM is the best method (with the accuracy of 96.7%) when *last-fired* feature is used, while HMM performs the worst. Using *change-point* feature, HMM outperforms all other methods. Using *binary* feature on the other hand shows that CRF (accuracy of 89.8%) is the best. Also all RNNs, NB and SVM do not perform well when adopting *change-point* feature. HMM and HSMM are not good when using *binary* feature representation. In a nutshell, for the majority of the methods, except HMM and HSMM, *last-fired* representation is the best one. In terms of recall which reflects better on performance in the presence of imbalanced data, the highest value is obtained by GRU (80.6%). This potentially indicate that RNNs are good to detect relevant class instances. CRF, for instance, score higher on precision, because the most frequent-class instances are favoured, but then it

Table 1: Activity recognition results on dataset *A*

Model	Feature	Precision	Recall	F-Measure	Accuracy
NB	Binary	48.3 ± 17.7	42.6 ± 16.6	45.1 ± 16.9	77.1 ± 20.8
	Change-point	52.7 ± 17.5	43.2 ± 18.0	47.1 ± 17.2	55.9 ± 18.8
	Last-fired	67.3 ± 17.2	64.8 ± 14.6	65.8 ± 15.5	95.3 ± 2.8
HMM	Binary	37.9 ± 19.8	45.5 ± 19.5	41.0 ± 19.5	59.1 ± 28.7
	Change-point	70.3 ± 16.0	74.3 ± 13.3	72.0 ± 14.2	92.3 ± 5.8
	Last-fired	54.6 ± 17.0	69.5 ± 12.7	60.8 ± 14.9	89.5 ± 8.4
HSMM	Binary	39.5 ± 18.9	48.5 ± 19.5	43.2 ± 19.1	59.5 ± 29.0
	Change-point	70.5 ± 16.0	75.0 ± 12.1	72.4 ± 13.7	91.8 ± 5.9
	Last-fired	60.2 ± 15.4	73.8 ± 12.5	66.0 ± 13.7	91.0 ± 7.2
CRF	Binary	59.2 ± 18.3	56.1 ± 17.3	57.2 ± 17.3	89.8 ± 8.5
	Change-point	73.5 ± 16.6	68.0 ± 16.0	70.4 ± 15.9	91.4 ± 5.6
	Last-fired	66.2 ± 15.8	65.8 ± 14.0	65.9 ± 14.6	96.4 ± 2.4
Vanilla	Binary	46.5 ± 17.7	64.8 ± 16.2	53.5 ± 16.3	86.8 ± 10.6
	Change-point	46.3 ± 19.5	63.8 ± 16.4	53.2 ± 17.9	61.4 ± 16.4
	Last-fired	61.9 ± 19.1	74.3 ± 12.8	67.2 ± 16.4	95.5 ± 3.4
LSTM	Binary	50.8 ± 18.4	63.9 ± 16.5	56.2 ± 17.1	86.7 ± 10.5
	Change-point	46.8 ± 18.7	63.6 ± 14	53.5 ± 16.7	61.4 ± 16.4
	Last-fired	63.7 ± 19.9	73.9 ± 16.8	68.1 ± 18.2	96.7 ± 2.6
GRU	Binary	47.3 ± 18.7	69.1 ± 14.9	55.4 ± 16.5	86.6 ± 10.7
	Change-point	42.9 ± 19	65.0 ± 15.3	51.0 ± 17.1	61.4 ± 16.4
	Last-fired	61.8 ± 16.3	80.6 ± 11.5	69.5 ± 14.0	96.1 ± 2.5
SVM	Binary	45.6 ± 17.9	69.1 ± 15.9	54.2 ± 15.9	85.4 ± 10.4
	Change-point	40.3 ± 19.1	63.4 ± 14.6	48.6 ± 17.0	55.9 ± 18.7
	Last-fired	58.6 ± 16.2	77.2 ± 14.0	66.3 ± 14.9	96.1 ± 2.4

Table 2: Activity recognition results on dataset *B*.

Model	Feature	Precision	Recall	F-Measure	Accuracy
NB	Binary	33.6 ± 10.9	32.5 ± 8.4	32.4 ± 8.9	80.4 ± 18.9
	Change-point	40.9 ± 7.2	38.9 ± 5.7	39.5 ± 5.9	67.8 ± 18.6
	Last-fired	43.7 ± 8.7	44.6 ± 7.2	43.3 ± 4.8	86.2 ± 13.8
HMM	Binary	38.8 ± 14.7	44.7 ± 13.4	40.7 ± 12.4	63.2 ± 24.7
	Change-point	48.2 ± 17.2	63.1 ± 14.1	53.6 ± 16.5	81.0 ± 14.2
	Last-fired	38.5 ± 15.8	46.6 ± 19.5	41.8 ± 17.1	48.4 ± 26.9
HSMM	Binary	37.4 ± 16.9	44.6 ± 14.3	39.9 ± 14.3	63.8 ± 24.2
	Change-point	49.8 ± 15.8	65.2 ± 13.4	55.7 ± 14.6	82.3 ± 13.5
	Last-fired	40.8 ± 11.6	53.3 ± 10.9	45.8 ± 11.2	67.1 ± 24.8
CRF	Binary	35.7 ± 15.2	40.6 ± 12.0	37.5 ± 13.7	78.0 ± 25.9
	Change-point	48.3 ± 8.3	51.5 ± 8.5	49.7 ± 7.9	92.9 ± 6.2
	Last-fired	46.9 ± 12.5	47.8 ± 12.1	46.6 ± 12.9	89.2 ± 13.9
Vanilla	Binary	26.7 ± 13.5	46.9 ± 24.8	32.5 ± 17.9	65.2 ± 34.7
	Change-point	39.6 ± 8	62.4 ± 15.3	48.3 ± 10.2	76.9 ± 13.9
	Last-fired	41.2 ± 12.3	64.4 ± 17.8	49.7 ± 13.6	87.9 ± 13.1
LSTM	Binary	29.1 ± 12.0	44.0 ± 22.0	33.9 ± 16.2	63.5 ± 32.7
	Change-point	40.0 ± 11.2	59.0 ± 16.4	47.5 ± 12.9	76.8 ± 14.2
	Last-fired	40.8 ± 10.7	60.1 ± 16.3	48.2 ± 12.3	87.2 ± 13.2
GRU	Binary	28.5 ± 15.9	36.3 ± 17.2	31.4 ± 16.2	64.5 ± 32.1
	Change-point	37.7 ± 7.6	53.5 ± 9.2	44.9 ± 7.1	76.4 ± 14.5
	Last-fired	41.7 ± 13.2	56.9 ± 17.9	47.5 ± 14.6	87.0 ± 12.9
SVM	Binary	39.6 ± 10.9	58.5 ± 17.4	46.7 ± 12.9	81.6 ± 18.5
	Change-point	32.3 ± 6.5	53.6 ± 7.5	40.0 ± 6.2	67.9 ± 28.5
	Last-fired	36.4 ± 5.4	54.6 ± 10.4	43.5 ± 6.6	86.2 ± 14.9

Table 3: Activity recognition results on dataset C

Model	Feature	Precision	Recall	F-Measure	Accuracy
NB	Binary	19.6 ± 11.4	16.8 ± 7.5	17.8 ± 9.1	46.5 ± 22.6
	Change-point	39.9 ± 6.9	30.8 ± 4.8	34.5 ± 4.6	57.6 ± 15.4
	Last-fired	40.5 ± 7.4	46.4 ± 14.8	42.3 ± 6.8	87.0 ± 12.2
HMM	Binary	15.2 ± 9.2	17.2 ± 9.3	15.7 ± 8.8	26.5 ± 22.7
	Change-point	41.4 ± 8.8	50.0 ± 11.4	44.9 ± 8.8	77.2 ± 14.6
	Last-fired	40.7 ± 9.7	53.7 ± 16.2	45.9 ± 11.2	83.9 ± 13.9
HSMM	Binary	15.6 ± 9.2	20.4 ± 10.9	17.3 ± 9.6	31.2 ± 24.6
	Change-point	43.8 ± 10.0	52.3 ± 12.8	47.4 ± 10.5	77.5 ± 15.3
	Last-fired	42.5 ± 10.8	56.0 ± 15.4	47.9 ± 11.3	84.5 ± 13.2
CRF	Binary	17.8 ± 22.1	21.8 ± 20.9	19.0 ± 21.8	46.3 ± 25.5 ‘
	Change-point	36.7 ± 18.0	39.6 ± 17.4	38.0 ± 17.6	82.2 ± 13.9
	Last-fired	37.7 ± 17.1	40.4 ± 16.0	38.9 ± 16.5	89.7 ± 8.4
Vanilla	Binary	15.4 ± 5.3	43.1 ± 18.1	22.2 ± 7.3	50.2 ± 22.4
	Change-point	31.3 ± 7.1	54.9 ± 11.3	39.5 ± 8.3	72.2 ± 13.0
	Last-fired	38.3 ± 16.3	59.6 ± 15.1	45.8 ± 14.8	86.7 ± 12.5
LSTM	Binary	16.8 ± 6.2	34.8 ± 12.5	22.1 ± 7.4	45.3 ± 21.2
	Change-point	31.0 ± 5.1	53.3 ± 6.5	38.9 ± 5.0	72.0 ± 13.0
	Last-fired	41.3 ± 17.2	57.3 ± 15.9	47.5 ± 16.1	87.4 ± 12.4
GRU	Binary	18.7 ± 8.3	33.2 ± 12.7	23.9 ± 9.6	46.7 ± 23.4
	Change-point	31.2 ± 8.3	47. ± 10.9	31.2 ± 8.5	71.6 ± 12.6
	Last-fired	40.4 ± 16.5	52.7 ± 16.4	45.4 ± 16.9	86.6 ± 12.3
SVM	Binary	19.4 ± 9.0	35.2 ± 12.7	24.0 ± 9.2	37.4 ± 19.0
	Change-point	25.6 ± 6.2	51.4 ± 9.5	34.0 ± 7.2	57.8 ± 15.5
	Last-fired	37.0 ± 7.9	55.5 ± 11.6	44.1 ± 8.5	87.5 ± 12.1

is not so good at when it comes to the infrequent classes. Overall, there is a clear hint that that recurrent architectures perform better than HM, NB and HSMM for most of the cases, while CRF is slightly better than these recurrent architectures on dataset A.

Table 2 refers to the results obtained on dataset B and shows that SVM is the best method when adopting *binary* representation achieving the accuracy of 81.6%. On the other hand, CRF is the best when using the *change-point* feature and *last-fired* representations with accuracy 92.9% and 89.2% respectively. It can be noted that HMM is not as good as the other methods achieving in the best case only 81.0% with the *change-point* representation. The closest successful model to CRF is Vanilla RNN and again overall RNNs deliver high recall rates compared to the other methods. *Change-point* and *last-fired* representations give the highest recall results except for CRF.

Table 3 reports the results on dataset C showing that CRF performs best for *change-point* and *binary* representations obtaining 82.2% and 89.7% respectively. Overall, none of the methods performs well when adopting *binary* representation. The results are slightly better with *change-point* but clearly better when applying the *last-fired* representation. RNNs again give the highest recall values for all representations. Overall, the results show that RNNs perform better than HMM, NB and HSMM in all cases, while CRF is slightly better than RNNs. But in terms of recall, these later outperform all methods for all feature representations. The reason behind this is that RNNs perform better for imbalanced data compared to CRF. RNNs variants generally perform equally well.

For abnormal activity detection, we considered LSTM only and compared against NB, HSMM, HMM, SVM and CRF. TPR and FPR accuracy percentages are correspondingly; 40.40% and 43.50% for NB, 58.36% and 96.20% for HMM, 68.85% and 32.2% for HSMM, 66.22% and 40.50% for CRF, 72.11% and 44.0% for One-class SVM and 91.43% and 40.96% for LSTM. We used only *last-fired* feature in this experiment. The results indicate that LSTM is the best to prune false negatives compared to the other methods. Methods like NB, One-class SVM which do not capture the data order performs the worst. The models ignore the frequency of the activity, but apply the temporal and contextual information to make a decision. Results show that LSTM is capable of encoding the order of activities. Hence, when an activity is introduced in a different context or in a different order, LSTM can detect such anomalies.

Our current approach may fail to detect abnormalities, when there is gradual deterioration regarding the health of an elderly. We are planning to deal with this issue in the future while collecting real-world data in which gradual deterioration can be observed.

5 Conclusion

In this paper, we showed that RNNs perform well on the problem of activity recognition. They are also able to cope quite well with imbalanced data as well as anomaly detection which is very important in the context of dementia. Compared to a number of traditional and popular techniques used for activity recognition such as SVM, NB, HMM and HSMM, they perform much better, while remained very competitive with CRF. Furthermore, the empirical experiments showed that the three variants of RNNs generally perform equally well, but LSTM seems to be slightly better across all datasets used in this study. Moreover, in terms of representation, there is no clear preference, but *last-fired* feature seems to be better, at least on the datasets A and C, compared to the *change-point* and *binary* representations. Overall the study allowed to confirm that RNNs are very appropriate for activity recognition and abnormal activity detection. In our future investigations, we will extend RNNs to deep neural networks. We will also aim at collecting a dataset from a smart home dedicated to elderly people with dementia to further study behaviour anomalies related to dementia.

Acknowledgments

We thank to Tim Van Kasteren for providing the code and datasets.

References

[1] Charissa A. and Sung-Bae C. Evaluation of deep convolutional neural network architectures for human activity recognition with smartphone sensors. *In Proceedings of the KIISE Korea Computer Congress*, page 858860, 2015.

[2] M. Amiribesheli and A. Bouchachia. Smart homes design for people with dementia. In *2015 International Conference on Intelligent Environments*, pages 156–159, 2015.

[3] K. Cho, B. Merrienboer, D. Bahdanau, and Y. Bengio. *On the properties of neural machine translation: Encoder-decoder approaches.* 2014.

[4] S. Choi, E. Kim, and S. Oh. Human behavior prediction for smart homes using deep learning. In *2013 IEEE RO-MAN*, pages 173–179, 2013.

[5] P. Dawadi, D. Cook, C. Parsey, M. Schmitter-Edgecombe, and M. Schneider. An approach to cognitive assessment in smart homes. In *Knowledge Discovery and Data Mining Workshop on Medicine and Healthcare*, 2011.

[6] P. Dawadi, D. Cook, and M. Schmitter-Edgecombe. Smart home-based longitudinal functional assessment. In *ACM UbiComp Workshop on Smart Health Systems and Applications*, 2014.

[7] M. Ermes, J. Parkka, and L. Cluitmans. Advancing from offline to online activity recognition with wearable sensors. *30thAnnual International Conference of the IEEE Engineering in Medicine and Biology Society*, page 44514454, 2008.

[8] M. S. Albert et al. The diagnosis of mild cognitive impairment due to Alzheimer's disease: Recommendations from the national institute on aging-alzheimer's association workgroups on diagnostic guidelines for alzheimer's disease. *Alzheimer's & Dementia*, Volume 7-Issue 3:270–279, 2011.

[9] H. Fang and C. Hu. Recognizing human activity in smart home using deep learning algorithm. In *33rd Chinese Control Conference*, pages 4716–4720, 2014.

[10] H. Fang, H. Si, and L. Chen. Recurrent neural network for human activity recognition in smart home. *In Proceedings of 2013 Chinese Intelligent Automation Conference*, pages 341–348, 2013.

[11] C. François. Keras. https://github.com/fchollet/keras, 2015.

[12] N. Hammerla, J. Fisher, P. Andras, L. Rochester, R. Walker, and T. Plotz. PD disease state assessment in naturalistic environments using deep learning. pages 1742–1748, 2015.

[13] N. Hammerla, S. Halloran, and T. Pltz. Deep, convolutional, and recurrent models for human activity recognition using wearables. *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, 2016.

[14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, Nov 1997.

[15] M. R. Hodges, K. Kirsch, M. W. Newman, and M. E. Pollack. Automatic assessment of cognitive impairment through electronic observation of object usage. In *Pervasive*, pages 192–209, 2010.

[16] T. Kirste, A. Hoffmeyer, P. Koldrack, A. Bauer, S. Schubert, S. Schrder, and S. Teipel. Detecting the effect of Alzheimer's disease on everyday motion behaviour. *Journal of Alzheimer's Disease*, Journal of Alzheimer's Disease:121–132, 2014.

[17] O. D. Lara and M. A. Labrador. A mobile platform for real time human activity recognition. *IEEE Conference on Consumer Communications and Networks*, 2012.

[18] O. D. Lara and M. A. Labrador. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys Tutorials*, 15(3):1192–1209, 2013.

[19] A. Lotfi, Caroline Langensiepen, Sawsan M Mahmoud, and M Javad Akhlaghinia. Smart homes for the elderly dementia sufferers: identification and prediction of abnormal behaviour. *Journal of ambient intelligence and humanized computing*, 3(3):205–218, 2012.

[20] F. J. Ordonez and D. Roggen. Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016.

[21] T. Plötz, N. Hammerla, and P. Olivier. Feature learning for activity recognition in ubiquitous computing. volume 2, pages 1729–1734, 2011.

[22] D. Riboni, C. Bettini, G. Civitarese, Z. H. Janjua, and V. Bulgari. From lab to life: Fine-grained behavior monitoring in the elderly's home. *In Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 344–349, 2015.

[23] D. Riboni, C. Bettini, G. Civitarese, Z. Haider Janjua, and R. Helaoui. Fine-grained recognition of abnormal behaviors for early detection of mild cognitive impairment. *IEEE International Conference on Pervasive Computing and Communications*, 2015.

[24] J. Saives, C. Pianon, and G. Faraut. Activity discovery and detection of behavioral deviations of an inhabitant from binary sensors. *IEEE Transactions on Automation Science and Engineering*, 12(4):1211–1224, 2015.

[25] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, may 2016.

[26] W. Thies and L. Bleiler. 2013 Alzheimer's disease facts and figures. *Alzheimer's & dementia: the journal of the Alzheimer's Association vol. 9, no. 2*, page 208245, 2013.

[27] D. A. Umphred, R. T. Lazaro, M. Roller, and G. Burton. Neurological rehabilitation. In *Elsevier Health Sciences, vol. 27, no. 5*, 2013.

[28] T. Van Kasteren, G. Englebienne, and B. J. A. Kröse. Human activity recognition from wireless sensor network data: Benchmark and software. *Activity Recognition in Pervasive Intelligent Environments*, pages 165–186, 2011.

[29] K. Wild. Aging changes. In *Geraotechnology, Vol. 9 No 2*, pages 121–125, 2010.

[30] J. Yang, M. Nguyen, P. San, X. Li Li, and S. Krishnaswamy. Deep convolutional neural networks on multichannel time series for human activity recognition. pages 3995–4001, 2015.

[31] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In 6^{th} *International Conference on Mobile Computing, Applications and Services*, pages 197–205, 2014.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Leman Akoglu, Hanghang Tong, and Danai Koutra. "Graph Based Anomaly Detection and Description: A Survey". In: *Data Min. Knowl. Discov.* 29.3 (May 2015), pp. 626–688. DOI: `10.1007/s10618-014-0365-y`. URL: `http://dx.doi.org/10.1007/s10618-014-0365-y`.

[2] S. S. Akter and L. B. Holder. "Activity Recognition Using Graphical Features". In: *2014 13th International Conference on Machine Learning and Applications.* Springer International Publishing, 2014, pp. 165–170.

[3] M. S. Albert and et al. "The diagnosis of mild cognitive impairment due to Alzheimer's disease: Recommendations from the National Institute on Aging". In: *Alzheimer's & Dementia* Volume 7 (3 2011), pp. 270–279. DOI: `10.1007/s00406-012-0349-0`.

[4] Alzhemier. *Alzheimer's Disease.* 2017. URL: `www.alz.org` (visited on 08/04/2017).

[5] M. Amiribesheli and A. Bouchachia. "Smart Homes Design for People with Dementia". In: *2015 International Conference on Intelligent Environments.* 2015, pp. 156–159. DOI: `10.1109/IE.2015.33`.

[6] Mohsen Amiribesheli, Asma Benmansour, and Abdelhamid Bouchachia. "A review of smart homes in healthcare". In: *Journal of Ambient Intelligence and Humanized Computing* 6.4 (2015), pp. 495–517. DOI: `10.1007/s12652-015-0270-2`.

[7] Oya Aran et al. "Anomaly Detection in Elderly Daily Behaviour in Ambient Sensing Environments". In: *Human Behaviour Understanding: 7th International Workshop* (2016). Ed. by Mohamed Chetouani, Jeffrey Cohn, and Albert Ali Salah, pp. 51–67.

[8] Damla Arifoglu and Abdelhamid Bouchachia. "Activity Recognition and Abnormal Behaviour Detection with Recurrent Neural Networks". In: 14th *International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2017)* 110 (2017), pp. 86–93.

[9] A.Wimo, B.Winblad, and L. Jonsson. "The worldwide societal costs of dementia: Estimates for 2009". In: *Alzheimer's & Dementia, vol. 6, no. 2* (2010), pp. 98–103.

[10] Asier Aztiria, Golnaz Farhadi, and Hamid Aghajan. "User Behaviour Shift Detection in Intelligent Environments". In: *Ambient Assisted Living and Home Care: 4th International Workshop* (2012). Ed. by José Bravo, Ramón Hervás, and Marcela Rodríguez, pp. 90–97.

[11] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". In: *CoRR* abs/1409.0473 (2014).

[12] A. Charissa and C. Sung-Bae. "Evaluation of deep convolutional neural network architectures for human activity recognition with smartphone sensors". In: *In Proceedings of the KIISE Korea Computer Congress* (2015), pp. 858–860.

[13] H.L Chieu, W.S. Lee, and L.P Kaelbling. "Activity recognition from physiological data using conditional random fields". In: In SMA Symposium. Singapore-MIT Alliance, 2006.

[14] K. Cho et al. "On the properties of neural machine translation: Encoder-decoder approaches". In: 8th *Workshop on Syntax, Semantics and Structure in Statistical Translation* (2014). URL: `http://arxiv.org/abs/1409.1259`.

[15] S. Choi, E. Kim, and S. Oh. "Human behavior prediction for smart homes using deep learning". In: *2013 IEEE RO-MAN*. 2013, pp. 173–179. DOI: `10.1109/ROMAN.2013.6628440`.

[16] D. J. Cook et al. "CASAS: A Smart Home in a Box". In: *IEEE Computer* 46.7 (2013), pp. 62–69. DOI: `10.1109/MC.2012.328`.

[17] M. Cook D.J.; Schmitter-Edgecombe. "Assessing the quality of activities in a smart environment". In: *Methods Inf. Med.* 48 (2009), pp. 480–485. DOI: `doi:10.3414/ME0592`.

[18] M. P. Cutchin. "The process of mediated aging-in-place: A theoretically and empirically based model". In: *Social Science and Medicine, vol. 57, no. 6* (2003), pp. 1077–1090. DOI: `https://doi.org/10.1016/S0277-9536(02)00486-0`.

[19] P. N. Dawadi, D. J. Cook, and M. Schmitter-Edgecombe. "Automated Cognitive Health Assessment Using Smart Home Monitoring of Complex Tasks". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 43.6 (2013), pp. 1302–1313. ISSN: 2168-2216. DOI: `10.1109/TSMC.2013.2252338`.

[20] Prafulla Dawadi, Diane J. Cook, and Maureen Schmitter-Edgecombe. "Smart Home-based Longitudinal Functional Assessment". In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. UbiComp '14 Adjunct. New York, NY, USA: ACM, 2014, pp. 1217–1224. DOI: `10.1145/2638728.2638813`. URL: `http://doi.acm.org/10.1145/2638728.2638813`.

[21] Prafulla Dawadi et al. “An Approach to Cognitive Assessment in Smart Home”. In: *Proceedings of the 2011 Workshop on Data Mining for Medicine and Healthcare*. DMMH ’11. San Diego, California, USA: ACM, 2011, pp. 56–59. ISBN: 978-1-4503-0843-4. DOI: 10.1145/2023582.2023592. URL: http://doi.acm.org/10.1145/2023582.2023592.

[22] Prafulla N. Dawadi, Diane J. Cook, and Maureen Schmitter-Edgecombe. “Modeling Patterns of Activities Using Activity Curves”. In: *Pervasive Mob. Comput.* 28.C (June 2016), pp. 51–68. DOI: https://doi.org/10.1016/j.pmcj.2015.09.007.

[23] David Duvenaud et al. “Convolutional Networks on Graphs for Learning Molecular Fingerprints”. In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*. 2015, pp. 2224–2232. URL: http://arxiv.org/abs/1509.09292.

[24] Shirin Enshaeifar et al. “Health management and pattern analysis of daily living activities of people with dementia using in-home sensors and machine learning techniques”. In: *PLOS ONE* 13.5 (May 2018), pp. 1–20.

[25] M. Ermes, J. Parkka, and L. Cluitmans. “Advancing from offline to online activity recognition with wearable sensors”. In: *30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (2008), pp. 4451–4454. DOI: 10.1109/IEMBS.2008.4650199.

[26] H. Fang and C. Hu. “Recognizing human activity in smart home using deep learning algorithm”. In: *33rd Chinese Control Conference*. 2014, pp. 4716–4720. DOI: 10.1109/ChiCC.2014.6895735.

[27] H. Fang, H. Si, and L. Chen. “Recurrent Neural Network for Human Activity Recognition in Smart Home”. In: *In Proceedings of 2013 Chinese Intelligent Automation Conference* (2013). Ed. by Zengqi Sun and Zhidong Deng, pp. 341–348.

[28] Abdur Rahim Mohammad Forkan et al. “A context-aware approach for long-term behavioural change detection and abnormality prediction in ambient assisted living”. In: *Pattern Recognition* 48.3 (2015), pp. 628 –641. DOI: https://doi.org/10.1016/j.patcog.2014.07.007.

[29] C. François. *Keras*. https://github.com/fchollet/keras. 2015.

[30] K. S. Gayathri, Susan Elias, and Balaraman Ravindran. “Hierarchical activity recognition for dementia care using Markov Logic Network”. In: *Personal and Ubiquitous Computing* 19.2 (2015), pp. 271–285. DOI: 10.1007/s00779-014-0827-7. URL: https://doi.org/10.1007/s00779-014-0827-7.

[31] A. Graves, A. R. Mohamed, and G. Hinton. “Speech recognition with deep recurrent neural networks”. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2013, pp. 6645–6649. DOI: 10.1109/ICASSP.2013.6638947.

[32] K. Greff et al. "LSTM: A Search Space Odyssey". In: *IEEE Transactions on Neural Networks and Learning Systems* 99 (2016), pp. 1–11. URL: `http://arxiv.org/abs/1503.04069`.

[33] S. Ha and S. Choi. "Convolutional neural networks for human activity recognition using multiple accelerometer and gyroscope sensors". In: *2016 International Joint Conference on Neural Networks (IJCNN)*. 2016, pp. 381–388. DOI: `10.1109/IJCNN.2016.7727224`.

[34] N. Hammerla et al. "PD Disease State Assessment in Naturalistic Environments Using Deep Learning". In: AAAI Press, 2015, pp. 1742–1748. URL: `http://dl.acm.org/citation.cfm?id=2886521.2886562`.

[35] Nils Y. Hammerla, Shane Halloran, and Thomas Plötz. "Deep, Convolutional, and Recurrent Models for Human Activity Recognition Using Wearables". In: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. IJCAI'16. AAAI Press, 2016, pp. 1533–1540. URL: `http://dl.acm.org/citation.cfm?id=3060832.3060835`.

[36] Michiel Hermans and Benjamin Schrauwen. "Training and Analyzing Deep Recurrent Neural Networks". In: *Proceedings of the* 26th *International Conference on Neural Information Processing Systems*. NIPS'13. Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 190–198. URL: `http://dl.acm.org/citation.cfm?id=2999611.2999633`.

[37] S. Hochreiter and J. Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780. ISSN: 0899-7667. DOI: `10.1162/neco.1997.9.8.1735`.

[38] Mark R. Hodges et al. "Automatic Assessment of Cognitive Impairment through Electronic Observation of Object Usage". In: *Pervasive Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 192–209.

[39] Vi. Jakkula and D. Cook. "Detecting Anomalous Sensor Events in Smart Home Data for Enhancing the Living Experience". In: *Proceedings of the* 7^{th} *AAAI Conference on Artificial Intelligence and Smarter Living: The Conquest of Complexity*. 2011, pp. 33–37.

[40] Zaffar Haider Janjua, Daniele Riboni, and Claudio Bettini. "Towards Automatic Induction of Abnormal Behavioral Patterns for Recognizing Mild Cognitive Impairment". In: *Proceedings of the* 31st *Annual ACM Symposium on Applied Computing*. 2016, pp. 143–148.

[41] Wonjoon Kang, Dongkyoo Shin, and Dongil Shin. "Detecting and predicting of abnormal behavior using hierarchical Markov model in smart home network". In: *2010 IEEE 17Th International Conference on Industrial Engineering and Engineering Management* (2010), pp. 410–414. DOI: `10.1109/ICIEEM.2010.5646583`.

[42] T. L. M. van Kasteren, G. Englebienne, and B. J. A. Kröse. "Transferring Knowledge of Activity Recognition Across Sensor Networks". In: *Proceedings of the 8th International Conference on Pervasive Computing*. Pervasive'10. Springer Berlin Heidelberg, 2010, pp. 283–300.

[43] Tim van Kasteren et al. "Accurate Activity Recognition in a Home Setting". In: *Proceedings of the 10th International Conference on Ubiquitous Computing*. UbiComp '08. 2008, pp. 1–9. DOI: `10.1145/1409635.1409637`. URL: `http://doi.acm.org/10.1145/1409635.1409637`.

[44] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980 (2014). arXiv: `1412.6980`. URL: `http://arxiv.org/abs/1412.6980`.

[45] Thomas N. Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: *CoRR* abs/1609.02907 (2016). arXiv: `1609.02907`. URL: `http://arxiv.org/abs/1609.02907`.

[46] T. Kirste et al. "Detecting the Effect of Alzheimer's Disease on Everyday Motion Behaviour". In: *Journal of Alzheimer's Disease* Journal of Alzheimer's Disease (2014), pp. 121–132. DOI: `10.1007/978-3-319-11866-6_12`. URL: `https://doi.org/10.1007/978-3-319-11866-6_12`.

[47] G.G. Landis J.R.; Koch. "The measurement of observer agreement for categorical data". In: *Biometrics 33 (1)* (1977), pp. 159–174.

[48] O. D. Lara and M. A. Labrador. "A mobile platform for real-time human activity recognition". In: *2012 IEEE Consumer Communications and Networking Conference (CCNC)*. 2012, pp. 667–671. DOI: `10.1109/CCNC.2012.6181018`.

[49] "Learning long-term dependencies with gradient descent is difficult". In: *IEEE Transactions on Neural Networks* 5.2 (1994), pp. 157–166. DOI: `10.1109/72.279181`.

[50] Junying Li, Deng Cai, and Xiaofei He. "Learning Graph-Level Representation for Drug Discovery". In: *CoRR* abs/1709.03741 (2017). arXiv: `1709.03741`. URL: `http://arxiv.org/abs/1709.03741`.

[51] A. Lotfi et al. "Smart homes for the elderly dementia sufferers: identification and prediction of abnormal behaviour". In: *Journal of ambient intelligence and humanized computing* 3.3 (2012), pp. 205–218. DOI: `10.1007/s12652-010-0043-x`. URL: `https://doi.org/10.1007/s12652-010-0043-x`.

[52] Jens Lundström, Eric Järpe, and Antanas Verikas. "Detecting and Exploring Deviating Behaviour of Smart Home Residents". In: *Expert Syst. Appl.* 55.C (Aug. 2016), pp. 429–440. ISSN: 0957-4174.

[53] Jens Lundström, Eric Järpe, and Antanas Verikas. "Detecting and exploring deviating behaviour of smart home residents". In: *Expert Systems with Applications* 55 (2016), pp. 429–440. DOI: `https://doi.org/10.1016/j.eswa.2016.02.030`. URL: `http://www.sciencedirect.com/science/article/pii/S0957417416300616`.

[54] Seelye Adriana M., Schmitter-Edgecombe M, and Crandall A Cook DJ. "Naturalistic assessment of everyday activities and prompting technologies in mild cognitive impairment". In: *Journal International Neuropsychologly Soc. 2013 (4):* (2013), pp. 442–52. DOI: doi:10.1017/S135561771200149X.

[55] Usman Naeem et al. "Activities of daily life recognition using process representation modelling to support intention analysis". In: *International Journal of Pervasive Computing and Communications* 11.3 (2015), pp. 347–371. ISSN: 1742-7371. DOI: 10.1108/IJPCC-01-2015-0002.

[56] Office for National Statistics. *Overview of the UK population: March.* 2017.

[57] H. Nikamalfard et al. "Knowledge discovery from activity monitoring to support independent living of people with early dementia". In: *Proceedings of 2012 IEEE-EMBS International Conference on Biomedical and Health Informatics.* 2012, pp. 910–913. DOI: 10.1109/BHI.2012.6211735.

[58] Giannis Nikolentzos et al. "Kernel Graph Convolutional Neural Networks". In: *CoRR* abs/1710.10689 (2017). URL: http://arxiv.org/abs/1710.10689.

[59] F. J. Ordonez and D. Roggen. "Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition". In: *Sensors* 16.1 (2016), p. 115. DOI: 10.3390/s16010115.

[60] Fco Javier Ordóñez, Paula de Toledo, and Araceli Sanchis. "Sensor-based Bayesian detection of anomalous living patterns in a home setting". In: *Personal and Ubiquitous Computing* 19.2 (2015), pp. 259–270. DOI: 10.1007/s00779-014-0820-1. URL: https://doi.org/10.1007/s00779-014-0820-1.

[61] S. J. Pan and Q. Yang. "A Survey on Transfer Learning". In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2010), pp. 1345–1359. ISSN: 1041-4347. DOI: 10.1109/TKDE.2009.191.

[62] C. Patterson. *The state of the art of dementia research: New frontiers 2018.* Tech. rep. Alzheimer's Disease International (ADI), 2018.

[63] D. J. Patterson et al. "Fine-grained activity recognition by aggregating abstract object usage". In: *Ninth IEEE International Symposium on Wearable Computers (ISWC'05).* 2005, pp. 44–51. DOI: 10.1109/ISWC.2005.22.

[64] Romain Paulus, Richard Socher, and Christopher D. Manning. "Global Belief Recursive Neural Networks". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems.* 2014, pp. 2888–2896.

[65] T. Plötz, N. Hammerla, and P. Olivier. "Feature Learning for Activity Recognition in Ubiquitous Computing". In: *Proceedings of the 22th International Joint Conference on Artificial Intelligence* 2 (2011), pp. 1729–1734.

[66] J. B. Pollack. "Recursive Distributed Representations". In: *Artif. Intell.* 46.1-2 (Nov. 1990), pp. 77–105. ISSN: 0004-3702. DOI: https://doi.org/10.1016/0004-3702(90)90005-K.

[67] Nastaran Mohammadian Rad et al. "Convolutional Neural Network for Stereotypical Motor Movement Detection in Autism". In: *CoRR* abs/1511.01865 (2015).

[68] Parisa Rashidi and Diane J. Cook. "Activity Knowledge Transfer in Smart Environments". In: *Pervasive Mob. Comput.* 7.3 (2011), pp. 331–343. DOI: `https://doi.org/10.1016/j.pmcj.2011.02.007`.

[69] D. Riboni et al. "Fine-grained recognition of abnormal behaviors for early detection of mild cognitive impairment". In: *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom).* 2015, pp. 149–154. DOI: `10.1109/PERCOM.2015.7146521`.

[70] D. Riboni et al. "From Lab to Life: Fine-grained Behavior Monitoring in the Elderly's Home". In: *In Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communications Workshops* (2015), pp. 344–349. DOI: `10.1109/PERCOMW.2015.7134060`.

[71] Charissa Ann Ronao and Sung-Bae Cho. "Human activity recognition with smartphone sensors using deep learning neural networks". In: *Expert Systems with Applications* 59 (2016), pp. 235 –244. ISSN: 0957-4174. DOI: `http://doi.org/10.1016/j.eswa.2016.04.032`. URL: `http://www.sciencedirect.com/science/article/pii/S0957417416302056`.

[72] Marco Schreyer et al. "Detection of Anomalies in Large Scale Accounting Data using Deep Autoencoder Networks". In: *CoRR* abs/1709.05254 (2017). arXiv: `1709.05254`. URL: `http://arxiv.org/abs/1709.05254`.

[73] Youngjoo Seo et al. "Structured Sequence Modeling with Graph Convolutional Recurrent Networks." In: *CoRR* abs/1612.07659 (2016). DOI: `https://doi.org/10.1007/978-3-030-04167-0_33`.

[74] Richard Socher, Christopher D Manning, and Andrew Y Ng. "Learning continuous phrase representations and syntactic parsing with recursive neural networks". In: *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop.* 2010, pp. 1–9.

[75] Richard Socher et al. "Semi-supervised Recursive Autoencoders for Predicting Sentiment Distributions". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* EMNLP '11. 2011, pp. 151–161. ISBN: 978-1-937284-11-4. URL: `http://dl.acm.org/citation.cfm?id=2145432.2145450`.

[76] N.K. Suryadevara et al. "Forecasting the behavior of an elderly using wireless sensors data in a smart home". In: *Engineering Applications of Artificial Intelligence* 26.10 (2013), pp. 2641–2652. DOI: `https://doi.org/10.1016/j.engappai.2013.08.004`.

[77] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. "Sequence to Sequence Learning with Neural Networks". In: *CoRR* abs/1409.3215 (2014). URL: `http://arxiv.org/abs/1409.3215`.

[78] Theano Development Team. "Theano: A Python framework for fast computation of mathematical expressions". In: *arXiv e-prints* abs/1605.02688 (2016).

[79] W. Thies and L. Bleiler. ““2013 Alzheimer’s disease facts and figures”. In: *Alzheimer’s & dementia: the journal of the Alzheimer’s Association vol. 9, no. 2* (2013), pp. 208–245.

[80] Antoine Jean-Pierre Tixier et al. “Classifying Graphs as Images with Convolutional Neural Networks”. In: *CoRR* abs/1708.02218 (2017).

[81] Y. Tong, R. Chen, and J. Gao. “Hidden State Conditional Random Field for Abnormal Activity Recognition in Smart Homes”. In: *Entropy* 17.3 (2015), p. 1358. DOI: https://doi.org/10.3390/e17031358.

[82] Niall Twomey et al. “Unsupervised Learning of Sensor Topologies for Improving Activity Recognition in Smart Environments”. In: *Neurocomput.* 234.C (Apr. 2017), pp. 93–106. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2016.12.049. URL: https://doi.org/10.1016/j.neucom.2016.12.049.

[83] T. Van Kasteren, G. Englebienne, and B. J. A. Kröse. “Human Activity Recognition from Wireless Sensor Network Data: Benchmark and Software”. In: *Activity Recognition in Pervasive Intelligent Environments* (2011), pp. 165–186. DOI: https://doi.org/10.2991/978-94-91216-05-3_8.

[84] Tim Van Kasteren, G Englebienne, and B Krose. “Recognizing activities in multiple contexts using transfer learning”. In: *Proceedings of the AAAI Fall Symposium on AI in Eldercare: New Solutions to Old Problems.* (2008).

[85] Shikhar Vashishth et al. “Dating Documents using Graph Convolution Networks”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Association for Computational Linguistics, 2018, pp. 1605–1615.

[86] Gilles Virone. “Assessing Everyday Life Behavioural Rhythms for the Older Generation”. In: *Pervasive Mob. Comput.* 5.5 (2009), pp. 606–622. DOI: https://doi.org/10.1016/j.pmcj.2009.06.008.

[87] B. Wallace et al. “Design of games for measurement of cognitive impairment”. In: *International Conference on Biomedical and Health Informatics (BHI).* 2014, pp. 117–120. DOI: 10.1109/BHI.2014.6864318.

[88] Kai Wang et al. “Research on Healthy Anomaly Detection Model Based on Deep Learning from Multiple Time-Series Physiological Signals”. In: *Scientific Programming* (2016). DOI: http://dx.doi.org/10.1155/2016/5642856.

[89] K. Wild. “Aging changes”. In: *Geraotechnology, Vol. 9 No 2.* 2010, pp. 121–125.

[90] J. Yang et al. “Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition”. In: 2015, pp. 3995–4001.

[91] Rui Yao et al. “Efficient Dense Labeling of Human Activity Sequences from Wearables using Fully Convolutional Networks”. In: *CoRR* abs/1702.06212 (2017). DOI: https://doi.org/10.1016/j.patcog.2017.12.024. URL: http://www.sciencedirect.com/science/article/pii/S0031320317305204.

[92] M. Zeng et al. “Convolutional Neural Networks for human activity recognition using mobile sensors”. In: *6th International Conference on Mobile Computing, Applications and Services.* 2014, pp. 197–205. DOI: 10.4108/icst.mobicase.2014.257786.

[93] Tongda Zhang et al. “Learning movement patterns of the occupant in smart home environments: an unsupervised learning approach”. In: *Journal of Ambient Intelligence and Humanized Computing* 8.1 (2017), pp. 133–146. DOI: https://doi.org/10.1007/s12652-016-0367-2.

[94] Chong Zhou and Randy C. Paffenroth. “Anomaly Detection with Robust Deep Autoencoders”. In: *Proceedings of the* 23rd *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* KDD ’17. 2017, pp. 665–674. DOI: 10.1145/3097983.3098052. URL: http://doi.acm.org/10.1145/3097983.3098052.