Ambulatory Monitoring Using Passive RFID Technology

Wickramasinghe Mudiyanselage Asanga Sampath Bandara Wickramasinghe

School of Computer Science
The University of Adelaide

This thesis is submitted for the degree of
Doctor of Philosophy

January 2017

I would like to dedicate this thesis to my beloved family . . .

Declaration

Wickramasinghe Mudiyanselage Asanga Sampath Bandara Wickramasinghe
January 2017

Acknowledgements

This thesis would not have been possible without the support of my supervisors Professor Frank Neumann, Dr Damith Ranasinghe and Professor Renuka Visvanathan. Especially, I would like to thank Dr Damith Ransinghe for the valuable opportunity given to me to conduct my further studies at the University of Adelaide and it was because of his encouragement and motivation that I selected to study the fields of pervasive computing and machine learning. I want to thank him for all the insightful and useful comments provided during publishing my work in various conference papers and journals, and the support provided during the candidature. I also would like to express my sincere gratitude to Professor Frank Neumann for the advice, guidance and encouraging comments during my research, especially at the final stages of my candidature. His guidance, useful and detailed feedback helped me immensely when writing this thesis. I also want to thank Professor Renuka Visvanathan for the excellent coorporation, encouragement and useful feedback provided during my PhD. I feel extremely privileged to have them as my supervisory panel.

I would like to thank my colleagues at the Auto-ID Lab, specially Roberto Shinmoto Torres and Yang Su. I also like to thank my close friends in Adelaide for the support and care provided during my PhD.

I would like to thank the School of Computer Science for providing me with an excellent working environment with all the resources. In deed, I would like to thank the School Manager Jo and the other office staff, Sharyn, Julie and Lenka for their support during administrative tasks, particularly in making conference travel arrangements.

I recognise that this research would not have been possible without the financial support from The Hospital Research Foundation (THRF) and the Department of State Development-Collaborative Pathway program, Government of South Australia (CPP39). I further acknowledge the financial assistance from the School of Computer Science, The University of Adelaide.

I am thankful to Alison-Jane Hunter for her professional support in editing this thesis.

Last but not least, I am grateful to my beloved wife, to my parents and to my parents-in-law for being there for me always. Their encouragement and moral support were invaluable.

Abstract

Human activity recognition using wearable sensors is a growing field of study in pervasive computing that forms the basis for ubiquitous applications in areas like health care, manufacturing, human computer interaction and sports. A new generation of passive (batteryless) sensors such as sensor enabled RFID (Radio Frequency Identification) tags are creating new prospects for wearable sensor based applications. As passive sensors are lightweight and small, they can be used for unobtrusive monitoring. Furthermore, these sensors are maintenance free as they require no battery. However, recognising activities from passive sensor enabled RFID tags is challenging due to the sparse and noisy nature of the data streams from these sensors because they need to harvest adequate energy for successful operation. Therefore, within this thesis, we propose methods to recognise activities in real time using passive RFID technology by alleviating the adverse effects of sparsity and noise. We mainly consider ambulatory monitoring to facilitate mitigating falls in hospitals and older care settings as our application context. Specifically, three aspects are considered: i) data acquisition from sensor enabled RFID tags; ii) monitoring ambulatory movements using passive sensor enabled RFID tags to recognise activities leading to falls; and iii) detecting falls using a dense deployment of passive RFID tags.

A generic middleware architecture and a generic tag ID format to embed sensor data and uniquely identify tag capabilities are proposed to acquire sensor data from passive sensor enabled RFID tags. The characteristics of this middleware are established using experiments with RFID readers and an example application scenario.

In the context of ambulatory monitoring using passive sensor enabled RFID tags, first, an algorithm to facilitate the online interpolation of sparse accelerometer data from passive sensor enabled RFID tags is proposed followed by an investigation of features for activity recognition. Secondly, two data stream segmentation methods are proposed that can segment the data stream on possible activity boundaries to mitigate the adverse effects posed by data stream sparsity on segmentation. Thirdly, an algorithm to model the sequential nature considering previous sensor observations for a given time and their class labels to classify a sparse data stream in real time is proposed. Finally, a classification algorithm based on structured prediction is proposed to both segment and classify the sensor data stream simultaneously. The proposed methods are evaluated using four datasets that have been collected from a passive sensor enabled RFID tag with an accelerometer and successful monitoring of ambulatory movements is demonstrated to be possible by employing innovative data stream processing methods, based on machine learning.

In order to detect falls, particularly long lie situation, using a dense deployment of passive RFID tags embedded in a carpet, an efficient and scalable machine learning based algorithm is proposed. This algorithm relies only on binary tag observation information. First, it identifies possible fall locations using heuristics and then the falls are identified using machine learning from features extracted considering possible fall locations alone. From an evaluation, it is demonstrated that the proposed algorithm could successfully identify falls in real time.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction and Motivation

In hospital and older care settings, falls are common and are detrimental to patients [1]. Most of the falls are also unwitnessed [2, 3]. A UK report examining 200,000 incident reports over 12 months found that inpatient falls were the most common (40%) type of safety incident reported [4]. Furthermore, according to this report, falls in hospitals were said to be directly responsible for 26 patient deaths, 530 hip fractures, and about 1000 other fractures within a year. In addition to the physical injuries reported, the psychological consequences of falls to the individual include anxiety, depression, loss of confidence and fear of falling, and ultimately a downward spiral of decline in health [5, 6]. Falls not only cause injuries but are also costly as patients then have a longer length of stay in hospitals [7, 8]. The cost of fall-related injuries in Australia alone is reported to be A$498.2 million in 2011 and this is estimated to increase up to A$1375 million by 2051 [9]. Current best practice recommendations contribute to fall prevention in hospitals [10], nevertheless falls rates still remain unacceptably high [11].

One of the recommended fall prevention strategies to reduce falls in hospitalised patients is to increase monitoring opportunities [9]. However, increasing monitoring requires substantial resources such as nursing staff, equally during day and night time. Therefore, technological interventions such as automatic monitoring of human activities are desired [1]. The activities performed by patients while in a hospital room include walking, lying, sitting and standing, and posture transitions such as sit-to-stand and stand-to-sit. These activities are usually referred to as ambulatory activities and studied under the domain of Human Activity Recognition (HAR). Recognising activities leading to falls such as getting out of bed, getting out of a chair and walking allows caregivers to take preventive measures once they are notified [12].

Currently, alarm systems based on sensors attached to the furniture have been proposed to identify patient transfers out of a bed or chair and walking [13–15]. These alarm systems then alert caregivers with the aim of staff promptly attending to the patient; thereby potentially reducing the risk of a fall or providing immediate assistance in case of a fall [14, 1, 16].

More recently, as opposed to sensors attached to the environment, it has been proposed that sensors worn on patients provide better opportunities to monitor multiple patients simultaneously [1]. Movement sensors attached to the body capture biomechanical characteristics of a person's movements. The use of battery powered body-worn sensors for HAR have been studied extensively in previous research [17, 18, 16]. However, battery powered sensors are obtrusive and bulky ($\approx 30\,grams$ [16]) as well as require maintenance, which hinders their application in monitoring older people.

According to studies that evaluate the acceptance of wearable sensors among older people, it has been identified that they prefer small, lightweight and low maintenance sensors [19–21]. Some people have even refused the use of sensors if they are complex to use, e.g. requiring battery maintenance [20]. The obtrusive nature of these sensors can be seen as a hindrance for translation of the technology to practice because older patients regard unobtrusiveness as one of the key acceptance criteria [22].

An emerging class of sensors that can be powered wirelessly such as passive (batteryless) sensor enabled Radio Frequency Identification (RFID) tags [23] is creating new possibilities for HAR. As opposed to battery powered sensors, passive sensor enabled RFID tags are: i) batteryless; ii) lightweight; and iii) small. Therefore, these types of sensors are expected to be unobtrusive, easy to wear and free of maintenance [24]. Consequently, such sensors possess ideal characteristics to be used as wearable sensors for older people where inconspicuousness, wearing comfort and ergonomic requirements are significant considerations for translation of technology into practice [22].

Although there are the clear advantages for passive sensor enabled RFID tags to be used as body-worn sensors, their data streams have two unique characteristics, namely *sparsity* and *noise*, which make HAR using these sensors challenging. Sparsity, the low data rates and variable time elapses between sensor observations, emanates from the limited capacity of passive sensors to power up the embedded sensors to sample data from the sensors regularly [23, 25] and data acquisition from passive sensor enabled RFID tags are random due to the use of the ISO 18000-6C protocol [26]. Furthermore, inadequate power to the embedded sensor, as well as having to sacrifice the accuracy and precision of sensor device measurements, for example, the use of a lower resolution (voltage), to achieve lower power consumption of sensing circuitry, results in increased noise in the acquired data (measurement noise).

The main goal of this thesis is to devise methods to monitor ambulatory movements of older people in real time using a passive sensor enabled RFID tag embedded with an accelerometer to facilitate fall prevention in hospital settings. To this end, this thesis proposes methods to acquire data from passive sensor enabled RFID tags and analyse them using machine learning techniques while mitigating the challenges presented by the unique nature of data streams from passive sensor enabled RFID tags. Previously, machine learning techniques have shown to perform well in HAR despite the variations among individuals when performing activities. Furthermore, timely detection of falls is still significant as falls are unwitnessed and long lie incidents in particular are detrimental to fallers. Therefore, this thesis also considers the use of commercially available passive RFID technology for fall detection using machine learning methods.

1.2 Main Contribution and Thesis Outline

This thesis focuses on the use of passive RFID technology for ambulatory monitoring. In Chapter 2, we initially provide the background in three main directions. First, we discuss Human Activity Recognition (HAR) with a review of the existing literature, followed by a description of common machine learning algorithms used in HAR. Secondly, an introduction to the passive sensor enabled RFID tags is presented, including an introduction to the RFID technology. Finally, datasets utilised in this study are presented with a description of the dataset characteristics.

The main contributions of this thesis are in the direction of mitigating the adverse effects of the data streams from sensor enabled passive RFID tags for HAR. In particular, these contributions focus on three aspects: i) data acquisition from passive sensor enabled RFID tags; ii) ambulatory monitoring using machine learning from passive sensor enabled RFID tags; and iii) use of passive RFID technology based dense sensing for fall detection. The remainder of this section summarises the main contributions of this thesis.

- *Facilitate sensor data acquisition from passive sensor enabled RFID tags:* In Chapter 3, we look at the data collection from passive sensor enabled RFID tags. In particular, sensor data from passive sensor enabled RFID tags are acquired by embedding the sensor data in the tag ID [23]. However, existing middleware for data acquisition from RFID systems and sensor networks do not support the data acquisition method used in passive sensor enabled RFID tags. Therefore, existing data acquisition middleware for RFID systems and sensor networks cannot be readily used.

We propose an extensible and generic middleware framework for managing both ID data streams and ID and sensor data streams that occur from passive sensor enabled RFID tags. The proposed middleware conforms to the standardised Electronic Product Code (EPC) global architecture [27]. We also propose a generic data format to embed sensor data in a tag ID, a new tag data model for integrated sensor and ID data representation, and an extensible data model to support subscription and reporting of ID and sensor data to client applications. This work has been published in proceedings of the IEEE International Conference on RFID [28].

- *Investigation of features for activity recognition from passive sensor enabled RFID tag data streams:* In Chapter 4, we conduct an investigation of features from passive sensor enabled RFID tag embedded with an accelerometer for machine learning based HAR. The passive sensor enabled RFID tag used in this thesis allows the acquisition of acceleration on three axes and the RFID infrastructure enriches the acceleration data with information such as the strength of the received signal and antenna that captured the sensor observation. In the case of machine learning based HAR, it is paramount to use features that provide descriptions of the interested movement patterns present in the data stream. Although there are evaluations of features for activity recognition using battery-powered sensors, to the best of our knowledge such an evaluation has not been carried out for data streams from passive sensor enabled RFID tag data streams.

  We propose novel features for ambulatory monitoring by analysing the characteristics of an older patient getting out of bed. Features were calculated from both acceleration data and information from the RFID platform. We also propose the dynamic sensor data augmentation algorithm to facilitate the interpolation of sparse accelerometer data from a sensor enabled RFID tag and evaluated five interpolants to calculate features that can be obtained using data streams with fixed sampling rates. Finally, we investigate whether features calculated using additional preprocessing is advantageous over the use of features readily available from the unprocessed data stream. This work was published in the International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services proceedings [29].

- *Online sensor data stream segmentation based on natural activity boundaries:* In Chapter 5, we look at online data stream segmentation for real-time activity recognition. It is a regular practice in the HAR domain to segment the sensor tag stream to enable calculation of features to capture activity characteristics. The sparsity of the sensor data streams from passive sensor enabled RFID tags results in issues such as the inclusion of sensor observations from the distant past for a given segment when fixed sample segmentation schemes are used. Segmenting the data stream on activity boundaries can alleviate these issues.

We propose two data stream segmentation methods based on detecting natural activity boundaries from sensor data streams to overcome the limitations in using conventional data stream segmentation methods. The possible activity boundaries are identified using the activity boundary score, which is calculated considering the trunk rotational motion from consecutive sensor observations. Therefore, sensor data segmentation schemes are simple, inexpensive, bear no assumptions on sampling rates and relay only on received sensor observations and hence are suitable for real-time applications. This work has been published in the IEEE Sensors Journal [30].

- *Real-time classification of sparse sensor data streams:* In Chapter 6, we consider sequence learning and classification using sparse data streams from passive sensor enabled RFID tags for real-time HAR. To capture the sequential nature of activities, sequence learning algorithms, such as Conditional Random Fields (CRF) [31], that model the sequential nature of observations as a first-order Markov chain have been used previously. This approach of modelling the sequential nature of activities can cause issues with sparse sensor data streams from passive sensor enabled RFID tags.

  We propose a sequence learning algorithm that is suitable for sparse sensor data streams from passive sensor enabled RFID tags for real-time ambulatory monitoring. In particular, instead of relying on consecutive observations, like in regular sequence learning algorithms, the proposed sequence learning algorithm considers recent past sensor observations and their activity labels to model their sequential nature, present in sparse data streams. We also provide a detailed description of features proposed to the capture sequential nature of activities. This work has been published in the IEEE Journal of Biomedical and Health Informatics [32].

- *Segmentation free activity classification in real time:* In Chapter 7, we consider combining the data stream segmentation and classification into a single step. Existing HAR approaches mostly rely on fixed size segmentation schemes. The use of fixed size segmentation methods poses issues such as misalignment of the activity boundaries with segment boundaries.

We propose a structured predictor based data stream classification approach that does not require explicit segmentation. Given a sequence of sensor observations, the structured predictor outputs a two-part structure that partitions the sequence of sensor observations into two parts and also holds the class label of the part 1. Then this output structure is used to classify a data stream in real time. In contrast with previous HAR approaches, where segmentation is considered as a step prior to data stream classification, the proposed approach simultaneously segments the data stream and assign class labels to each sensor observation. Thus the segmentation is performed based on a learned prediction model.

- *Device free fall detection using RFID base dense sensing:* In Chapter 8, we look at an environment instrumentation approach to detect falls instead of using body-worn sensing devices. In fact, timely detection of falls is also important to provide immediate assistance in case of a fall to avoid long lie incidents. Existing fall detection approaches rely on obtrusive wearable sensors, privacy invasive video monitoring and floor based sensors that require custom made components.

  We propose the use of a smart carpet and an efficient machine learning based approach to detect falls. The smart carpet is an embodiment of a dense deployment of commercially available passive RFID tags in a commercially available carpet to create a 2D monitoring area. The proposed algorithm relies on binary tag observations formulated as a binary image to identify possible fall locations considering unobserved tag patches initially and later classify them using machine learning based classifiers to recognise falls. As this algorithm only relies on binary tag observation, it is resilient to the noisy nature of data from the RFID platform. This work has been published in the Pervasive and Mobile Computing [33] journal.

Chapter 2

Human Activity Recognition and RFID Technology

2.1 Introduction

In this chapter, we present the background related to Human Activity Recognition (HAR), machine learning algorithms used in HAR and provide a description of RFID technology and datasets used, as this thesis focuses on monitoring ambulatory movements of older people using passive RFID technology. We have organised this chapter into main three parts.

The first part is on HAR (see Section 2.2). Initially, we present general steps that can be found in machine learning based HAR followed by a discussion on sensor deployment strategies. Then we specifically review the types of sensors and activity recognition approaches used in previous activity recognition studies in the context of wearable sensors. As most of the discussed activity recognition approaches relied on machine learning based activity classifiers, a description of commonly-utilised machine learning classifiers is presented in Section 2.2.4.

The second part details the RFID and sensor technology focused on in this thesis. Initially, we provide a brief description of RFID technology. Then we provide a detailed description of the passive sensor enabled RFID tags, which is the sensor platform to be used for monitoring ambulatory movements. In particular, we discuss the passive sensor enabled RFID tag called WISP—Wireless Identification and Sensing Platform [23]. Henceforth, we simply refer to passive sensor enabled RFID tags as *sensor tags*. The information related to RFID technology and passive sensor enabled RFID tags is important for understanding the characteristics of the sensor tag data streams acquired from the sensor tags. m=

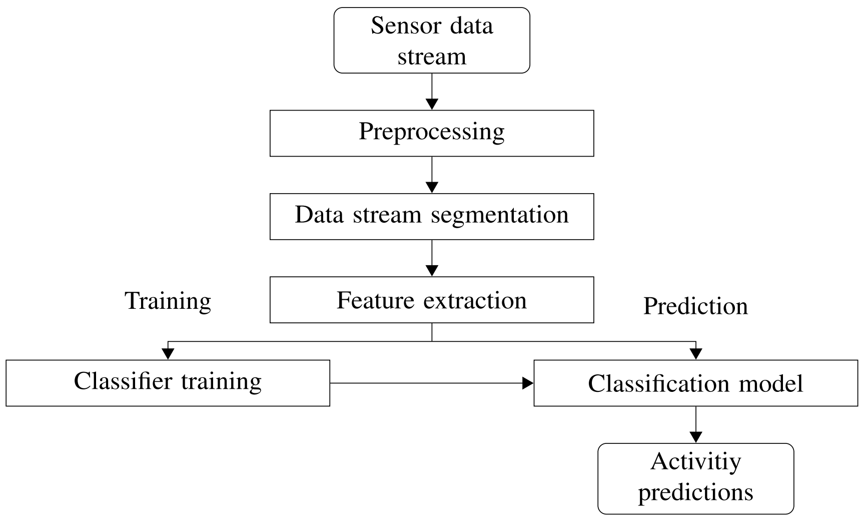


Fig. 2.1 General steps of machine learning based activity recognition

Finally, we present the details of the three datasets that have been collected from the passive sensor enabled RFID tags. We also present the dataset settings, as well as the unique characteristics of the collected sensor data streams.

2.2 Human Activity Recognition

Activity recognition is a broad field of study having a wide array of applications in areas such as human computer interaction, assisted living, medicine and manufacturing. Much work related to human activity recognition has been carried out in the fields of pervasive computing and computer vision. Figure 2.1 illustrates the general steps in human activity recognition.

As illustrated in Figure 2.1, initially, data are captured by a variety of sensors. To collect sensor data, researchers use a variety of sensor deployment strategies which are discussed in detail in Section 2.2.1.

The data streams acquired by the sensors are, often, preprocessed to remove the noise present in the sensor data. For instance, some researchers have utilised a three point median filter on acceleration signals to remove noise [34, 18, 35] and some have interpolated the signal to achieve an equal sampling rate when multiple sensors with different sampling rates are used [18, 36–38].

Followed by preprocessing, a sensor data stream is usually segmented in order to extract features. Commonly, fixed sized segmentation methods, such as fixed time [39–42] or fixed sample methods [43, 17, 44, 45], are used. In fixed sized segmentation methods, the segment size needs to be selected; while some studies select the segment by considering the HAR performance [46, 47], others consider the amount of information required by the subsequent feature extraction task [17]. Furthermore, dynamic data stream segmentation approaches for activity recognition have also been proposed [48, 49].

Features for activity recognition are extracted from data stream segments [44]. These features predominantly include time-domain statistical features and frequency domain features. In machine learning based HAR, these extracted features are arranged into a vector, which is usually termed as a feature vector. A detailed description of the commonly used features, including an evaluation, is presented in Chapter 4.

Researchers have utilised a wide array of classification algorithms for activity recognition, which are discussed in Section 2.2.3. Initially, classification models need to be learnt using collected datasets. Then the learnt classification models can be used to predict an activity that is represented by a feature vector, typically not seen during the classifier learning. In this context, the dataset used for learning the classification model is known as a training dataset and the dataset used to evaluate the performance of the learnt classification model is referred to as a testing dataset.

It is also important to note that although most of the research follow the steps described above in activity recognition, there are deviations which are discussed in Section 2.2.3.

2.2.1 Sensor Deployment Strategies

Human activity recognition approaches can be broadly categorised based on the sensor deployment approach as: i) environmental instrumentation where the sensors are deployed in the environment; and ii) subject instrumentation where the sensors are attached on various parts of the subject's body; iii) hybrid methods, where both subject instrumentation and environmental instrumentation have been used [50–52].

In the environmental instrumentation approach, researchers have used various types of sensors such as video cameras [53, 54], motion sensors [55], activation sensors [55], pressure sensors [15, 7], temperature sensors [55, 15] and accelerometers attached to everyday objects [56]. Although these approaches do not require users to wear sensors on them, they are limited by the monitoring area as the sensors are attached to the environment. In particular, the use of cameras as sensors to recognise human activities has been explored extensively in the computer vision domain. However, the correctness of these methods depends on various parameters such as the level of illumination, camera angle and subject visibility (not occluded by any object or another person) [51, 57]. Furthermore, video monitoring is considered to be privacy invasive [58]. These issues hinder the use of cameras in applications that require continuous monitoring such as older care [59].

On the other hand, in the subject instrumentation approach, a wide array of sensor types have been used. These sensor types include, accelerometers [60, 61], gyroscopes [62, 63], acoustic sensors [64], barometric pressure sensors [18] and cameras [65]. The main advantage of using body-worn sensors over the use of the sensors deployed in the environment is the continuous monitoring ability provided by the body-worn sensors because they are not limited by the location of the monitored person. Furthermore, as opposed to using environmental sensors, body-worn approaches allow researchers to associate data with individuals unambiguously. Therefore, multiple people can be monitored in the same environment. Additionally, compared with vision based approaches, body-worn sensor systems are immune to environmental parameters such as illumination. However, the major drawback of using body-worn sensors is that users may forget to wear the sensors, particularly if the sensor wearer is cognitively impaired [20].

The hybrid approaches, which use both environmental sensors and body-worn sensors, allow researchers to identify physical activities and high-level activities of daily living using localisation as well as object usage [39, 56, 66]. However, the limitations of using body-worn sensors as well as sensors placed in the environment are still present in hybrid approaches.

The study presented in this thesis focuses on the use of body-worn sensors for activity recognition, mainly due to their ability to associate the data with a specific sensor wearer. Therefore, in the remaining sections, body-worn sensors for activity recognition have been considered.

2.2.2 Body-Worn Sensor Systems

In this section, different types of body-worn sensors commonly used in HAR research are discussed. Previous researchers attached a wide array of sensors to different parts of the human body to capture information related to human motion. As mentioned previously, these sensor types include, accelerometers [62, 17, 61], gyroscopes [62, 63], acoustic sensors [64] and barometric pressure sensors [18]. Usually, these sensors are assembled into sensing devices having either a single sensor or multiple sensors. While some research use a single sensing device [34, 18, 45], others have attached multiple sensing devices in multiple body locations [17, 67, 68] to recognise human activities.

A single accelerometer has been used to obtain activity information in several studies that focus on recognising activities such as such as walking, standing, sitting and posture transitions [34, 69, 61, 70]. These studies have attached the sensor to a participant's waist [34, 71, 70] and chest in [69, 61]. The study presented in [72] recognised swimming strokes and attached the sensor on the back of the participant while activity recognition on a smart wrist watch was considered in [42].

All the studies using multiple sensors in a single sensing device have utilised an accelerometer [60, 18, 73]. The accelerometer had been complemented with a gyroscope and attached the compound sensing device to the chest [62], waist [73] and thigh [45]. In fact, the work in [45] considered placing a mobile phone in a pocket and utilised the accelerometer and the gyroscope in the mobile phone to recognise activities. Use of the gyroscope allowed these researchers to capture the rotational movement of the sensor wearer's body accurately.

On the other hand, Doukas and Maglogiannis [60] used a sensing device with a 3D accelerometer and an acoustic sensor attached to a foot to identify falls. Bianchi et al. [18] have used a single sensing device with an accelerometer and a barometric pressure sensor to recognise falls, where the barometric pressure sensor allowed the measuring of the vertical displacement more accurately by considering the pressure difference. Using this sensing device, Bianchi et al. identified slow falls, which is non-trivial using only an accelerometer.

Multiple sensing devices, each with an accelerometer, have been used to capture activity information. For instance, studies in [74, 68] have utilised two accelerometers; the former attaching them to the left and right waists while the later attached them at the waist and ankle. Four sensing devices have been used in [75, 41] where both studied selecting the chest and the thigh to attach the sensors. For other two sensors, the study in [75] selected wrists while the study in [41] selected the hip and a side of the body. Five sensing devices attached to the trunk and four limbs to recognise activities have been evaluated in [17, 76, 77, 36]. To investigate the number and position of the sensors for activity recognition Kern et al. [43] attached 12 accelerometers above all the joints.

Some studies relied on attaching multiple sensing devices, each having two or more sensing modalities to capture activity information. The sensing device used in [48, 67, 47] consisted of an accelerometer and a gyroscope. Nam et al. [65] used a smart pendant equipped with a camera and a 3D accelerometer for HAR. The environment captured by the camera has been used to identify the relative motions such as walking backwards and forward, which could not be identified based on kinematic sensors. A pair of shoes with accelerometers and pressure sensors embedded insole to recognise falls was proposed by Sazonov et al. [78]. Furthermore, accelerometers have been used with other sensing devices such as a Global Positioning System (GPS) device to identify speed and distance travelled [79], and ventilation sensors to estimate energy expenditure [80]. These studies often focus on recognising a larger number of activities compared with the number of activities that are recognised by using only a single sensing device.

All these studies use battery powered sensors. The use of batteries as a power source increases the size and weight of the sensor. Most of the work uses multiple sensors placed in different wearing positions or multi-modal sensing devices which also contribute significantly to the overall size and weight of the sensor deployment. Therefore, these sensors are obtrusive to be used in older patient monitoring settings, where unobtrusiveness is identified to be one of the major user acceptance criteria [81, 22].

Previously, our research team had employed passive sensor enabled RFID tags embedded with an accelerometer to recognise activities [46, 61]. As opposed to battery powered sensors, these sensors harvest energy similar to regular RFID tags and hence no battery is required. Therefore, these sensors are lightweight and require no maintenance, and hence ideal for unobtrusive monitoring of older patients. In Section 2.3.2, we discuss details of the passive sensor enabled RFID tags, which are used to collect datasets used in this thesis.

2.2.3 Activity Recognition Algorithms

In this section, we review different types of algorithms commonly used in activity recognition research. We broadly categorise these algorithms into empirical algorithms and machine learning based algorithms.

Empirical Algorithms

Several empirical algorithms to recognise activities have been proposed [62, 34, 16, 61]. These algorithms rely on researchers' intuition and signal analysis techniques to identify the motion patterns captured by the sensors.

The algorithm proposed by Najafi et al. [62] relied on movement signals from a 2D accelerometer and 1D gyroscope segmented into 60 s segments to recognise activities such as posture transitions (sit-to-stand and stand-to-sit), lying and walking. The signals were processed multiple times using Discrete Wavelet Transformation (DWT) to retain useful information at different stages. The lying state was identified based on vertical acceleration and, later the lying condition was identified using frontal acceleration. The sit-to-stand or stand-to-sit posture transitions were identified using the trunk rotational angle calculated based on the gyroscope signal and determining the vertical displacement. Najafi et al. [62] have evaluated their approach using data collected from 9 community-dwelling older people (age 66 ± 14 years) and achieved a mean sensitivity of 93.6% and mean specificity of 95.1%.

Karantonis et al. [34] proposed an algorithm that executes on a body-worn sensing device having a 3D accelerometer to recognise activities such as falls, sitting, standing, and lying. They initially determined the sensor wearer's state, active or at-rest, by specifying a threshold for the resultant acceleration and any significant peaks in the acceleration were interpreted as falls. The identification of upright, lying, inverted, standing and sitting were performed by specifying ranges for the trunk tilt angle measured from the direction of the gravity. For instance, trunk angles $< 60^\circ$ were considered to be upright and further if the trunk angle is $< 20^\circ$, then the sensor wearer was considered to be seated. The evaluation was carried out in a laboratory setting using six healthy participants, one with 60 years of age and rest in the age group from 22-23. They were able to achieve an accuracy of $91 \pm 6\%$ from this evaluation. A later study by the same researchers [18], has incorporated a barometric pressure sensor and improved the activity recognition performance; in particular to identify slow falls based on the pressure difference. By evaluating the new algorithm with three datasets collected from 30 young healthy participants they, reported an accuracy of over 90% for detection of falls events.

Wolf et al. [16] utilised a single sensor attached to the leg of a patient to identify falls. They have followed an approach similar to [34], where activities, such as standing or lying on the bed, were determined by specifying a threshold to the orientation of the leg. Although they have evaluated this in a geriatric ward, performance was not reported.

An algorithm to identify bed-exit events using acceleration signals and received signal strength (RSSI) of a passive sensor enabled RFID tag worn over the sternum has been proposed in [61]. First, the data stream was segmented into 20 s segments and identified the lying state using the acceleration signals. Then the sit-to-stand transition was identified considering the RSSI and trunk tilt angle approximated using the acceleration signals. From an evaluation with 10 young adults (age 26.4 ± 2.12 years), the approach in [61] have achieved a sensitivity of 93% and specificity of 98% to recognise stand-sit-lying and a sensitivity of 90% and specificity of 94% to recognise lying-sit-stand.

When empirical algorithms are considered, they are often based on the expertise of the algorithm developer. They are simple and use a single sensing device, but require repeated processing of the signal such as filtering [62, 34, 61] and application DWT [62]. Most of these algorithms, rely on researcher specified thresholds using heuristics and these thresholds are expected to be biased towards individuals. These thresholds may require adjustments for individual users and can hinders large scale deployments [82].

Machine Learning Based Approaches

Unlike in empirical approaches, where researchers use heuristics to capture relevant motion patterns, in machine learning approaches motion patterns are uncovered using machine learning algorithms. In this section, some of the noted work that uses machine learning based HAR is presented in chronological order.

Mäntyjärvi et al. [74] proposed the use of a feed-forward neural network to recognise walking related activities using data from two 3D accelerometers. They have utilised a 256-sample sliding window with 25% overlap and applied DWT on that segment. The power of the wavelet coefficients in layers 5-8 were used as features for their neural network. They have also evaluated the application of two feature transformation techniques, namely Principle Component Analysis (PCA) and Independent Component Analysis (ICA). From an evaluation using data collected from 6 participants, they were able to achieve a mean accuracy over 85% using the ICA method.

The work in [43] utilised a Naïve Bayes (NB) classifier to recognise 8 activities and evaluated different sensor position combinations. This approach used mean and variance as features, calculated using a 50-sample window. From an evaluation, this approach showed that using only lower body sensors, a good performance can be achieved for activities involving legs. When recognising other activities such as shaking hands, writing on the whiteboard and keyboard typing, a better performance was obtained using all the sensors.

The study by Bao and Intille [17] evaluated a Decision Tree (DT) (C4.5), NB and K-Nearest Neighbour (KNN) as classifiers to recognise 20 activities using a dataset that has been annotated by users instead of an observer. They have used features based on Fast Fourier Transformation (FFT) and the correlation between acceleration axes of the sensors. These features were calculated using a 512-sample sliding window with 50% overlap. The evaluation was carried out using 20 young participants (age: 21.5 ± 6.6 years) and the DT achieved the highest accuracies of $77 \pm 4\%$ for user specific training and $73 \pm 8\%$ for leave-one-participant-out cross validation.

The Hidden Markov Model (HMM) combined with a boosted decision stump was used in [83] to model the temporal relationships between activities. In this study, a boosted decision stump classifier is trained, using frequency domain features from a 0.25 s window, for each class and its margin is fitted with a sigmoid function to obtain the probabilistic input to the HMM. The HMM model was applied to a 15 s sequence with 5 s overlap to obtain real-time predictions on the data stream. From an evaluation using a dataset collected from 2 participants (12 hours of data), this approach achieved an overall accuracy of 92% when trained with 80% of data and tested with the remaining 20%.

Similar to the approach in [83], Suutala et al. [75] proposed an HMM based approach to recognise activities. However, Suutala et al. [75] utilised a Support Vector Machine (SVM) instead of the decision stump. They have used mean and standard deviations as features to train SVM classifiers and evaluated the approach with data collected from 13 participants. From this evaluation, an accuracy of over 96% has been achieved.

Doukas and Maglogiannis [60] utilised an SVM classier to recognise falls using an accelerometer and acoustic sensor. They extracted amplitude and frequency peaks at the current time as features from audio data. They considered the output as an estimation of the movement type and the falls were detected based on the number of sequential occurrences of a specific movement type. Furthermore, a Kalman filter was also used to improve the fall detection. Based on the results using a dataset collected from 2 participants, they claimed that the fall detection accuracy can be increased by eliminating false predictions in walking. Specifying the threshold for the continuous fall movement type as 10, they were able to detect all fall events and run events with an accuracy of 97%.

Ermes et al. [79] used a custom decision tree with neural network nodes to recognise sporting activities. They have used a small neural network of size 7:5:1 and the output was thresholded at 0.5 to obtain a binary split at each node. Their approach was evaluated using a dataset collected from 12 participants (age: 27 ± 9 years), which achieved an 89% accuracy.

Junker et al. [48] utilised an HMM trained to recognise hand gestures where they have considered separate HMMs for each activity. They have also proposed a data stream segmentation method that uses a similarity search. The identified segments were then classified using an HMM to recognise hand gestures. The proposed approach was evaluated with two datasets from 4 participants aged between 25-35 years; one contained object interactions and the other involving dilatory intake. From these experiments, they were able to recognise gestures with 93% recall and 74% precision for the first data set and 79% recall and 73% precision for the second dataset.

Stikic et al. [39] evaluated the use of semi-supervised approaches, namely self-training, co-training and active learning with NB for activity recognition. In self-training, high confidence predictions from a single classifier were incorporated into the training dataset for the next iteration and the classifier is trained iteratively. Instead of a single classifier, co-training requires two classifiers trained with non-overlapping feature sets where each feature set is assumed to contain sufficient to identify each class independently [84]. Active learning, on the other hand, queries the class label of the samples based on query function. They have experimented with two query strategies: i) instances with lower prediction confidence; and ii) disarrangement between two classifiers. They have evaluated the use of semi-supervised learning using the publicly available PLCouple1 dataset [85], which contains data from wearable accelerometers and infra-red motion sensors. Using this dataset, they extracted 48 features from accelerometer data and the activation counts were taken as features for the IR sensor. In the cases of co-training, they have utilised features from accelerometers and features from motion sensors in the two classifiers. Their evaluation included experimenting with various ratios of labelled data ranging from 0.3% to 12.5% and co-training depicted better performance compared with self-training. In the case of active learning, with the dataset having 12.5% of labelled data, the first query strategy obtained the highest accuracy of 64.2%.

A neural network has been utilised in [35] to recognise activities that included static activities (lying, sitting, standing), posture and activity transitions (lie-to-stand, stand-to-lie, lie-to-sit, sit-to-lie, sit-to-stand, stand-to-sit, walk-to-stand, stand-to-walk) and dynamic activities (walking, walking-upstairs, walking-downstairs and running). In contrast with [74], the approach in [35] utilised linear discriminant analysis instead of ICA. In particular, a two-stage method has been proposed in [35] to recognise activities. First, the state of the sensor wearer was identified. Secondly, the identified state was used for recognising activities. Both the state and activity recognition were carried out using neural network classifiers. They evaluated their approach using a dataset collected from 6 healthy adults (mean age 26 years) and achieved an overall accuracy of 98%.

Liu et al. [76] utilised a DT based on C4.5 algorithms within the active learning framework. Similar to the active learning approach proposed in [39], Liu et al. [76] utilised two strategies to query class labels from the user: i) instances that had a disagreement between classifiers which are trained using sensors in different locations; and ii) instances having a lower classification confidence. By evaluating the proposed method with the dataset used in [17], Liu et al. [76] were able to achieve an accuracy over 75% using 50% of the annotated training data.

Wu et al. [86] used an SVM classifier to recognise activities from a sensor embedded in clothing (smart garment). They have extracted features using DWT and recognised 9 activities. They collected data from 7 participants for training the classifier. The testing data was collected from the same 7 participants and other 6 new participants. They achieved an overall accuracy of 95% for testing data from the participants who are in the training dataset and a 94% overall accuracy for the remaining 6 participants.

The SVM classier has also been used in [40] to recognise activities by identifying motion primitives. Given sensor observation sequence for an activity was segmented using fixed size segmentation. Then statistical features were extracted from these segments and these feature vectors were clustered. Subsequently, the cluster centroids were treated as motion primitives. Given a sensor observation sequence for classification, the approach in [40] segmented the data stream and then extracted statistical features. Then these features were assigned to the closest motion primitive. Subsequently, a histogram of motion primitives for a sensor observation sequence was considered as the feature vector for classification using an SVM classifier. This feature vector construction is similar to the bag-of-words approach used in natural language processing [40]. An experiment has been conducted with data collected from 6 participants performing 9 activities and this approach achieved an overall accuracy of 93%.

Wang et al. [87] attached several RFID antennas and RFID tags on the body to recognise activities. They have extracted features from RSSI to recognise motion patterns using a 5 s sliding window and utilised an SVM classifier. Experiments were conducted using a dataset collected from 4 participants and achieved an overall accuracy of 94%.

Zhang and Sawchuk [73] used a classification technique based on an over-complete dictionary and sparse representation. Initially, they have extracted statistical and frequency domain features using a 4 s sliding window with 50% overlap. Then the feature vectors from each class were arranged into matrices and these matrices were concatenated to construct the over-complete dictionary. Their main intuition here is that any testing feature vector can be obtained using a linear combination of records in the constructed over-complete dictionary. Therefore, the classification was done by recovering the sparse representation using the dictionary and identifying regions of the coefficients with higher values. However, the proposed sparse recovery method is an optimisation problem and hence requires high computation during testing. From an evaluation with a dataset collected from 14 participants (age 30 ± 7 years), the proposed method achieved a 95% accuracy.

Conditional Random Fields (CRF) for activity recognition was used by Shinmoto Torres et al. [46]. They also evaluated several data stream segmentation methods to be used with sparse acceleration data streams from a passive sensor enabled RFID tag. Furthermore, they have proposed a method for real-time inferencing in CRF. From their evaluation using two datasets with 14 older participants (age 75 ± 5 years) (see Section 2.4.2), they were able to achieve accuracies of $78 \pm 7\%$ for the first data set and $95 \pm 4\%$ for the second dataset.

Use of a deep Convolutional Neural Network (CNN) to recognise activities was proposed in [37]. This study also proposed a data stream representation, termed as an *activity image*, that is the Discrete Cosine Transformation (DCT) of sensor observations for a given interval arranged into a matrix. In cases of lower confidence between activity classifications, this approach has utilised a series of binary SVM classifiers trained using hand-crafted features to complement the learnt features from the CNN. The proposed approach in [37] has been evaluated using three publicly available datasets and achieved accuracies over 97%.

The study in [42] utilised a decision tree classifier, particularly a modified version of the Random Forest algorithm, in an application that executes on a smart watch to recognise activities. The modifications to the RandomF orest include replacing CART with C4.5 and splitting criteria selection using both information gain and feature computational cost. Since the overall computational cost involved with having a DT ensemble is higher than a single DT, this approach selected only one DT from the DT ensemble. This selection was made considering both the accuracy and the computational cost of the features. This approach was evaluated using a dataset collected from 3 participants and achieved an accuracy of 96%.

Deep learning to recognise activities using wearable sensors was also considered in [38]. However, unlike the work in [37], a deep CNN with Long Short Term Memory Neural Network (LSTMNN) has been used in [38]. Furthermore, the study in [38] has used raw sensor values instead of transforming them, similar to [37], as input to their classifier. Unlike other neural networks, LSTMNNs can model the sequential nature of the data and their references to the history can be controlled. The approach in [38] was evaluated using two publicly available datasets and depicted better performance.

Ronao and Cho [45] used a CNN classifier to recognise activities using smartphones. Similar to the study in [38], Ronao and Cho used the unprocessed sensor data as input to their classifier. Specifically, they have utilised a 128-sample sliding window with 50% overlap to obtain inputs to the classifier. With an evaluation using a dataset collected from six participants, they were able to achieve 95% accuracy. Furthermore, they were able to successfully classify typically confused activities such as walking-up and walking-down stairs with a high accuracy (99%).

In summary, machine learning methods allow the researchers to analyse data from multiple sensors effectively, which is impractical when devising empirical algorithms. There are references to the use of various supervised learning approaches, such as Naïve Bayes (NB), Decision Tree (DT), Support Vector Machines (SVM), K-Nearest Neighbour, Hidden Markov Models (HMM), Conditional Random Fields (CRF) and Neural Networks (NN). The most notable difference in these methods lies in how they have used machine learning algorithms in their studies. While most of these researchers utilise classification algorithms for activity recognition directly [17, 41], some researchers have incorporated heuristics with machine learning [79, 35, 88]. Furthermore, modifications to existing machine learning classifiers have also been proposed; for instance, in [83, 75] a discriminative learning approach was utilised to obtain observations for their HMM prior to modelling the sequential nature in activities. On the other hand, Shinmoto Torres et al. [46], proposed a real-time inferencing approach for linear chain CRF.

2.2.4 Common Machine Learning Algorithms Used in Activity Recognition

In this section, a brief description of the commonly used machine learning techniques is presented. We denote here the feature vector as $\mathbf{x} \in \mathbb{R}^d$ and corresponding activity as $y \in \mathscr{C}$, where $\mathscr{C}$ is the set of class labels or activities having k number of classes.

Naïve Bayes (NB)

NB is a generative model that assumes features are conditionally independent. It fits a probability distribution on input and output variables using the training dataset and learns the underlying process. Being a traditional machine learning classifier, NB assumes that the training pairs $(\mathbf{x}_i, y_i), i = 1, \cdots, m$ are i.i.d. (independent and identically distributed).

Given an instance $\mathbf{x}$ for classification, the decision rule of this classifier can be given as:

$$\hat{y} = \underset{y \in \mathscr{C}}{\arg\max}\ p(y) \prod_{j=1}^{d} p(\mathbf{x}^j | y). \tag{2.1}$$

where $\mathbf{x}^j$ represents the j_{th} component of the vector $\mathbf{x}$.

In Equation Eq.(2.1), $p(.)$ represents the respective probability functions and they are estimated using a training dataset. This estimation depends on the selected probability distribution function. Typically, for real-valued input, the conditional probability distribution $p(\mathbf{x}^j | y)$ is assumed to be normal.

Support Vector Machine (SVM)

The Support Vector Machine (SVM) is rooted on the structured risk minimisation concept [89]. The standard SVM is a binary classification algorithm, i.e. $y_i \in \{+1, -1\}$. Similar to NB, SVM also assumes that the training data is i.i.d. During training, the

SVM classifier learns a hyperplane that separates the classes with the maximum margin, which leads to a better generalisation performance by solving the following Eq.(2.2) convex optimisation problem.

$$\begin{aligned} & \min_{\mathbf{w},\xi_i} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i \\ \text{subject to:} \quad & \forall i \quad y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1-\xi_i \\ & \forall i \quad \xi_i \geq 0 \end{aligned} \tag{2.2}$$

In Equation Eq.(2.2), **w** is the learnt model (weight vector), ξ_is are slack variables introduced to account for linearly inseparable datasets and C is the penalty for margin violations. Although there are slightly different variations of SVM optimisation problems, for instance, as in [90], all follow the same principle of structured risk minimisation.

Then, the SVM decision function is given by:

$$\hat{y} = \text{sign}(\langle \mathbf{w}, \mathbf{x}\rangle) \tag{2.3}$$

where $\hat{y}$ is the prediction for a feature vector **x**.

The optimisation problem Eq.(2.2) can be solved using its dual form Eq.(2.4).

$$\begin{aligned} & \max_{\alpha_i} \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{ij}\alpha_i\alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j\rangle \\ \text{subject to:} \quad & 0 \leq \alpha_i \leq C \\ & \sum_i \alpha_i y_i = 0 \end{aligned} \tag{2.4}$$

In the dual form Eq.(2.4), the variables α are called dual variables and the number of dual variables are equal to the number of constraints in the primal form Eq.(2.2). The $\mathbf{x}_i$s where the corresponding α_i is non-zero are termed as support vectors and only the support vectors contribute to the final classification model. Thus, classification is obtained by:

$$\hat{y} = \text{sign}(\sum_{i=1}^{l}\alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i\rangle). \tag{2.5}$$

The inner product $\langle \mathbf{x}_i, \mathbf{x}_j\rangle$ here can be replaced by a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$, which is known as the kernel trick. In this context, a kernel function efficiently calculates the inner product between the inputs in a different dimensional space than the input space. Therefore, SVM can achieve non-linear decision boundaries using kernels. Consequently, the resulting hyperplane is a complex decision surface in the input space.

As indicated by Eq.(2.3), SVM primal formulation performs a linear classification in the feature space and model is represented by a single vector. When the dual formulation is used with large training datasets (e.g. see Section 2.4), the number of support vectors tend to increase, and consequently increases the complexity of the classification function Eq.(2.5) when compared with the classification function of the primal form Eq.(2.3). Therefore, primal formulation is more desirable when there are a large number of features.

Decision Trees (DT) and Random Forest (RF)

A Decision Tree is a simple classification model that is built using a hierarchy of if-else rules. In fact, a decision tree partitions the feature space into rectangular regions by iteratively selecting a single feature at a time. During training, the classifier selects the most suitable feature and the position to split the dataset based on different parameters such as information gain and entropy. This splitting is continued, often, until all the data points in a particular region are from a single class or the minimum number of data points in a region is achieved. The commonly used decision tree algorithms are CART [91] and C4.5 [92].

The most notable features in decision tree algorithms are their simplicity and interpretability [93]. However, DT can over-fit easily and reduce the generalisation performance; therefore, small changes in feature values can alter the decision significantly. To overcome over-fitting, techniques such as Bagging [93] are used in developments like Random Forest (RF) [94].

In RF [94], a number of decision trees (DT), B, are trained using subsets of data sampled uniformly with replacement from the training dataset, D (i.e. bagging). In RF, randomly selected subsets of features are evaluated at each branch to decide the feature to be used for partitioning the subset of the data at the corresponding node (i.e. splinting criteria), as opposed to evaluating the entire set of features as in traditional DT learning settings. This training procedure results in a large collection of *de-correlated* trees. The output from RF is obtained using the majority vote of all the trees in the DT ensemble; thus RF achieves a higher generalisation performance than a single DT based on the concept of the law of large numbers.

Neural Networks(NN)

The cornerstone of neural networks is the perceptron model presented by Rosenblatt [95] that mimics the single neuron in the brain. Perceptron is a linear model which is essentially represented using a hyperplane in its input space. NNs initially gained popularity in the 1980s and 1990s. NN is essentially a layered arrangement of neurons or units where the first layer is the input layer and the last layer is the output layer. In between the input and output layers, there can be one or more hidden layers with varying number of units. In classification setting, the output layer has k number of outputs, and units in the upper layer are connected to units in the lower layer to form a network. In NN terms, the arrangement of these units is termed the architecture of the network. In a neural network, edges connecting nodes are associated with a weight and the network architecture, together with these weights, define the neural network model. Using complex network architectures neural networks can model sophisticated functions for classification [93]. Although neural networks perform well in practice, there is no theoretical framework as for other classifiers such as NB and SVM.

The neural networks are trained using back-propagation using gradient descent methods [93]. Selection of an architecture is significant when using neural networks and this is usually performed by experimentation with several architectures.

Hidden Markov Models (HMM)

The hidden Markov model is a generative probabilistic graphical model that captures the sequential nature of observations. In HMM, it is assumed that the discrete state of the model is hidden and it is visible through the observations. It is also assumed that, the hidden state makes a transition at each time step. An HMM model, λ with N states and M observation symbols is defined by its transition probabilities (A), set of observation probabilities (B) and the initial state distribution (π):

$$\lambda = (A, B, \pi). \tag{2.6}$$

Given a sequence of observation $\mathbf{X} = \{\mathbf{x}_t\}_{t=0}^{T}$, HMM training considers adjusting the parameters in λ to maximise the probability of $P(\mathbf{X}|\lambda)$. This is usually performed using the Baum-Welch algorithm [96]. Once the model parameters are determined, given a sequence of observations the most likely state path can be obtained efficiently by using dynamic programming techniques, particularly using the Viterbi algorithm [96].

Conditional Random Fields (CRF)

CRF is a probabilistic graphical model and is based on the conditional probability distribution of exponential family Eq.(2.7) [31]. Being a graphical model, CRF can naturally model the sequential nature of activities. Given a sequence of feature vectors, $\mathbf{X} = \{\mathbf{x}_t\}_{t=0}^{T}$, CRF predicts the corresponding sequence of labels, $\mathbf{Y} = (y_t)_{t=0}^{T}$.

$$\begin{aligned} p(\mathbf{Y}|\mathbf{X};\lambda) &= \frac{1}{Z(\mathbf{X},\lambda)} exp\left(\sum_j \lambda_j \sum_{t=1}^{T} F_k(y_{t-1}, y_t, \mathbf{X})\right) \\ Z(\mathbf{X},\lambda) &= \sum_{\mathbf{Y}} exp\left(\sum_j \lambda_j \sum_{t=1}^{T} F_k(y_{t-1}, y_t, \mathbf{X})\right) \\ \hat{Y} &= \underset{\mathbf{Y}\in\mathscr{Y}}{\arg\max}\ p(\mathbf{Y}|\mathbf{X};\lambda) \end{aligned} \tag{2.7}$$

In this equation, F_ks are feature functions, λ is the weight vector (model) and $Z(\mathbf{X},\lambda)$ is a normalising constant. Feature functions F_k are used to capture the sequential nature of the activities by correlating previous, $(\mathbf{x}_{t-1}, y_{t-1})$, and current, $(\mathbf{x}_t, y_t)$, training samples in a sequence.

2.3 RFID and Passive Sensor Enabled RFID Tags

We mainly focus on using data collected from sensor tags for HAR. As mentioned in Section 1.1, sensor tags are light-weight and small compared to battery powered sensors and hence expected to be unobtrusive, particularly for older people. In this section we briefly discuss the Radio Frequency Identification (RFID) technology and discuss passive sensor enabled RFID tags, particularly the Wireless Identification and Sensing Platform (WISP).

2.3.1 RFID Preliminaries

Radio Frequency Identification (RFID) technology is used to identify objects uniquely [97, 98]. An early form of RFID technology dates back to the second world war when British researchers invented a transceiver that sent back an identification signal when hit by radar [97, 98]. The widespread commercial adoption of RFID technology was due to the establishment and standardisation of the Electronic Product Code (EPC). Organisations initially employed barcode labels to store product-related information but these labels require line-of-sight access. Therefore, barcode labels are not possible to read when they are covered with other materials or with dirt. The RFID technology,

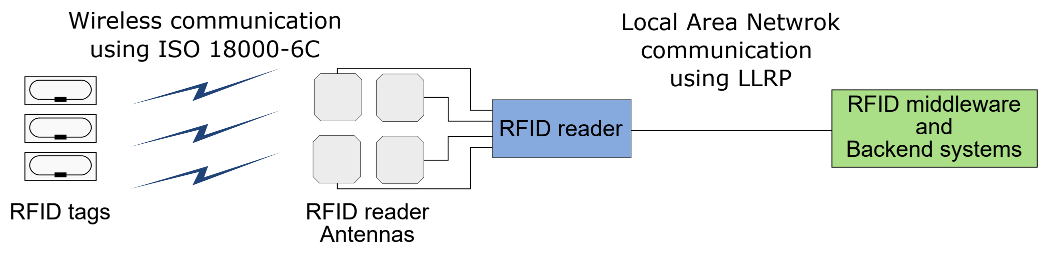

Wireless communication using ISO 18000-6C
Local Area Netwrok communication using LLRP
RFID reader
RFID middleware and Backend systems
RFID tags
RFID reader Antennas

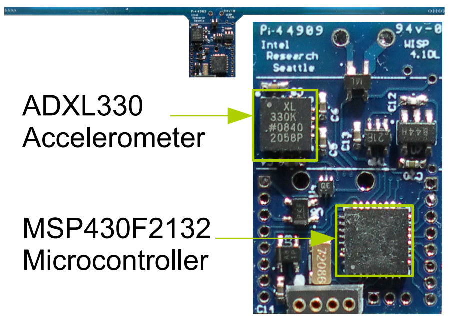
Fig. 2.2 A typical RFID system indicating the wireless communication between RFID tags and he RFID reader using ISO 18000-6C protocol, and RFID reader communication with the backend systems over local area network using Low Level Reader Protocol (LLRP)

in the form of RFID label tags, was seen as an alternative to barcode labels due to the unseen discoverability of RFID tags [97, 99].

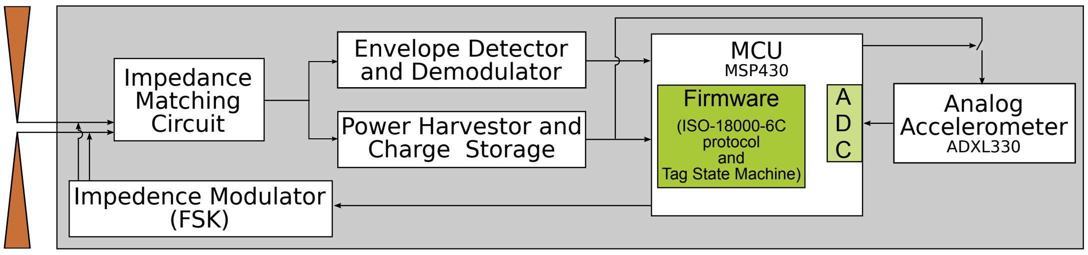
A typical RFID system consists of several components that include: i) RFID tags; ii) RFID readers; iii) RFID reader antennas; and iv) backend systems. Figure 2.2 illustrates how these components are arranged in an RFID system. Modern RFID tags usually contain a non-volatile memory, often an Electrically Erasable Programmable Read Only Memory (EEPROM) that can store the ID of the tag and other information specified by the tag user. The portion of the tag memory where the user specified data is stored is commonly referred to as *user memory*. The information from the RFID tags are captured by RFID antennas and RFID antennas are connected to RFID readers. RFID readers can query RFID tags and obtain the information stored in the tag memory if required. RFID antennas also transmit Radio Frequency (RF) signals and this transmission is controlled by the RFID reader. The backend systems obtain the information from the RFID tags by communicating with RFID readers. In the context of this thesis, we mainly focus on RFID technology that uses the Ultra High Frequency (UHF) radio spectrum.

It is important to note that in RFID technology, an RFID reader initiates the communication between the RFID tags in its field of view, which is defined by the number and the arrangement of the RFID antennas connected to it. This communication is governed according to the ISO 18000-6C protocol [26]. This is a random access protocol where an RFID reader first singulates a specific RFID tag in its field of view and then acquires the tag ID to identify the tag uniquely and hence the object to which the RFID tag is attached. If backend systems require information on the tag memory, they are obtained by instructing the RFID reader to access a designated memory location of the RFID tag as defined in the ISO 18000-6C protocol.

There are three main types of RFID tags: i) passive RFID tags; ii) active RFID tags; and iii) semi-passive tags. The passive RFID tags harvest power from the RF energy radiated by the RFID antenna and usually send data back to the RFID reader



(a)


Impedance Matching Circuit
Envelope Detector and Demodulator
Power Harvestor and Charge Storage
MCU
MSP430
Firmware
(ISO-18000-6C protocol and Tag State Machine)
A D C
Analog Accelerometer
ADXL330
Impedence Modulator (FSK)

(b)

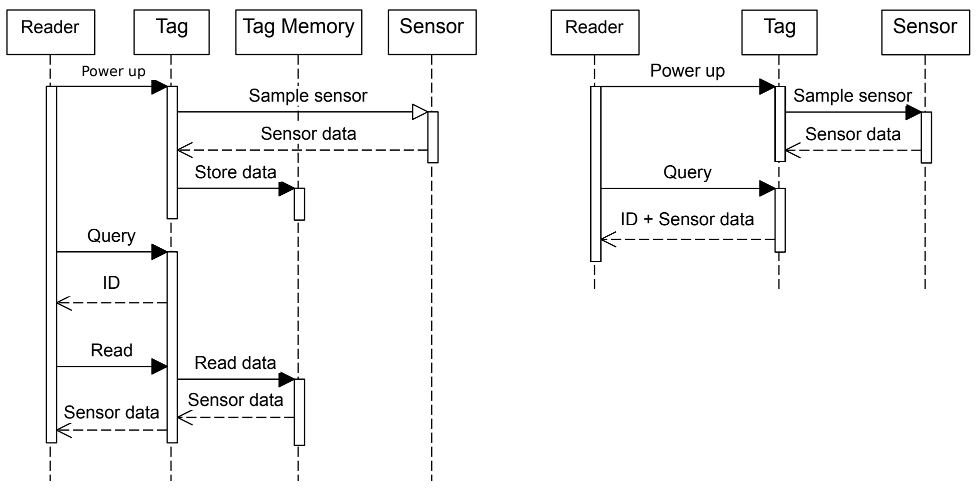
Fig. 2.3 (a) WISP indicating the MSP430F2132 microcontroller (MCU) and ADXL330 3-axes accelerometer; (b) Block diagram of the WISP with an accelerometer

using a mechanism known as backscattering [99]. Unlike transmitting data that require a considerable amount of power, backscattering requires less power as it reflects the incoming RFID signals. In contrast, active RFID tags have a battery and transmit signals back to the RFID reader antenna. Therefore, active RFID tags can communicate over larger distances. Semi-passive tags combine the best of both worlds where they contain a battery that powers the circuitry but backscatter to send data back to the RFID reader, similar to a passive RFID tag. These tags have a higher operational range than passive RFID tags and a higher operational life than active tags.

2.3.2 Passive Sensor Enabled RFID tags

Passive sensor enabled RFID tags or sensor tags follow the principle of passive RFID tags, i.e. they operate using the power harvested from the incident RF energy transmitted by RFID antennas. We utilised a WISP[1] (Wireless Identification and Sensing Platform) [23] tag, which is a passive sensor enabled RFID tag embedded with a 3D accelerometer (ADXL330), illustrated in Figure 2.3. Unlike other passive RFID tags, WISP is equipped with a general purpose low power micro-controller unit (MSP430) which implements the ISO 18000-6C protocol to communicate with the RFID Reader.

[1]Details of the WISP can be obtained from: https://wisp.wikispaces.com



(a) Reading from tag memory (b) Embedding within ID

Fig. 2.4 Sensor data acquisition approached from passive sensor enabled RFID tags

There are two main approaches for sensor data acquisition from sensor tags: i) reading sensor data logged in non-volatile tag memory; and ii) directly embedding sensor data within a tag Identifier (ID). Figure 2.4a abstracts the operation of the first approach where the sensor data is first stored in the tags' non-volatile memory and then read, similar to reading user data stored in a regular RFID tag using the `READ` command defined in ISO 18000-6C [26] after singulating the tag using the `QUERY` command. As also discussed in [28, 100], this approach requires a considerable amount of power and incurs delays as data needs to be written to and read from a non-volatile memory. The second approach is to embed the sensor data in a tag ID. Although embedding sensor data in a tag ID sacrifices the range of the unique tag identifiers, this approach eliminates the energy intensive and time consuming operation of writing sensor data to the tags' non-volatile memory as shown in Figure 2.4b. Therefore, sensor tags are expected to have a longer communication range and a higher data rate compared with the first approach. Similar to the first approach, in the second approach, the sensor is sampled when adequate energy is harvested but is kept in the volatile memory of the tag. During the next inventory round, which is marked by a `QUERY` command, the sensor tag sends sensor data by embedding them in the tag ID.

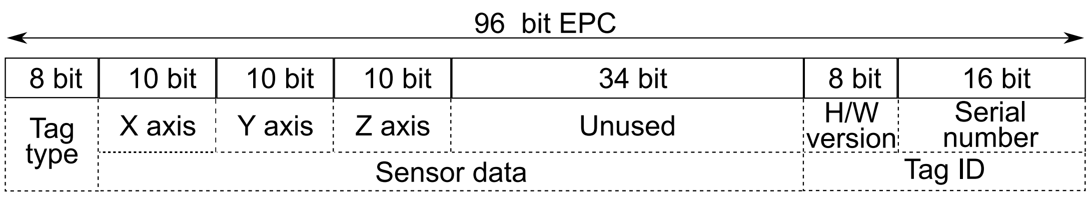


Fig. 2.5 96-bit EPC tag ID format used to acquire sensor data

Table 2.1 Summary of the datasets

Dataset	Number of participants	Age	Number of Antenna
HYA	10	26.4 ± 2.12	4
HOA	14	74.6 ± 4.9	4 (Room1) and 3 (Room2)
FOA	26	84.2 ± 5.2	3

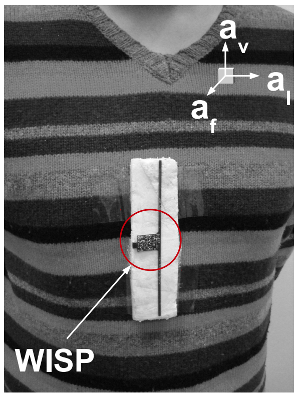
Figure 2.5 illustrates the 96 bit Electronic Product Code (EPC) tag ID format re-defined for sensor data acquisition. The EPC here is logically partitioned into 3 sections: i) Tag type (8 bits); ii) Sensor data (64 bits); and iii) Tag ID (24 bits). The *Tag type* is used to identify the type and hence capabilities of the tag. The *Sensor data* section is used to embed acquired sensor data. To embed acceleration data as shown in Figure 2.5, only 30 bits in this section are utilised because each acceleration axis was sampled at 10-bit resolution. The *Tag ID* section is composed of the WISP hardware version and the tag serial number, and a WISP can be uniquely identified using this *Tag ID*. Using this approach, when the WISP is adequately powered, a data stream with an upper bound sampling rate of 40 Hz can be obtained.

The 5-tuple $[a_f, a_l, a_v, RSSI, aID]$ was obtained from each datum sent by a WSIP and received by a reader, where a_f, a_l and a_v represent frontal, lateral and vertical (longitudinal) accelerations measured with respect to the acceleration sensor (Figure 2.6a), the *RSSI* (Received Signal Strength Indicator) represents the strength (power) of the radio signal of a sensor observation sent by the sensor and received by a specific antenna and measured by an RFID reader, and *aID* represents the identifier of the antenna that captured the observation.

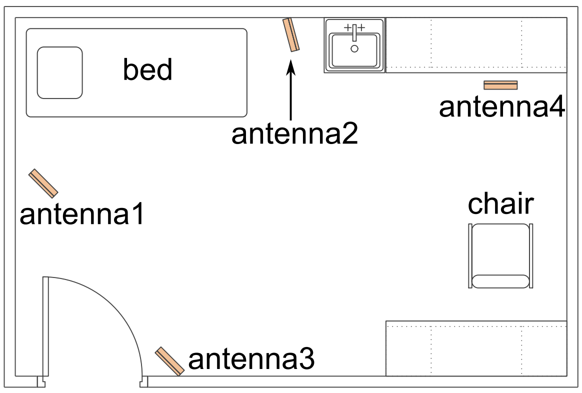
2.4 Datasets Used in This Thesis

In this section, we present the datasets collected using the sensor tags to recognise activities leading to falls. Three datasets from different populations are used: i) healthy young adults [61] (HYA); ii) healthy old adults [46] (HOA); and iii) hospitalised old adults [101] (FOA). A summary of the datasets is listed in Table 2.1 and described in detail in the following sections. These datasets are publicly available in the project web site[2]. The datasets were collected and annotated by Mr Roberto Shinamoto Torres with the help of Dr Shailaja Nair and Mr Stephen Hoskins.

[2]http://autoidlab.cs.adelaide.edu.au/research/ahr



(a)



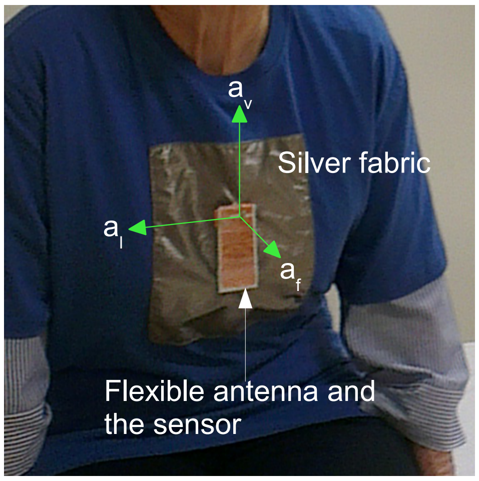
(b)

Fig. 2.6 (a) The body-worn WISP indicating the acceleration sensor reference frame; (b) The clinical room setting and the antenna placement used in the HYA dataset

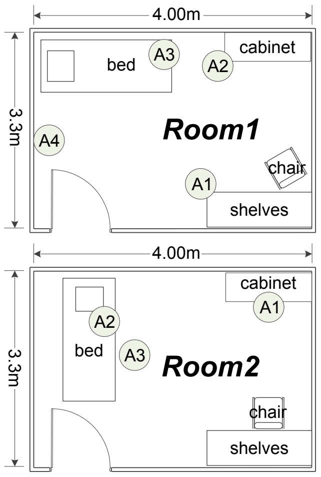
2.4.1 Healthy Young Adult Dataset (HYA)

Ten healthy young volunteers aged between 23 and 30 years (mean 26.4 $\pm$ 2.12 years) were recruited to participate in this study. This study was reviewed by the Human Research Ethics Committee TQEH/LMH/MH and no ethical matters of concern were identified. Each volunteer wore a WISP over a garment at the sternum level (Figure 2.6a). The data collection procedure was carried out in a clinical study room (Figure 2.6b), furnished with a hospital bed and a chair, at the Basil Hetzel Institute, Woodville, South Australia. To energise and acquire data from the WISP four circularly polarised RFID antennas were used. As illustrated in Figure 2.6b, these antennas were configured to illuminate the entire room.

Three activity routines were defined: i) getting into bed; lying on bed and getting out of the bed; ii) walking (between the bed and the chair); and iii) sitting down and/or getting up from a chair. Each participant performed three separate activity scripts where each activity script was obtained by random ordering of activity routines defined previously. Thus, each trial contained data collected from a single volunteer using a selected activity script. Activities were recorded and annotated while activities are performed by an observer using an in-house software tool. This dataset contains five ground truth labels: i) sitting-on-bed; ii) lying-on-bed; iii) standing; iv) walking; and v) sitting-on-chair. The during annotation, the moment at which activity transition occurs demarcated the end of the current activity. For instance, end of sitting-on-bed is demarcated by transition to lying on bed or standing. Annotation errors have been documented and corrected off-line.



(a)



(b)

Fig. 2.7 (a) An older person wearing the W^{2}ISP sensor (≈3 grams, size 18×20×2 mm), flexible antenna (36×85×2 mm) and isolating silver fabric (230×220 mm); (b) *Room1* and *Room2* room and equipment configuration. All ceiling antennas (A3 in *Room1* and A2 and A3 in *Room2*) were inclined towards the middle section of the bed.

2.4.2 Healthy Older Adult Dataset (HOA)

This dataset was collected using the Wearable WISP (W^{2}ISP) (see Figure 2.7a) which uses a smaller antenna designed for wearable applications [24]. The study protocol (protocol number 2011129) was reviewed and approved by Human Research Ethics Committee of The Queen Elizabeth Hospital, South Australia. For this study, the participants had to be 65 years or older, living at home, able to consent to the study and mobilise independently. The participants were recruited from geriatric clinics and from volunteer lists from other studies. First, potential participants were contacted over the phone and then participation information sheets were mailed. During the trial informed consent was obtained from the participants and no honorarium was paid. Fourteen participants responded and volunteered to participate in the study. The study was completed over a two-month period. During the trial, participants were informed what types of activities they were required to perform and the worn sensor was used to monitor these activities. Similar annotation process described in Section 2.4.1, a researcher who was present during the trial annotated the sensor data using an annotating software developed to record sensor data. A trial with each volunteer lasted 60 to 90 minutes.

Data collection was carried out in two different clinical room settings (*Room1* and *Room2* illustrated in Figure 2.7b) that differed in layout, antenna placement and number of antennas deployed. These room settings were designed to investigate a general hardware deployment option (i.e. antenna placement) suitable for hospital ward rooms furnished with a bed and armchair. *Room1* antenna deployment was designed to illuminate the whole room. In contrast, *Room2* deployment was designed to only illuminate areas where patients are likely to spend the most amount of time. In both room configurations, RFID antennas are placed above the level of the bed and at or near ceiling height to reduce possible obstructions of interrogation signals from RFID reader antennas and responses from the W^{2}ISP. Similar to HYA dataset, circularly polarised antennas have been used in the data collection.

During the study, participants performed activities which included: i) lying on the bed; ii) sitting on the bed; iii) getting out of the bed; iv) sitting on the chair; v) getting out of the chair; and vi) going from *A* to *B* (*A* and *B* represent the bed, chair or door). Each participant performed activities on two broadly scripted activity lists. No particular order was used for selecting the scripts and the number of scripted routines. No instructions on how to perform activities were given to the participants.

Data from 10 participants were collected from the *Room1* deployment and *Room2* contained data from 5 participants. One participant contributed to both datasets. Due to a malfunctioning of the sensor prototype, data from the common participant from *Room1* was not usable, hence ignored. Data collected from *Room1* and *Room2* configurations are simply referred as *HOA-Room1* and *HOA-Room2* respectively.

2.4.3 Hospitalised Older People Dataset (FOA)

This dataset included data collected from 26 hospitalised inpatients, with a male to female ratio of 9:17, in the geriatrics ward of the Queen Elizabeth Hospital in Adelaide, South Australia. Participants were recruited from the hospital and no incentive or honorarium was paid for participation. Only participants with no cognitive impairments, who can ambulate with or without a walking aid were selected for this trial.

The FOA dataset was also collected using the W^{2}ISP, as in the healthy older participant dataset (Section 2.4.2). Participants wore the W^{2}ISP at their sternum level over their hospital garment. Data collection was carried out in their hospital room (see Figure 2.8) from 2 pm to 4 pm. Some patients are admitted into rooms with two beds and some were in single bedrooms. As shown in Figure 2.8, the patient monitoring area was equipped with three circularly polarised RFID reader antennas. This configuration is similar to *Room2* in Figure 2.7b. Antenna 1 was placed approximately 1.7 m from the ground and facing towards the chair. Antenna 2 and Antenna 3 were placed near the ceiling on top of the bed, approximately 2.6 m and 2.7 m from the ground respectively.

Patients performed sets of broadly scripted activity routines that included: i) walk to the chair; ii) sit on the chair; iii) get out of the chair; iv) walk to the bed; v) lie on

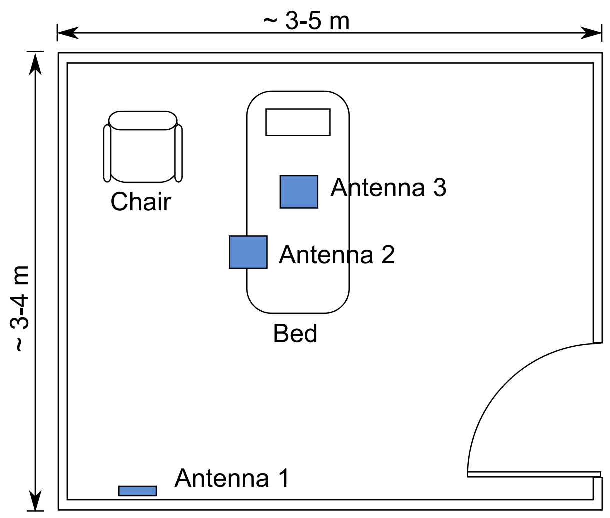


Fig. 2.8 Approximate hospital room configuration and layout

the bed; vi) get out of the bed; and vii) walk to the door. Patients were not instructed how to perform these activities and the researchers allowed them to carry out these activities at their own pace. They were also instructed to request to withdraw from the study in the event of discomfort or distress. Each trial last about 20 to 25 minutes and participants repeated the trial three to five times, depending on their comfort level. During the trials, one of the two research team members who were present asked them to perform the activities from the activity scripts while the other researcher annotated the activities as and when they were performed (ground truth) which is required for evaluation using a software tool developed to capture sensor data. Similar to the annotation process described in Section 2.4.1, start of an activity transition was used as the demarcation of the end of the current activity.

It is observed that during the trials patients' head-rests on the beds were kept either flat or inclined as they would have the head-rests when watching television or reading. Furthermore, while patients were sitting on the bed, they did not keep their legs straight. In addition, all the patients sat on the bed with their legs off the bed. Therefore, multiple posture variations have been captured in this dataset.

2.4.4 Characteristics of the Datasets

In this section, characteristics of the three datasets described above are presented. Table 2.2 shows the statistics of these datasets. From this table we can observe that for all three datasets there is a considerable variation in the inter-sensor observation time differences (time difference between consecutive sensor observations), as indicated by the higher standard deviations; i.e. data streams from sensor tags are sparse. Furthermore, when considering the inter-sensor observation time differences whose

Table 2.2 Dataset statistics

	HYA	*HOA-Room1*	*HOA-Room2*	*FOA*
Mean of the inter-sensor observation time difference (s)	0.18 (0.32)	0.36 (2.43)	0.79 (9.62)	0.82 (6.09)
Mean of the inter-sensor observation time differences considering the values up to 95th percentile (s)	0.14 (0.10)	0.20 (0.26)	0.31 (0.26)	0.36 (0.34)
Length of data per participant (s)	495 (253)	2116 (1006)	3596 (268)	763 (219)

Corresponding standard deviation values presented within parenthesis.

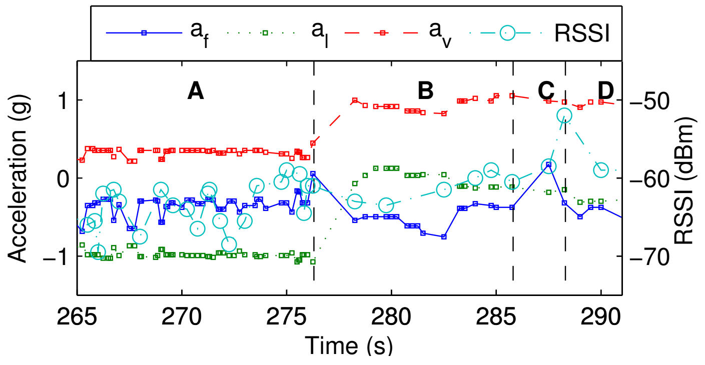


Fig. 2.9 Acceleration and RSSI (for A3) signal patterns collected from W^{2}ISP for *HOA-Room2* when a participant got out of the bed from lying on the bed (lying on side) (A: lying-on-bed; B: sitting-on-bed; C: standing; and D: Ambulating). This also indicates the sparsity (variable inter sensor observation time differences) of sensor data streams.

values are less than or equal to the 95th percentile of the values, the mean and standard deviation values are considerably less compared with the entire set of values. For instance, in the case of HOA-Room2 and FOA datasets, mean values have been reduced to less than half and the corresponding standard deviations are reduced significantly. This indicates that, at times, there are significant intervals with no data from the sensor tags.

The sparse nature of sensor tag data streams are illustrated in Figure 2.9 which shows a data stream segment collected from *Room2* of healthy older adults (Section 2.4.2). From this figure, we can observe that while the participant is lying on the bed, the sensor tag is responding frequently compared with other activities. Furthermore, we can see that for near the boundary between lying-on-bed and sitting-on-bed (approximately 276 s), there are no sensor observations for a period of approximately 2 s. This sparsity is mainly due to two reasons.

First, the ability to sample inbuilt sensors and send data back to the RFID reader depends on the successful powering and reading of the sensor tag; this is affected by several factors. The increase of the distance between the RFID antenna and WISP reduces the strength of the incident radiation on the sensor tag causing less power to be harvested. In fact, the signal strength indicated by *RSSI* at a point distance, d away of an RFID antenna is $RSSI \propto 1/d^4$ [99, 102]. Since circularly polarised antennas have been used during data collection, the influence on the antenna polarisation on power harvesting with the change of RFID tag antenna orientation relative to the RFID reader antenna was expected to be negligible. Furthermore, occlusion of the radio signals from radio frequency opaque objects such as a human body can cause radiation from the RFID reader antenna to be absorbed. This not only reduces the incident RF energy on the tag but also reduce the strength of the backscattered signal hindering its detection at the RFID reader antenna. In such instances, readability of the sensor tag will be severely reduced or at times will not be able to be read at all. Power harvesting is also hindered by destructive interference caused by the multipath effect and radio signals from other devices that use the same radio spectrum (920-926 MHz in Australia). As a result, regions where RFID tags cannot operate are formed.

Secondly, the data acquisition from passive RFID sensors is non-deterministic due to the use of the ISO 18000-6C protocol [26]. As mentioned earlier in Section 2.3.1, the communication is initiated by the RFID reader. Therefore, even while the tag is adequately powered, the data streams from sensor tags have variable inter sensor observation time differences.

2.5 Conclusion

In this chapter, we looked at Human Activity Recognition (HAR) and commonly used machine learning algorithms used in HAR. We further presented a brief description of the RFID technology, focusing on the passive sensor enabled RFID tags. Finally, detailed descriptions of the datasets that have been collected from the sensor tags used in this thesis were presented.

One of the main considerations in HAR is the selection of the sensor deployment strategy; environmental sensor deployment and body-worn sensor deployment. As explained in Section 2.2.1, although deploying sensors in the environment is less intrusive to the people being monitored, monitoring can only be performed where the sensors have been deployed. Furthermore, this method poses further challenges to uniquely identify individuals when the monitoring area is occupied by multiple residents. In addition, the use of cameras as environment sensors have raised privacy concerns. In contrary, the body-worn sensor deployment is not limited by the monitoring area. Furthermore, as sensors are attached to the bodies of people being monitored, identification of individuals is a trivial task. According to the previous literature on body-worn sensor based HAR, most of the studies consider the use of bulky battery powered sensors, which are obtrusive, particularly for older people. In order to take advantage of the favourable aspects of body-warn sensors and to minimise the adverse effects, particularly wearing discomfort, we specifically selected the passive sensor enabled RFID tag (i.e. sensor tag) technology to acquire human motion. As sensor tags are light-weight, small and requires no maintenance, it is expected that they are more comfortable to wear compared to the existing battery powered sensors.

In contrast with the sensor data streams from battery powered sensors, as explained in Section 2.4.4, data streams from sensor tags are noisy and intermittent. Furthermore, occasionally, data from sensor tags can be unavailable for longer durations. This is attributed by the power harvesting nature of sensor tags and RFID communication protocol. Therefore, these unique characteristics pose challenges for accurate activity recognition based on sensor tags.

In this thesis, our main focus is to devise methods to recognise activities using sensor tag data streams by overcoming the aforementioned challenges in the data stream to utilise the positive aspects of passive sensor enabled RFID tags, particularly in the domain of ambulatory monitoring of older people.

Chapter 3

Sensor Data Acquisition from Passive RFID sensors

3.1 Introduction

Previously, in Chapter 2, we highlighted that the research based on wearable sensors mostly uses large battery-powered sensors which are uncomfortable for the wearer. Therefore, in this thesis, we consider the use of passive sensor enabled RFID tags, which are more comfortable for the wearer as they are lightweight and small mainly due to not requiring a battery.

Sensor tag data streams are unique because sensor data are embedded in the tag ID, such as the EPC (Electronic Product Code)[27] (see Section 2.3.2). Sensor tags employ a *query-only approach* to sending sensor and ID data where a *Query* command for a tag ID results in reading the ID from the sensor tag's memory and acquiring sensor data and embedding in the ID (Figure 2.4b). The alternative is to *Write* sensor data to memory and subsequently, acquire the data by a *Query* and a *Read* command [103](Figure 2.4a). However, this approach consumes more power (148 μW for writing vs. 10 μW for reading [104]) and degrades performance because: i) the nature of radio wave propagation (i.e. Friis Transmission Equation [102]) implies that tags that consume more power must be activated at shorter ranges; ii) higher power consumption results in slower sensor data update rates (i.e. duty cycling); and iii) the two round trip times required to acquire sensor and ID data results in sensor data acquisition delays. Therefore, new middleware (see Figure 2.2) is needed to process data from sensor tags using the lower power *query-only approach* to: i) maximise the read range of sensor tags; and ii) maximise the rate at which the tags can stream data back to readers [23].

Implementing middleware for managing sensor tag data requires addressing several key issues. First, an efficient mechanism is required for discovering a sensor tag's capability (e.g. types of embedded sensors) so that: i) data can be extracted and transformed correctly from a tag's ID; and ii) appropriate filtering and aggregation operations can be performed on the sensor data (as determined by the sensor data type such as temperature or acceleration). Secondly, an application agnostic sensor data and ID data subscription specification and a data model for reporting high level sensor tag data (e.g. average temperature) to client applications are needed.

Currently for RFID, among others, both commercial [105–107] and open-source [108–111] middleware are available. A number of middleware platforms [105, 112, 113] have the capability to gather data from wireless sensor networks where sensor and ID (the unique object identifier) data streams are processed separately as they are generated from different sources (e.g. RFID tags and wireless sensor networks). However, the applications that receive data (ID and sensor data) still have the challenge of associating sensor data (such as ambient sensors in the environment) with unique objects (such as seafood cases in a cold chain management application). For example, in the middleware proposed by Bade and Lamersdorf [112], sensor information is processed to identify unacceptable environmental states and subsequent RFID reader scans identify the objects possibly in the undesirable state. Conversely, in Gama et al. [111] sensor data are polled from a sensor network upon receiving events from RFID readers and are presented as a set of environmental information that corresponds to all the observed tags. The middlware proposed in Wang et al. [110] reported sensor and ID data separately and the data association is derived based on the sensor location. In fact, in the case of sensor tags, sensor data and ID data integration is at the hardware layer (since the sensor is on the same platform as the ID carrier). Therefore, these approaches are not suitable for sensor tags due to differences in low level data collection and high level data reporting, and sensor and ID data integration. Furthermore, the middleware developed is not generic but application specific.

Also, from the perspective of application development and interoperability, it is important to abstract from implementation and provide a technology and implementation agnostic specification for subscribing to and reporting of high level events. Although the standardised Application Level Event (ALE) reporting specification [27] has been extended by Gama et al. [111] and Wang et al. [110] for sending sensor events from sensor networks, they do not integrate ID and sensor data at the middleware level and are reported as separate high level events.

Although some existing middleware support sensor data acquisition from sensor tags [105–107], they do not support the query-only approach, instead supporting the approach shown in Figure 2.4a. However, these approaches are not agnostic to

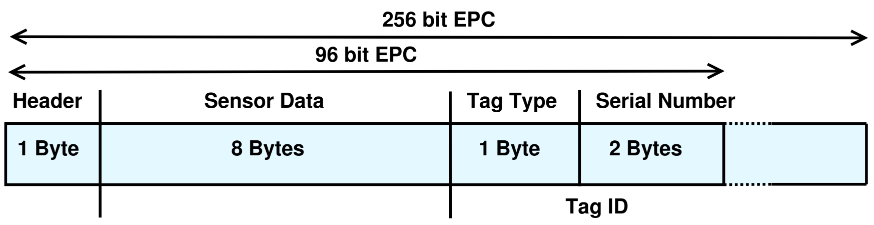


Fig. 3.1 Proposed tag format for 96-bit and 256-bit Electronic Product Codes

tag type (ID tag, sensor tag) since the tag must be first singulated and subsequently discovered to be a sensor tag (by way of a lookup, as in Igarashi et al. [114]) prior to acquiring sensor data from a tag's memory. Furthermore, as highlighted earlier, using the approach in Figure 2.4a is detrimental to the performance of a sensor tag in terms of power consumption and time taken to obtain sensor data.

Therefore, existing middleware have no support for data acquisition from sensor tags that follows the query-only approach. Most of the middleware fail to integrate ID and sensor data and report them as a single high level event. Additionally, other middleware solutions are limited by their application specific nature. Consequently, we were motivated to design and implement a middleware that defines a simple, flexible and extensible tag data model for both RFID tags and sensor tags, that supports the query-only approach and provides a generic framework for implementing, collecting, filtering, aggregating and reporting of both ID and sensor data.

In this chapter, we develop WINDWare (**W**ireless **I**dentification a**n**d Sensor **D**ata Management Middle**ware**), a generic middleware framework that not only addresses the lack of a middleware for simultaneously managing both RFID tag and sensor tag data but also addresses the above challenges to facilitate application development in ubiquitous computing based on sensor tags. Additionally, WINDWare will accelerate the adoption of sensor tags such as the WISP. Furthermore, our middleware conforms to existing standards because we extended the Application Level Event (ALE) interface of the EPCglobal architecture framework as proposed in [103] to facilitate subscription and reporting of data. The work presented in this chapter has been published in the IEEE International Conference on RFID proceedings [28].

3.2 Middleware Architecture

3.2.1 Proposed Tag Data Format

We propose a tag ID format (shown in the Figure 3.1) for sensor tags based on that used for WISPs [23]. Our proposed format uses the Electronic Product Code (EPC) defined in the Tag Data Standard [27]. The **Header** field is set to `0x3D` (an EPC header

Table 3.1 Definition of RFID standards related terms

Term	Description
Event Cycle	Smallest unit of interaction with the ALE implementation during which the ALE interface implementation (middleware) interacts with one or more Readers on behalf of an ALE client
ECSpec	Event Cycle Specification; data element defined in the ALE API which defines the parameters for **ECReport**, **ECReportSpec**
ECReport	Event Cycle Report; reports tag information for the current event cycle as specified in corresponding **ECReportSPec**
ECReportSpec	Event Cycle Report Specification; provides the content definition for the **ECReport**: readers, event cycle duration, and tag filtering
ECReportOutputSpec	EC Report Output Specification; specifies what information on the final set of EPCs in **ECReport** is reported
ROSpec	Reader Operation Specification; specifies reader operation parameters (e.g. identifier, the boundary specification, priority) and configurations such as start and end triggers, and antenna configurations
ROReportSpec	Reader Operation Report Specification; appears as a sub-element of **ROSpec** which describes the contents of the report sent by the Reader and defines report trigger events
ROBoundarySpec	Reader Operation Boundary Specification; defines the span of the operation (indicating the start trigger and stop trigger)
ROSpecStartTrigger	Appears as an element within **ROBoundarySpec**; defines the trigger event for the reader to initiate report generation

Source: EPCglobal Inc, "Epcglobal ratified standards." [Online]. Available:http://www.gs1.org/gsmp/kc/epcglobal/ [27]

currently unused and reserved for future use) to identify a sensor tag. This allows the middleware to process the sensor data in addition to the ID data, as well as manage existing EPCs from ID tags. The content and the format in the **Sensor Data** section depend on the tag type. Proposed tag type definitions are based on WISP tag types.[1] A combination of the **Tag Type** and **Serial Number** are used to identify each sensor tag uniquely. Although the serial number is two bytes for a 96-bit EPC, using a 256 bit EPC can significantly increase the range of serial numbers, which is essential for large scale sensor deployments. Therefore, for a specific tag, only the content of the sensor data field is variable.

3.2.2 Architecture Overview

For seamless integration with existing applications, we base our architecture on the EPCglobal architecture [27]. Figure 3.2 illustrates the system's architecture of WINDWare. Below, we discuss how our architecture meets a number of key requirements identified for RFID middleware in [115] as well as the specific processing requirements of sensor tags we identified in Section 2.4.4.

[1] see: https://wisp.wikispaces.com/Working+with+WISP+firmware

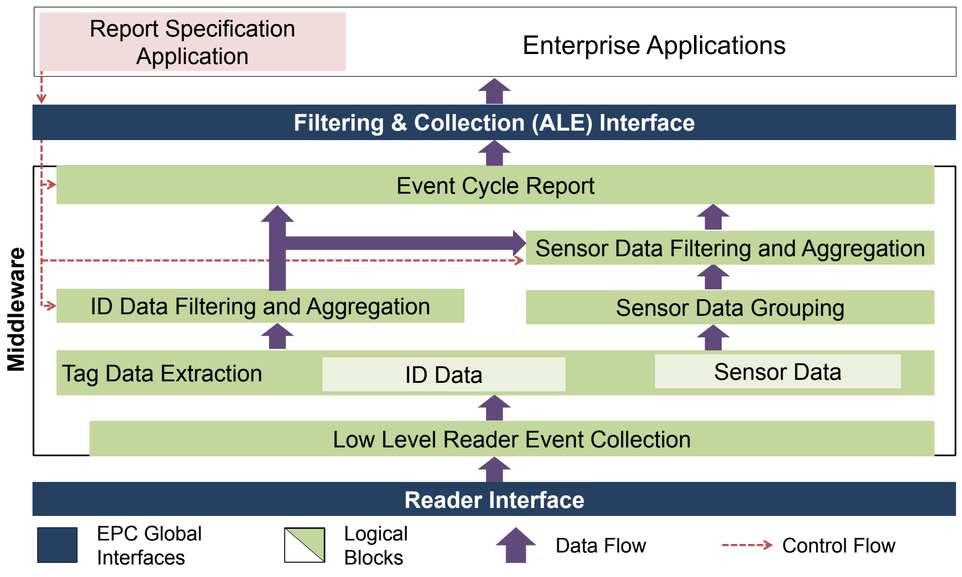


Fig. 3.2 Sensor tag data management middleware architecture

Low level reader event collection A standard reader interface (such as the Low Level Reader Protocol [27]) is used to collect tag reads from physical devices (e.g. RFID readers).

Tag data extraction In the event of an ID tag no additional operations are required, however, in the event of a sensor tag, data embedded in the EPC (Figure 3.1) must be removed and the tag ID is reconstructed without the sensor data, which allows the treatment of a sensor tag as a regular RFID tag. The sensor data and ID data are then integrated into a single data structure defined by the Tag Data Model (see Section 3.3.1) to further maintain their association.

ID data filtering and aggregation ID data are intended to be used for unique identification of items. Based on the reader configuration, reader may report a tag more than once in a single event cycle. This module performs filtering (e.g. removal of duplicate IDs) and aggregation operations (e.g. count products of the same category as opposed to reporting tag IDs) specified by the user based on ID level data.

Sensor data grouping Since it is possible to have multiple sensor readings from the same sensor tag, it is necessary group the sensor readings by **Tag ID**. Filtering is not performed here to remove duplicates because this process can eliminate potentially crucial sensor values.

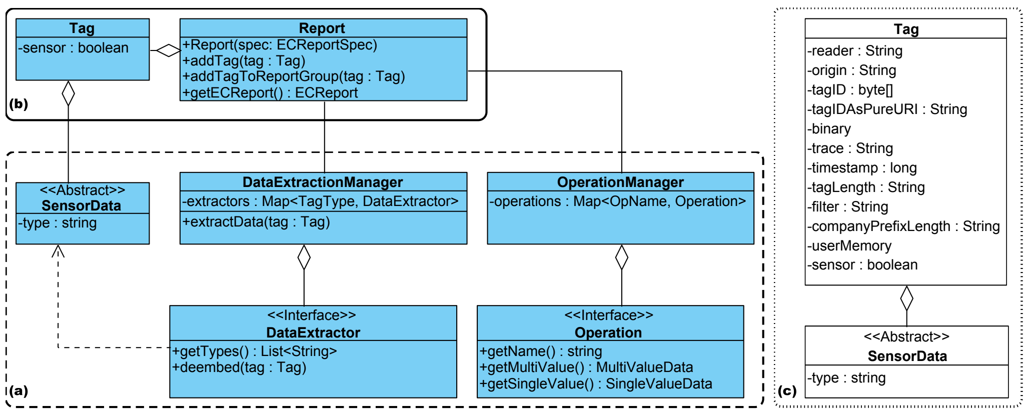


Fig. 3.3 Extensions: (a) new classes; (b) modified classes; (c) tag data model

Sensor data filtering and aggregation Commonly, applications are interested in subsets of collected data (filtered data) [115, 116] such as temperature values above a threshold, or the temperature of a certain product. Sensor data can also be aggregated (e.g. in the time or space domains) based on application requirements. For instance, to provide the average temperature or to combine temperature data from different readers observing a physical location.

Event Cycle Report An **EventCycle** is constructed according to a specification called **ECReportSpec** (see the ALE Specification [27]) generated by a client (see report specification application in Figure 3.2). While the **ECReportSpec** specifies several parameters, the two most important parameters are: i) the time interval during which tag reads are collected and processed; and ii) the readers from which tags are collected. Then Event Cycle Report module is responsible for managing the construction and transmission of the user specified report.

3.3 Middleware Implementation

3.3.1 Proposed Tag Data Model

We have implemented our middleware by extending the Fosstrak open-source middleware [109] which only supports ID tag data management. Selection of Fosstrak is based on: i) its conformance to the EPCglobal architecture [109]; and ii) the support for LLRP (Low Level Reader Protocol) [27], a standard **Reader Interface**, to communicate with RFID readers.

The class diagram in Figure 3.3 illustrates our middleware design in Fosstrak. We have only outlined classes added to (Figure 3.3a) or modified (Figure 3.3b) in the existing Fosstrak implementation to reduce the complexity of the diagram. A complete specification of Fosstrak is available from the developer guide.[2] Modifications are made to support the specific functions outlined in Figure 3.2 for sensor tag data: i) tag data extraction; ii) sensor data grouping; iii) sensor data filtering and aggregation; and iv) event cycle report. Since ID management functionality is already a part of Fosstrak, other aspects of our proposed architecture utilised existing Fosstrak capabilities.

Figure 3.3c illustrates the proposed tag data model (as an extension of the Fosstrak tag data model) for representing sensor data. A new boolean attribute *sensor* is introduced to the `Tag` class to indicate the containment of sensor data. All implementations of a sensor tag specific data models are left to the discretion of the user/developer and must extend the abstract class `SensorData`. The `SensorData` class defines the attribute *type* to identify the type of sensor data reported by a sensor tag (Figure 3.3c).

3.3.2 Tag Data Extraction

`DataExtractionManager` and `DataExtractor` classes provide the framework for tag data extraction where sensor data is de-embedded from the tag identifier where the former acts as a façade and delegates the processing of the tag to the appropriate `DataExtractor` which is identified by the tag type. For instance, acceleration data extractor for a sensor tag with an accelerometer. The `DataExtractionManager` can obtain valid tag types (e.g. `0D` and `0B` for acceleration tags) for an implementation of `DataExtractor` by calling the `getTagTypes` method on the corresponding `DataExtractor` implementation. We have not specified data formats for various sensors and therefore it is possible to have multiple `DataExtractors` for the same sensor where the treatment of the sensor data field (Figure 3.1) is left to the discretion of the end-user and/or developer; thus providing for a flexible and extensible design. Finally, `DataExtractionManager` converts a sensor tag EPC to a Universal Resource Indicator (URI) by only considering the tag ID portion of the EPC (see Figure 3.1), to allow Fosstrak to apply existing ID level filtering and aggregation operations to the ID data of sensor tags.

3.3.3 Sensor Data Grouping, Filtering, and Aggregation

Unlike with ID data, a variety of filtering and aggregation operations can be performed on sensor data depending on application domain requirements and sensor types (e.g. acceleration sensors). `Report` class was modified to implement sensor data grouping based on *logical readers* [27](i.e. the behaviour of a reader from the

[2] http://fosstrak.github.io/fc/docs/developer-index.html

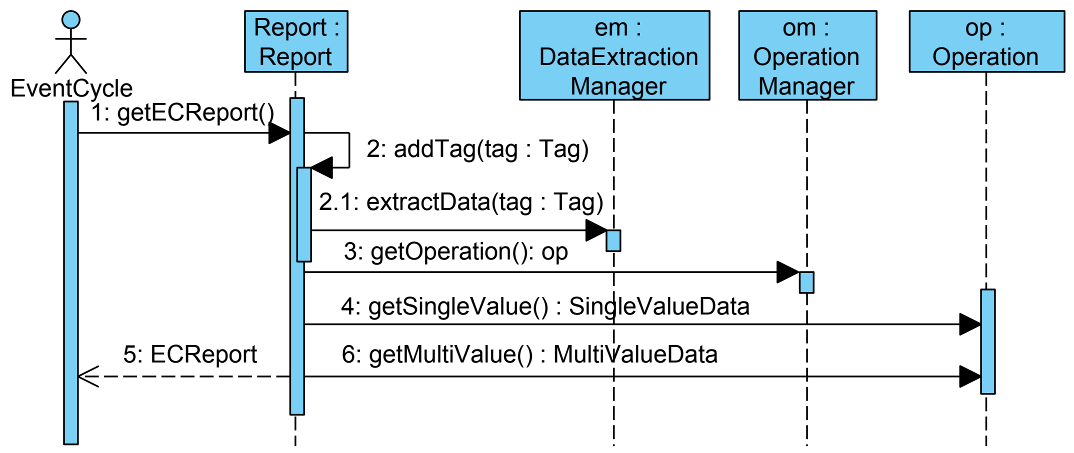


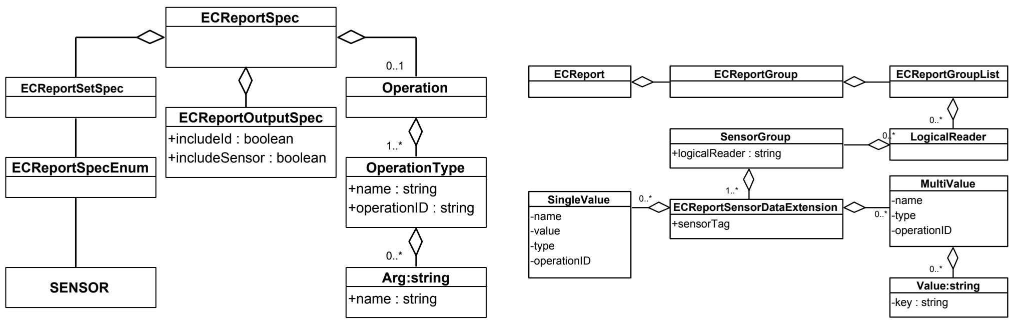
Fig. 3.4 Interaction with Fosstrak

perspective of a report subscriber) and tag IDs. Sensor data filtering is managed by the `OperationManager`, which is responsible for maintaining a list of implementations of the `Operation` interface. Each implementation of the `Operation` interface can encapsulate specific filtering or aggregation operations to be performed on the acquired sensor data. The operations, specified in the **ECReportSpec** by subscribers, are matched against the registered operation implementations using the operation name (see Figure 3.3a `Operation` interface) to support sensor data filtering and aggregation requests.

3.3.4 Event Cycle Report

The sequence diagram in Figure 3.4 visualises the behaviour of an **EventCycle** (see ALE in [27] and Table 3.1). During an **EventCycle** the `getECReports` method in the modified `Report` class that returns an **ECReport** object is invoked to prepare a report according to the **ECReportSpec** specification, which is provided as a constructor argument to the `Report` class upon instantiation when subscribing to the report.

Figure 3.5a shows extensions to the **ECReportSpec** data model to allow clients to subscribe not only to ID tag but also sensor tag data. The corresponding attribute descriptions are provided in Table 3.2. Furthermore, to enable users to specify sensor data filtering and aggregation operations with specific arguments, the new structure, **OperationType**, is introduced to the **ECReportSpec**. The unique *operationID* in **OperationType** allows the users to specify the same operation multiple times with different arguments and isolate each of the returned results without ambiguity. For instance, for an operation that performs Fast Fourier Transformation (FFT), different arguments can be: i) first 10 FFT coefficients and ii) last 10 FFT coefficients



(a) **ECReportSpec** data model extensions to specify sensor data operations

(b) **ECReport** data model extensions to report sensor data

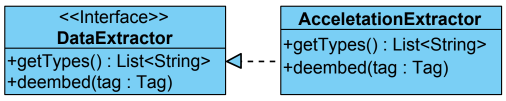
Fig. 3.5 **ECReportSpec** and **ECReport** data model extensions (see Table 3.2 for attribute descriptions)

Table 3.2 Descriptions of the attributes in extended **ECReportSpec** and **ECReport** data models

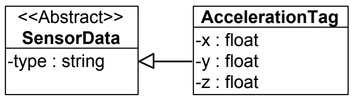
Element	**Attribute**	**Description**
ECReportOutputSpec	*includeId*	If *true* ID data of the ID tags are included
ECReportOutputSpec	*includeSensor*	If *true* ID data of the sensor tags are included
OperationType	*name*	Name of the operation which should correspond to the implemented operation name
OperationType	*operationID*	Uniquely identifies the operation. Operation implementations return results with this ID
Arg	*name*	Name of the argument and is dependent on the enclosing operation
SingleValue/MultiValue	*name*	The name distinguishes the multiple outputs from a single operation
SingleValue/MultiValue	*type*	Data type of the output (e.g. float, double, int)
SingleValue/MultiValue	*operationID*	The *operationId* of the operation which produce the output
SingleValue	*value*	Value returned by the operation
Value	*key*	The key provides unique identification or meta-data for a specific value within a MultiValue element.

The ALE standard [27] defines three kinds of report sets that clients can subscribe to: i) CURRENT (tags in the current **EventCycle**); ii) ADDITIONS (new tags in the current **EventCycle**); and iii) DELETIONS (tags in the previous **EventCycle** that are not in the current **EventCycle**) for ID tag filtering. In the case of sensor data, ADDITIONS and DELETIONS are meaningless, and CURRENT is inadequate. Therefore a new report set, SENSOR, is defined to specify filtering and aggregation operations on sensor data within the current **EventCycle** and these operations are only evaluated if they are specified along with the SENSOR report set.

A new structure for reporting sensor data to subscribers (clients) is introduced to the **ECReport** while maintaining flexibility as well as extendability to support future changes and multiple sensor types. Figure 3.5b shows the design of **ECReport** extensions where the attribute descriptions are provided in Table 3.2. Two data structures were defined to represent single-valued (**SingleValue**) and multi-valued (**MultiValue**) outputs from filtering and aggregation operations. Furthermore, the optional attribute *key* specified in **Value** provides unique identification or meta-data related to the represented value. The value of the *key* is determined by the *operation* implementation (e.g. for operations *FFT*, which is the Fast Fourier Transform, and raw sensor data reporting, the *key* can be the FFT component and timestamp of the reading respectively). The ability to represent the output of any operation is a significant advantage of this framework.



(a) `AccelerationExtractor`



(b) `AccelerationTag`

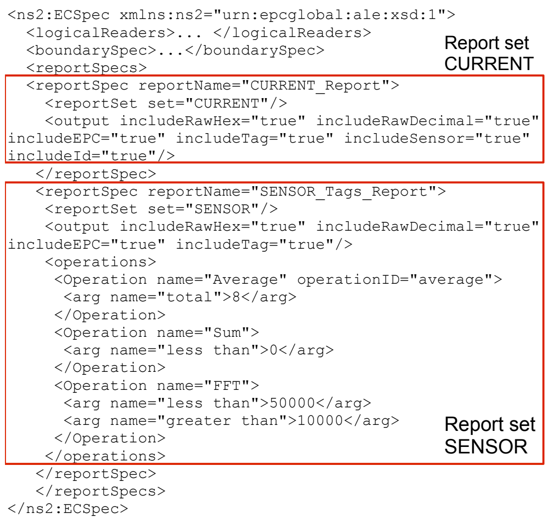
Fig. 3.6 Acceleration tag implementation

3.4 Experiments and Results

Initially, the proposed design for WINDWare was implemented as specified in Section 3.3 using the Fosstrak-1.2.0 release. We implemented the `AccelerationExtractor` class which implements the `DataExtractor` interface and a tag data model for WISP tags with accelerometer sensors (Figure 3.6a). The sensor data, embedded in an EPC from a WISP, are stored in `AccelerationTag` which is assigned as sensor data to the corresponding `Tag` (see Figure 3.3c). Figure 3.5 shows the implementation.

We have implemented five concrete implementations of `Operation` interface to generate sum, average, raw sensor data representation, resultant acceleration and FFT for a sensor tag with an acceleration sensor that reports x, y and z acceleration components. We used **SingleValue** to report results from operations, which return one value (such as sum and average operations). For operations which return multiple values, such as the FFT, **MultiValue** structure was used (Section 3.3.4).

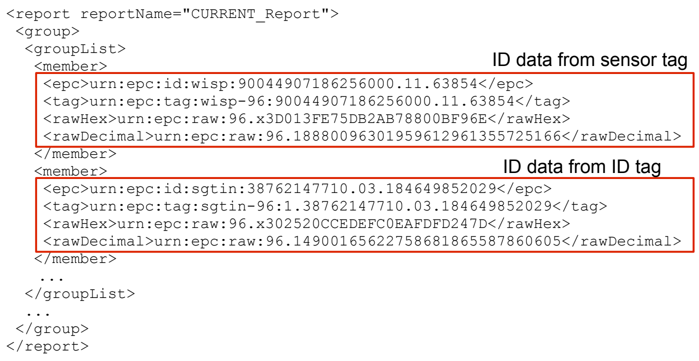
We conducted two sets of experiments: i) with the real hardware; and ii) with an RFID emulator. Our experiments were designed to: i) evaluate ID tag filtering functionality to confirm that existing Fosstrak functionality is not affected, and the sensor tags are correctly processed by existing Fosstrak operations for ID tags; and ii) evaluate sensor data filtering and aggregation operations.



```
<ns2:ECSpec xmlns:ns2="urn:epcglobal:ale:xsd:1">
  <logicalReaders>... </logicalReaders>
  <boundarySpec>...</boundarySpec>
  <reportSpecs>
  <reportSpec reportName="CURRENT_Report">
    <reportSet set="CURRENT"/>
    <output includeRawHex="true" includeRawDecimal="true"
includeEPC="true" includeTag="true" includeSensor="true"
includeId="true"/>
   </reportSpec>
   <reportSpec reportName="SENSOR_Tags_Report">
    <reportSet set="SENSOR"/>
    <output includeRawHex="true" includeRawDecimal="true"
includeEPC="true" includeTag="true"/>
    <operations>
     <Operation name="Average" operationID="average">
      <arg name="total">8</arg>
     </Operation>
     <Operation name="Sum">
      <arg name="less than">0</arg>
     </Operation>
     <Operation name="FFT">
      <arg name="less than">50000</arg>
      <arg name="greater than">10000</arg>
     </Operation>
    </operations>
   </reportSpec>
   </reportSpecs>
</ns2:ECSpec>
```

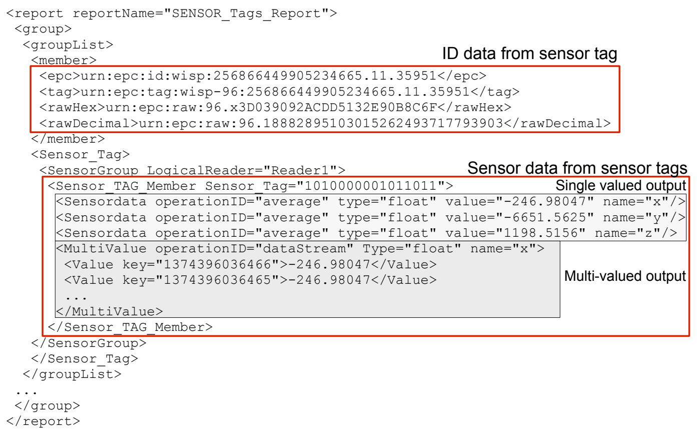
Report set
CURRENT
Report set
SENSOR

(a) Report specification (**ECReportSpec**)



```
<report reportName="CURRENT_Report">
 <group>
  <groupList>
   <member>
    <epc>urn:epc:id:wisp:900449071862560000.11.63854</epc>
    <tag>urn:epc:tag:wisp-96:900449071862560000.11.63854</tag>
    <rawHex>urn:epc:raw:96.x3D013FE75DB2AB78800BF96E</rawHex>
    <rawDecimal>urn:epc:raw:96.18880096301959612961355725166</rawDecimal>
   </member>
   <member>
    <epc>urn:epc:id:sgtin:38762147710.03.184649852029</epc>
    <tag>urn:epc:tag:sgtin-96:1.38762147710.03.184649852029</tag>
    <rawHex>urn:epc:raw:96.x302520CCEDEFC0EAFDFD247D</rawHex>
    <rawDecimal>urn:epc:raw:96.14900165622758681865587860605</rawDecimal>
   </member>
    ...
  </groupList>
  ...
 </group>
</report>
```

(b) **ECReport** for report set CURRENT in (a) with ID data from both ID tags and sensor tags



```
<report reportName="SENSOR_Tags_Report">
 <group>
  <groupList>
   <member>
    <epc>urn:epc:id:wisp:256866449905234665.11.35951</epc>
    <tag>urn:epc:tag:wisp-96:256866449905234665.11.35951</tag>
    <rawHex>urn:epc:raw:96.x3D039092ACDD5132E90B8C6F</rawHex>
    <rawDecimal>urn:epc:raw:96.18882895103015262493717793903</rawDecimal>
   </member>
   <Sensor_Tag>
    <SensorGroup LogicalReader="Reader1">
     <Sensor_TAG_Member Sensor_Tag="1010000001011011">
      <Sensordata operationID="average" type="float" value="-246.98047" name="x"/>
      <Sensordata operationID="average" type="float" value="-6651.5625" name="y"/>
      <Sensordata operationID="average" type="float" value="1198.5156" name="z"/>
      <MultiValue operationID="dataStream" Type="float" name="x">
       <Value key="1374396036466">-246.98047</Value>
       <Value key="1374396036465">-246.98047</Value>
       ...
      </MultiValue>
     </Sensor_TAG_Member>
   </SensorGroup>
   </Sensor_Tag>
  </groupList>
 ...
 </group>
</report>
```

ID data from sensor tag
Sensor data from sensor tags
Single valued output
Multi-valued output

(c) **ECReport** for report set SENSOR in (a) with ID and sensor data from sensor tags

Fig. 3.7 Hardware experiment subscription and reporting

3.4.1 Experiments With a Hardware Setting

Our aim is to test the overall functionality of WINDWare. We used an Impinj Speedway Revolution (R420-GX11M) reader with a circularly polarised antenna, two WISPs with acceleration sensors, and regular RFID tags. We specified two report sets (Figure 3.7a): i) CURRENT; and ii) SENSOR. In the CURRENT report set the WINDWare reported the ID data as expected in the original Fosstrak implementation and provided positive evidence that existing Fosstrak functionality is not affected by our extensions and additions (Figure 3.7b). Furthermore, with the CURRENT report set, it was able to report the ID information of the sensor tags and confirm that the extensions to Fosstrak were successfully able to process the sensor tag data (see Figure 3.7b). When the report set SENSOR is used, only the ID and sensor data related to sensor tags were reported, as shown in Figure 3.7c. Furthermore, the results of the sensor data filtering and aggregation operations were available, which is the expected behaviour. Results from the specification of both CURRENT and SENSOR reports demonstrate the sensor tag data management capability of WINDWare.

3.4.2 Experiments With a Reader Emulator

Rifidi Emulator[3] can emulate: i) the LLRP protocol supported RFID readers; and ii) multiple EPC tags and tag types. Therefore, the Rifidi Emulator is employed to emulate the functionality of one LLRP supported reader and multiple ID and sensor tags required for the experiment. Three report sets were specified: i) CURRENT and SENSOR; ii) CURRENT; and iii) SENSOR. In all report specifications, the report event cycle is specified as two seconds based on a near real-time monitoring application context (see section 3.4.3). All implemented sensor filtering operations were specified in subscription arguments for the SENSOR report set. Tests were carried out having: i) only ID tags; ii) only sensor tags; and iii) a mixture of ID and sensor tags (1:1 ratio for an unbiased assessment) to evaluate the middleware against each tag type. Having only the ID tags provides a baseline for comparing our middleware implementation, as ID tags are processed by the existing ALE implementation in Fosstrak. We recorded the time taken to generate the reports (duration of `getECReportMethod` in **EventCycle**) specified in the subscribed **ECReportSpec**. The emulated reader was configured with the following LLRP settings: i) for **ROBoundarySpec**, **ROSpecStartTriggerType** (which defines the trigger event for the reader to initiate the generation of reports according to the defined **ROSpec**) was specified as *periodic* with the period of *5000* ms;

[3] http://www.rifidi.org/

and ii) for reporting tags **ROReportSpec** (see Table 3.1) was configured to trigger upon each tag detection or end of **ROSpec**.

Figure 3.8a shows the behaviour of the middleware for report set CURRENT specified in **ECReprtSpec** which reported only the ID data for both ID tags and sensor tags. No difference in the mean time between sensor and ID tags is observed.

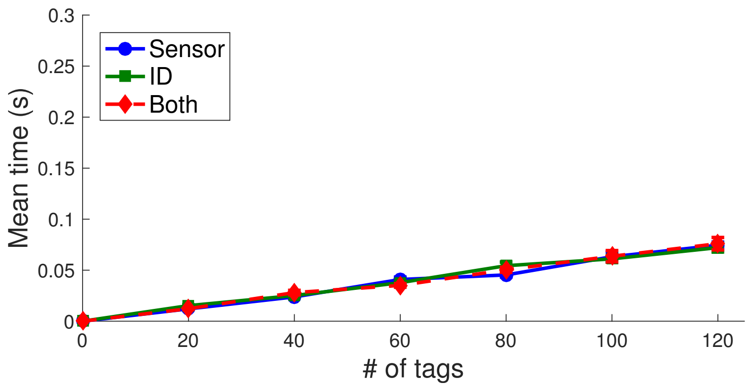
The graph in Figure 3.8b depicts the report generation time when the report set SENSOR (only generate reports for sensor tags) is specified. It is observed that the sensor operations appear to consume linearly increasing time with respect to the sensor tags in the field of view. Where both ID and sensor tags are in the field, a report is only generated for 50% of the tags seen by the reader, as ID tags are filtered out from the SENSOR report set. Hence time measured reflects the time taken to process the sensor tags and filter ID tags.

Figure 3.8c illustrates the mean report generation time when the report sets CURRENT and SENSOR are requested. The SENSOR report set is requested with all the implemented filtering operations. The resulting report contained two separate report sections, i.e. the CURRENT report set and the SENSOR report set. The results are similar to 3.8b but now time taken to process ID tags is also included.

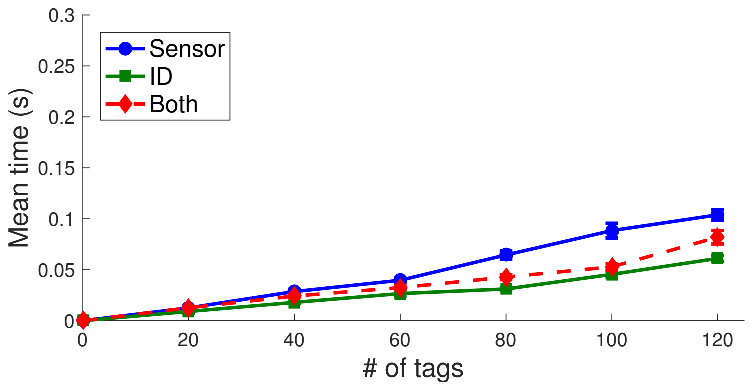
3.4.3 Example Application

To demonstrate our middleware, we developed a prototype monitoring application to identify potential damage to goods (e.g. miss-handling of fragile items such as LCD screens) in a supply chain. The damage monitoring scenario is performed in a laboratory environment using Impinj Speedway Revolution (R420-GX11M) reader with two circularly polarised antennas, as shown in Figure 3.9a. WISP tags having a 3D acceleration sensor were attached to items being monitored.

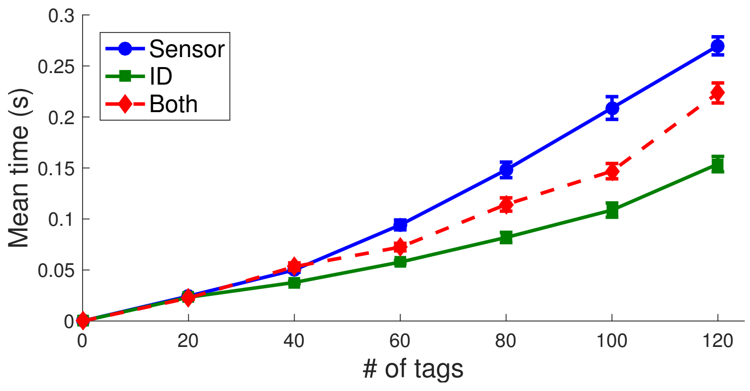
Identification of potential damage is twofold: i) an item dropping onto the ground is identified by detecting the free fall where the resultant acceleration approaches zero; and ii) a damage due to a high impact or a shock is identified by high resultant acceleration. In order to receive notifications on potentially damaged items, **ECReportSpec** depicted in Figure 3.9b is used, where the *Resultant Acceleration* operation with `operationID` *threshold* is specified by two arguments. These arguments takes acceleration values in terms of gravity and a free fall is detected by `less-than` 0.5g while `greater-than` 1.5g detecting high impacts. We have also subscribed to the raw sensor data streams (`operationID`: *dataStream*) and the resultant acceleration (`operationID`: *ra*) to validate the potential damage events reported by the *threshold* operation.



(a) Report set CURRENT

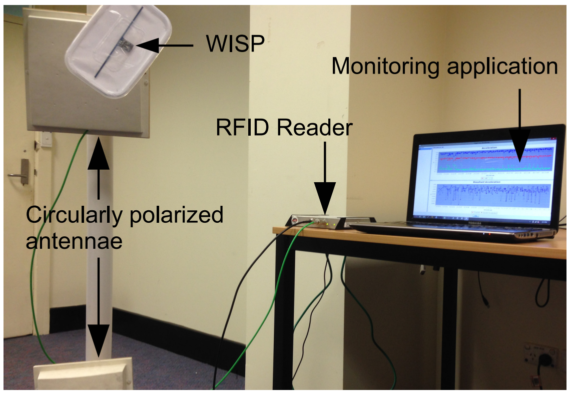


(b) Report set SENSOR



(c) Report sets CURRENT and SENSOR

Fig. 3.8 Report sets from the experimental results with the Rifidi Emulator

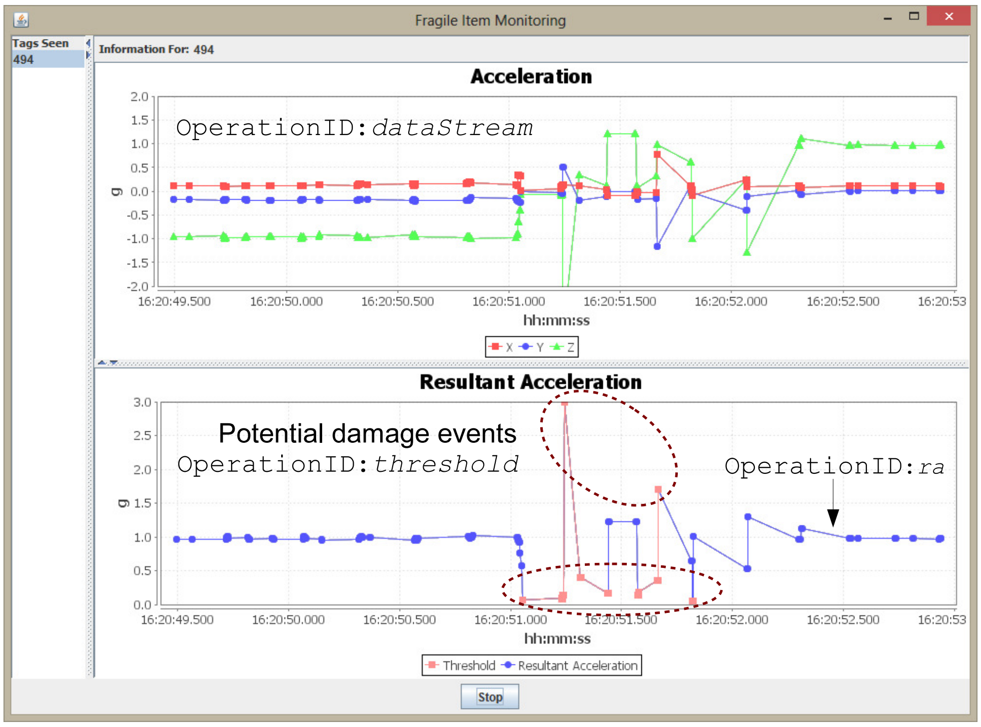


(a) Damage monitoring scenario equipment set-up

```
<operations>
  <Operation name="Data" operationID="dataStream"/>
  <Operation name="ResultantAcceleration" operationID="ra"/>
  <Operation name="ResultantAcceleration"
      operationID="threshold">
    <arg name="greater_than">1.5</arg>
    <arg name="less_than">0.5</arg>
  </Operation>
</operations>
```

Specification to evaluate resultant acceleration and filter to select potential damage events

(b) **ECReportSpec** for damage monitoring application



(c) Screenshot of the prototyped damage monitoring application

Fig. 3.9 An application of WINDWare for real-time monitoring

Figure 3.9c shows a screenshot of the developed monitoring application. This application subscribes to WINDWare using the defined **ECReportSpec** (Figure 3.9b). Items are dropped onto the ground to simulate free fall and high impact (upon collision with the ground). In Figure 3.9c, we have highlighted (plotted in red) the results reported by the filtering operation to select potential damage events based on evaluating the resultant acceleration from the WISP attached to the item. By only subscribing to the operation with `operationID` *threshold*, events of interest (free fall and shock) are received. Therefore, use of sensor data management middleware such as WINDWare: i) reduces the complexity of client applications by allowing application developers to focus on business logic; ii) permits development of applications that are agnostic to underlying sensing infrastructure; and iii) reduces network traffic from large volume sensor data streams.

3.5 Discussion

In this chapter, we presented the design, implementation and evaluation of WINDWare (**W**ireless **I**dentification a**n**d Sensor **D**ata Management Middle**ware**), a middleware for passive sensor enabled RFID tags. The proposed middleware (WINDWare) provides: i) extracting of sensor and ID data; ii) operations on sensor data (filtering and aggregation); and iii) sensor data subscriptions and reporting. The generic middleware architecture is implemented by extending Fosstrak. Furthermore, WINDWare adheres to the standardised EPCglobal architecture through implementing the ALE specification and using the extensions provisioned by the standards. The real-world applicability of WINDWare is demonstrated through a prototype application. The proposed middleware processes the ID data and sensor data in unison, maintaining the important relationship between sensor data and its source. This association is paramount, particularly for real-time applications such as human activity recognition [61].

The proposed tag data format makes extending the WINDWare trivial as there are no modifications required to existing implementations. Similarly, as discussed in Section 3.3, sensor data operations can be integrated seamlessly into the implementation, without modifications to the existing sensor data subscription or reporting implementations. Thus, WINDWare facilitates the acquisition and manipulation of sensor data from passive sensor enabled RFID tags. The WINDWare source code is available at WINDWare GitHub page[4] and was partly developed by Mr Yu Yan whilst an undergraduate at the University of Adelaide.

[4]https://github.com/AdelaideAuto-IDLab/windware

Chapter 4

Features for Activity Recognition

4.1 Introduction

In Chapter 2, we observed that most of the studies on Human Activity Recognition (HAR) using wearable sensors relied on battery powered sensors and these studies had used a wide array of features. Broadly, these features are calculated based on the statistical measures and frequency domain properties of sensor data streams, as well as considering the biomechanical motion of sensor wearers [44, 51, 52].

In this thesis, we utilise a sensor tag, which is a passive acceleration sensor embedded in an RFID platform (see Section 2.3.2) for HAR instead of a battery powered sensor. Unlike battery powered sensors, data streams from sensor tags are sparse, with low and variable sampling rates because passive sensors need to harvest adequate energy prior to powering and sampling a sensor [23]. Furthermore, due to insufficient power to the embedded accelerometer may increase the measurement noise. As a result of this sparsity, features that require a data stream with a regular sampling rate cannot readily be extracted. For example, frequency-domain features [17, 44, 117] require a data stream with a regular sampling rate. More recently, autoregressive coefficients have successfully been used as features in HAR [35]; these features also require a data stream with a regular sampling rate to be meaningful.

In this chapter, we initially provide a brief overview of the motion patterns that are observed in ambulatory monitoring in an older care setting (see Section 4.2). The sensor tag used in this thesis has two data sources (see Section 2.3.2): i) frontal (a_f), lateral (a_l) and vertical (a_v) acceleration values from the embedded accelerometer; and ii) the information from the RFID platform. In particular, we describe the features extracted from the acceleration signal (see Section 4.3.1) and features extracted using the information from the RFID platform (see Section 4.3.2). Since the data streams from sensor tags are sparse , we propose a dynamic sensor data augmentation algorithm,

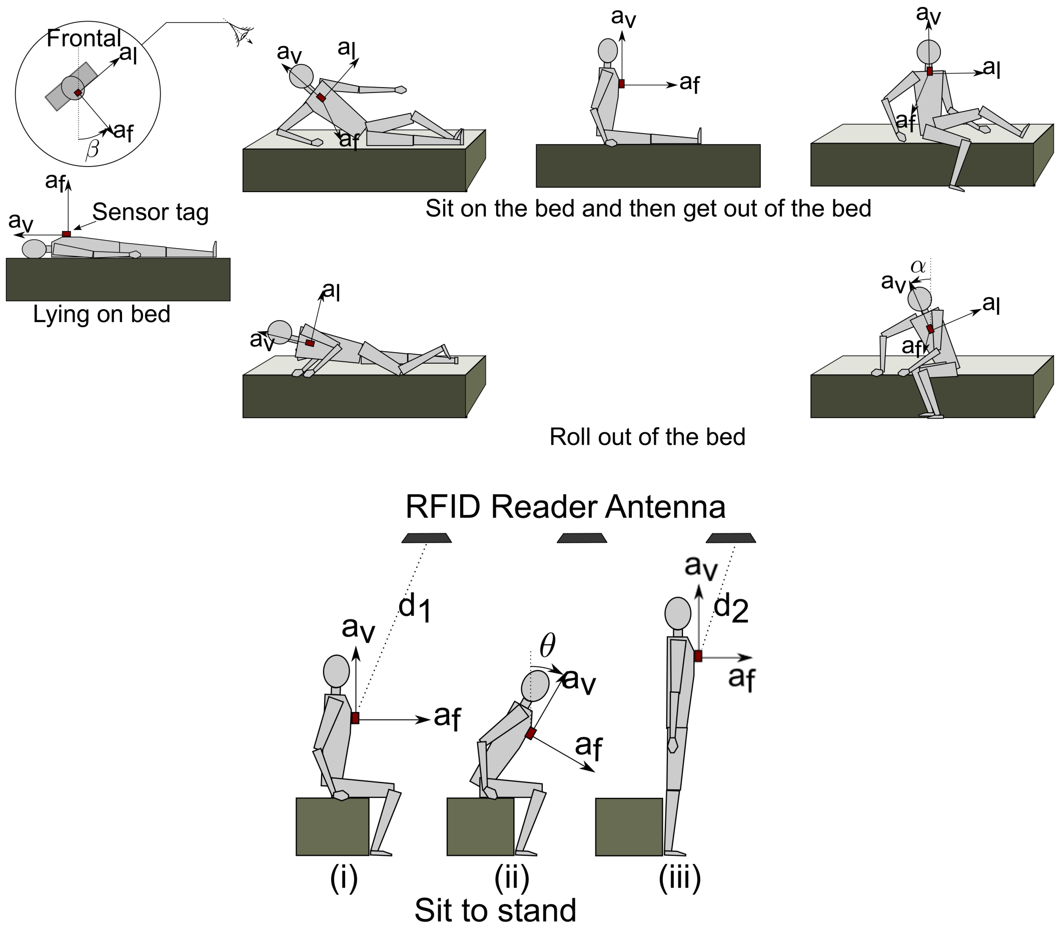


Fig. 4.1 Analysis of ambulating out of the lying posture

an approach to facilitate online interpolation of sparse acceleration data streams from sensor tags to allow extraction of features that require a data stream with a regular sampling rate (see Section 4.4). Our approach reduces interpolation errors when interpolating sparse sensor observations that are temporally distant by augmenting the sensor data stream using the most recent sensor observations. We implement five real-time sensor data stream interpolation methods with an increasing order of complexity. Finally, we provide evaluations, using the four datasets described in Section 2.4, on the interpolation methods and different feature sets that are possible from unprocessed and interpolated sensor tag data streams. In this chapter, we extend the work which was published in the International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services proceedings [29].

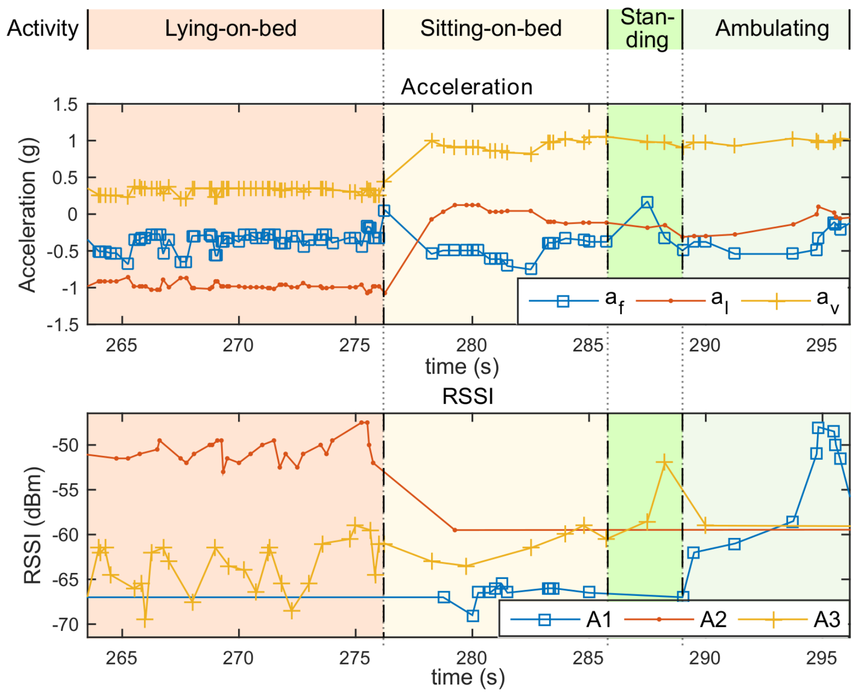


Fig. 4.2 Acceleration and RSSI signal patterns collected from W^{2}ISP in *HOA-Room2* for the motion sequence shown in 4.1, where a participant got out of the bed from lying-on-bed (lying-on-side).

4.2 Motion Analysis

Human motion analysis provides the basis for formulating features used in this thesis; in particular, we analysed the movements involved in getting out of the bed, as shown in Figure 4.1. Participants may lift their trunk to a vertical position using their dominant arm when they are lying in a supine posture and then turn their body, facing the edge of the bed to place their legs on the ground. On the other hand, they can also roll over to the edge of the bed and then use the weight of the legs and arm to raise their trunk to a vertical position. When the sit-to-stand posture transition is considered, as also first shown by Najafi et al. [62], participants first bent forward (stage (ii) in sit-to-stand transition in Figure 4.1) and then stand up.

Figure 4.2 shows the RSSI signal in *HOA-Room2* for an activity sequence roll-out-of-the-bed illustrated in Figure 4.1. These motion patterns are captured by the accelerometer (see Figure 4.2). The measured acceleration signals (a_f, a_l and a_v) and rotational motion of the trunk on sagittal (θ), coronal (α) and transverse (β) planes, approximated using the acceleration signal, can be used to describe these movement patterns. For instance, during the sit-to-stand transition, the acceleration in the vertical direction a_v and the trunk rotational motion on the sagittal plane (θ) provides a good description of the motion [62]. Furthermore, as shown in Figure 4.1,

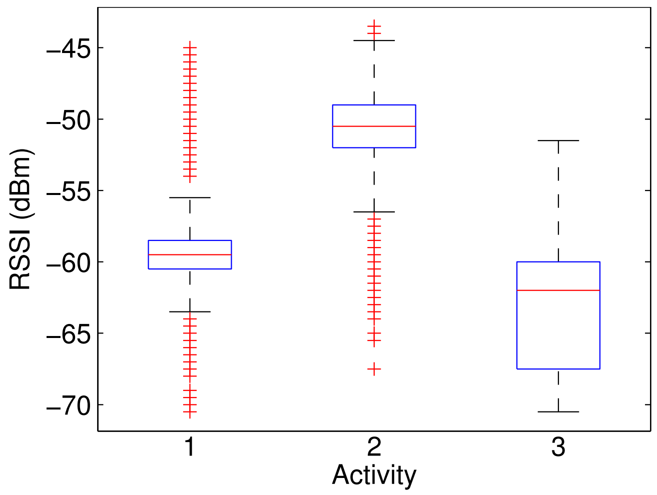


Fig. 4.3 Distribution of the RSSI values received at A3 in *HOA-Room2* (1: sitting-on-bed; 2: lying-on-bed; 3: out-of-bed). Boxes are drawn at the 1st and 3rd quartile, the central mark on each box is the median and whiskers represent $\approx \pm 2.7\sigma$

the rotational motion of the trunk on coronal (α) and transverse (β) planes also provide information on the getting-out-of-the-bed of a participant, apart from the directly measured acceleration signal.

It is noteworthy that the distance between the sensor and a fixed RFID antenna also varies as a result of human motion. For instance, during the sit-to-stand transition illustrated in Figure 4.1, the distance from the sensor tag to the RFID antenna while the participant is sitting on the bed (d_1) is greater than the distance from the sensor tag to the RFID antenna while the participant is standing (d_2); i.e. $d_1 > d_2$. This change in distance is reflected in the RSSI measured by the RFID platform.

The RSSI indicates the power level of the backscattered tag response detected by the reader and this power is given by:

$$RSSI = P_t G_t^2 G_{path}^2 K \tag{4.1}$$

where P_t is the output power of the reader, G_t is the gain of the reader antenna, K is the backscatter gain of the sensor tag and G_{path} is the one-way path gain of the deterministic multipath channel, determined according to Eq.(4.2) [99].

$$G_{path} = \left(\frac{\lambda}{4\pi d_0}\right)^2 |H| \tag{4.2}$$

In Equation Eq.(4.2), d_0 is the direct path length, λ is the wavelength of the RF signal and H is the channel response due to the multipath. From Eq.(4.1) and Eq.(4.2), we can see that the RSSI is hypersensitive to distance d_0 as $RSSI \propto 1/d_0^4$. This indicates that the RSSI can be used to approximate that distance between the sensor and a given antenna.

From Figure 4.2, we can see that when a participant is lying on the bed, there is a lower RSSI value compared with when the participant is in a standing or seated position because of the larger distance between the sensor tag and A3. Figure 4.3 shows the distribution of absolute RSSI values for sensor observations collected from A3 in *HOA-Room2*. From this Figure, we can see that although there are overlaps between the distributions, each activity has a different RSSI distribution. This difference with respect to each activity arises due to the aforementioned variation in distance between the sensor tag and RFID antennas while performing activities. Considering this behaviour of the RSSI, an approximation of the participant's location within the room relative to a fixed antenna can be made.

The following sections describe the calculation of features based on the above observations, considering the two information sources; acceleration and the information from RFID platform.

4.3 Features from Sensor Tag Data Streams

We denote the i^{th} sensor datum at time t_i, $t_i > t_{i-1}$, $i \in \mathbb{N}$, using the pair (t_i, s_i), where $s_i = [a_f, a_v, a_l, RSSI, aID]$. Because of the irregular nature of the data collection, the sequence of collected data $X = \{(t_i, s_i)\}_{i \geq 1}$, $i \in \mathbb{N}$ is a non-uniformly sampled time series. We use subscripts to denote subsequences; for instance $X_{[a,b]}$ denotes the sensor data stream segment for time interval $[a,b]$, where $b > a$. For this feature calculation, we consider a fixed time sensor data stream segment of length δt for each sensor observation. Therefore, for the i^{th} sensor observation, features are extracted using the sensor data subsequence $X_{[t_i - \delta t, t_i]}$.

4.3.1 Features from Acceleration Signals

As can be seen from Section 4.2, acceleration signals capture both translational motion and rotational motion. In this section, we describe the acceleration based features used in this thesis under three categories: i) instantaneous features; ii) time-domain features; and iii) frequency domain features.

Instantaneous Features

Nine instantaneous features are extracted considering the most recent sensor observation. We have utilised raw acceleration values, frontal (a_f), lateral (a_l) and vertical

Table 4.1 Features from statistical measures

#	Feature Name	#	Feature Name	#	Feature Name
1	mean	5	standard deviation	9	maximum value
2	mode	6	mean absolute deviation	10	range
3	median	7	median absolute deviation	11	skewness
4	variance	8	minimum value	12	kurtosis

(a_v), from the sensor tag as features. We also approximate the angles illustrated in Figure 4.1: i) θ—angle between the gravity vector and the coronal plane of the body (Eq.(4.3)); ii) α—angle between the gravity vector and the sagittal plane of the body (Eq.(4.4)); and iii) β—angle between the coronal plane and the world horizontal axis (which it perpendicular to the gravity), particularly while the patient is lying on bed.

$$\theta \approx tan^{-1}(a_f/\sqrt{a_v^2 + a_l^2}) \tag{4.3}$$

$$\alpha \approx tan^{-1}(a_l/a_v) \tag{4.4}$$

$$\beta \approx tan^{-1}(a_l/a_f) \tag{4.5}$$

This approximation relies on the components of gravity on the directions of the accelerometer axes to determine the orientation and acceleration due to human motion is assumed to be negligible compared with the acceleration due to gravity. These angular measurements provide a general description of the sensor wearer's postures and posture transitions, as illustrated in Figure 4.1. Although the angle θ has been used previously in [62, 69, 61], the angles α, β from the acceleration signals have not been considered previously for activity recognition.

Based on these angles (θ, α and β), we can also estimate the acceleration due to human motion relative to a reference frame having z-axis in the vertical in the word and the x-axis on the frontal direction of the body by projecting the acceleration vectors to the axes of the reference frame as follows.

$$a_x \approx a_v sin(\theta) + a_f cos(\theta) \tag{4.6}$$

$$a_y \approx a_v sin(\alpha) + a_l cos(\alpha) \tag{4.7}$$

$$a_z \approx 1 + a_v cos(\theta) cos(\alpha) + a_l sin(\alpha) + a_f sin(\theta) \tag{4.8}$$

Time-Domain Features

Most of the previous research relied on time-domain features for HAR [34, 62, 17, 80]. We consider 12 statistical measures as features listed in Table 4.1. In addition, to the 12 statistical features, we consider whether the maximum value occurs prior to the minimum value in the data stream segment and denote this as *Maximum Prior to Minimum* (*MPM*). This information is represented as follows in Eq.(4.9).

$$MPM(X) = \begin{cases} 1, & \text{if } \arg\max_{x_i \in X} < \arg\min_{x_i \in X} \\ 0, & \text{otherwise} \end{cases} \tag{4.9}$$

.

In Equation Eq.(4.9), X is a one-dimensional vector for the sensor observation segment for a given measure, such as a_f or θ. In particular, this feature is based on previous work by Najafi et al. [62], where they captured posture transitions using a similar methodology.

The histogram based on the signals can be used to estimate the distribution of the signal [117]. Therefore, we created a histogram with ten equal bins and utilised them as features and we denoted this as *hist*. Furthermore, as also highlighted in previous activity recognition research, the correlation between the acceleration axes (*corr*) contains information to distinguish activities. We have also considered this feature for our experiments [17].

The change in vertical velocity and distance using vertical acceleration have been approximated and used as features [62, 69]. Inspired by these features, we approximate the change in velocity (Δv) and displacement (Δd) for the body motion in all three directions relative to the human body (anteroposterior axis-x; mediolateral axis-y; and dorsoventral axis-z) for the motion during a given segment as given in Eq.(4.11).

$$\Delta v \approx \int_0^{\delta t} a\,dt \tag{4.10}$$

$$\Delta d \approx \int_0^{\delta t} v\,dt \tag{4.11}$$

The integration in the above equations were approximated using trapezoidal integration. Furthermore, we have considered the change in the resultant velocity Δv_r calculated based on Eq.(4.11) using resultant acceleration without gravity $a_r = \sqrt{a_x^2 + a_y^2 + a_z^2}$. Previous, this feature has been shown to possess activity information in [69].

Coefficients of an autoregressive model (*ARC*) have also been considered as features for activity recognition [35]. As shown in Section 2.4.4, the data streams from sensor tags have a variable sampling rate. However, the ARC features need to be extracted from a data stream with fixed sampling rate. Therefore, data streams from sensor tags need to be interpolated in order to calculate the ARC features.

Frequency-Domain Features

We consider frequency domain features calculated based on Fast Fourier Transformation (FFT) applied to the information within a segment. Frequency domain features based on Fourier transformation have been used in activity recognition research [17, 44, 117]. When the sensor is adequately powered, a data stream with an upper bound sampling rate of 40 Hz can be obtained (see Section 2.3.2). Therefore, information regarding human activities which constitute of lower frequencies can be captured using frequency-domain features. These features were calculated with respect to acceleration signals, a_f, a_l, a_v, and the resultant acceleration signal, a_r.

We have considered lower order coefficients as features [44]. This feature captures the dominant frequency associated with each activity and we considered first 80 FFT coefficients.

We have also calculated the signal energy (SE) based on Fourier transformation. Given that FFT is applied to obtain n coefficients, the energy is calculated as:

$$SE = \sum_{i=1}^{n/2+1} (2x_i)^2 \tag{4.12}$$

Here x_i is the i^{th} Fourier coefficient. In Equation Eq.(4.12), up to $n/2+1$, elements are considered owing to the Nyquist frequency. In addition, we divided the frequency spectrum into ten equal bands and calculated the energy for each band to be used as features.

We consider the spectral density as a feature [117]. Specifically, as described in [117], we considered the first ten FFT coefficients, excluding the zero-order component (9 features) to calculate the spectral density.

In addition, we also consider spectral entropy (PSE) as a feature. This was calculated using the spectral density of the entire signal, $P(x_i)$ as given in Eq.(4.13).

$$\begin{aligned} P(x_i) &= \frac{1}{N}|x_i^2| \\ p_i &= \frac{P(x_i)}{\sum_{i=1}^{n} P(x_i)} \\ PSE &= -\sum_{i=1}^{n} p_i ln(p_i) \end{aligned} \tag{4.13}$$

4.3.2 Features from the RFID Platform

One of the advantages of using sensor tags is that the availability of information from the RFID platform such as RSSI and the antenna (aID) that captured a given sensor observation; this has been explained in Section 2.3.2. We have mentioned in Section 4.2 that the RSSI and aID depicts variations based on the distance and location of the sensor tag. These variations can be used to recognise activities [87].

Similar to acceleration signals, we utilise the eight statistical features that are $\{1,3,5,8,9,10,11,12\}$ listed in Table 4.1. Furthermore, we also utilise the feature Maximum Prior to Minimum Eq.(4.9). These features are calculated considering each antenna in the RFID deployment.

It is further observed that the RFID reader antennas that collect sensor observations vary with the location of the participant since the RFID antenna facing the sensor is most likely to both power and collect the response from the sensor tag. For instance, from Figure 4.2, we can observe that while the participant is lying-on-the-bed, sensor observations are not captured by the A1 antenna. To represent this information we utilise a binary feature, $ant \in \{0,1\}^{|\mathscr{A}|}$ where $\mathscr{A}$ is the set of antennas in an RFID deployment.

Furthermore, following the work by Krishnan and Cook [49] and Shinmoto Torres et al. [46], we also use a mutual information based measure to quantify this information. We calculate the mutual information by counting the combinations of observations originating from different antennas and dividing the counts by the total number of observations within the sensor observation series to normalise them. The normalisation is important to obtain a consistent representation, as the number of observations for a given sensor observation sequence is variable. Thus a matrix of mutual information mi can be obtained. An element $mi_{a,a'}$ represents the weighted number of co-occurrences of the observations from antenna a and antenna a', and its value is obtained as in Eq.(4.14).

$$mi_{a,a'} = \frac{1}{n}\sum_{i=1}^{n-1} \mathbf{1}_{[\{a,a'\}=\{ant_i,ant_{i+1}\}]} \tag{4.14}$$

In Equation Eq.(4.14), $\mathbf{1}_{[x]}$ assumes 1 if x is true and 0 otherwise, ant_i is the antenna ID for the i^{th} sensor observation and n is the number of observations in the considered subsequence of sensor observations. The considered combination of antennas is presented within $\{\cdot,\cdot\}$.

We utilise two approaches to extract features based on mutual information from a segment. We specifically use the approach $MI2$, described in [46]. We refer to this features as the *mutual information weighted sensor observation count* (miW) since it is obtained as a weighted count of observations where weights are retrieved from a pre-calculated mutual information matrix considering the antenna for the current sensor observation as the reference. Given a pre-calculated mutual information *mi* according to Eq.(4.14) using available data, $miW_i \in \mathbb{R}^{|\mathscr{A}|}$ for the i^{th} sensor observation is obtained as in Eq.(4.15).

$$miW_i(a) = mi_{ant_i,a} \frac{1}{|X_{[t_i-\delta t,t_i]}|} \sum_{j=1}^{|X_{[t_i-\delta t,t_i]}|} \mathbf{1}_{[ant_i=ant_j]},\ a \in \mathscr{A} \tag{4.15}$$

The other approach is to create a mutual information matrix per segment and consider the combinations of different antennas as features. Thus, $\binom{|\mathscr{A}|}{2}$ number of features can be obtained. We refer to this feature as *antenna co-occurrences* (*miC*).

4.4 Interpolating Sparse Acceleration Signals

Although, there are methods to obtain frequency-domain features based on Non Uniform Fast Fourier Transform (NUFFT) from sparse data streams [118, 119], other features such as autoregressive coefficients and discrete cosine transformation coefficients [120] that requires a regular sampling rate cannot be readily extracted using these approaches. Therefore the common approach to achieve a regular sampling rate is to interpolate the signal. To this end, we interpolate the acceleration signals from the sensor tags within a given segment, $X_{[t_i-\delta t,t_i]}$, to obtain a data stream segment with a regular sampling rate.

However, interpolating between sensor observations that are temporally distant would lead to large interpolation errors due to the unavailability of sufficient information to approximate the acceleration signal, as indicated by large temporal gaps in our datasets (see Section 2.4.4). For example, consider a function $f : [a,b] \mapsto \mathbb{R}$, where $[a,b]$ denotes a bounded interval in $\mathbb{R}$, approximated by an n^{th} order polynomial interpolant P using a distinct set of $n+1$ data points $D = \{x_0 \ldots x_n\} \subset [a,b]$. Then the interpolation error $E_n^f(x) = f[x_0 \ldots x_n] \prod_{j=0}^{n}(x-x_j)$, where $f[x_0 \ldots x_n]$ represents the n^{th} order divided difference [121]. $E_n^f(x)$ clearly indicates that the interpolation error increases with $(x_{j+1}-x_j)$. Consequently, interpolating between sensor observations that are temporally distant leads to extracting poor contextual information related to the current activity.

To overcome this issue we use the dynamic sensor data augmentation algorithm in Algorithm 1, which considers a sequence of sensor observations $X_{[t_i-2\delta t,t_i]}$ to interpolate the i^{th} segment. Different interpolants require different numbers of minimum data

Algorithm 1 Dynamic sensor data augmentation

```
Require: $X$ // Received sensor observations, $i$ // Current index, $N$ // Number of sensor observations required by an interpolant, $\delta t$ // Segment size
Ensure: $\hat{X}^{int}$ // Observations for interpolation.
1: $\hat{X}_i = X_{[t_i - \delta t, t_i]}$
2: $\hat{X}^{aug} = \{\emptyset\}$
3: $s = \underset{j \in \{j | t_j \leq t_i, t_j > t_i - 2 \times \delta t\}}{argmax} \quad t_i - t_k$
4: if $|\hat{X}_i| < N$ then
5:     for $j = 1$ to $N - |\hat{X}_i|$ do
6:         $t_{s-j} = t_s - j \times \delta t$
7:         $\hat{X}^{aug}_j = (t_{s-j}, s_s)$
8:     end for
9:     $\hat{X}^{int} = \hat{X}^{aug} \cup \hat{X}_i$
10: end if
```

Table 4.2 Summary of interpolation methods used

Pre-processing	Notation
Raw signal	*Raw*
Linear interpolation	*Lin*
Cosine interpolation	*Cos*
Cubic interpolation	*Cub*1
Cubic convolution interpolation [122]	*Cub*2
Lagrange interpolation	Lag

points, N, for successful interpolation. If there is an inadequate number of data points for interpolation (line 4 in Algorithm 1), then the furthest sensor observation (t_s, s_s) from (t_i, s_i), where $t_s \leq t_i$, is replicated at time steps of δt from t_s until the required number of data points for interpolation is obtained (line 5 in Algorithm 1). For example, if a single sensor observation needs to be augmented, then it is augmented as $(t_{s-1}, s_{s-1}) = (t_s - \delta t, s_s)$.

We consider piecewise interpolation using five interpolation methods summarised in Table 4.2. Figure 4.4 illustrates an example resultant signal when the aforementioned interpolation methods are applied to a 1D sparse acceleration data stream. We select these interpolants, considering the complexity and the number of data points required for the interpolation. A lower number of data points results in a poor approximation but these methods are less affected by larger sampling intervals. In contrast, a higher number of data points, typically, provides a better approximation of a signal but these methods are more influenced by larger sampling intervals as in our case. Both linear and cosine interpolants require two data points. In contrast to the linear interpolation

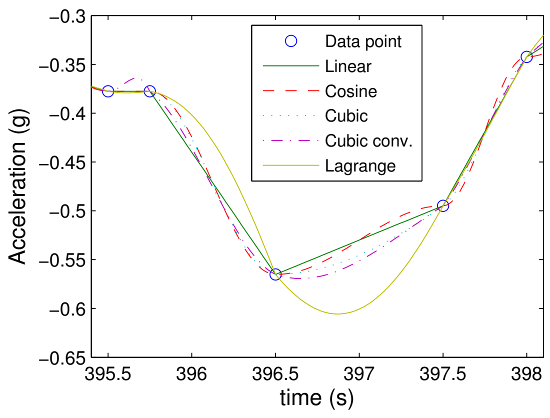


Fig. 4.4 Segment of the a_f acceleration signal from a sensor tag interpolated to achieve 40 Hz sampling rate using different interpolation methods

method, the cosine interpolation method produces a smoother signal, as shown in Figure 4.4. Cubic and fourth-order Lagrange polynomial interpolants require 4 and 5 data points respectively. The cubic convolution [122] interpolation method, which was initially proposed for image data interpolation, only requires 3 data points and produces a signal similar to the cubic interpolation method (see Figure 4.4) but with one less data point.

4.5 Experiments

In this study, we evaluate HAR performance when different interpolation methods, summarised in Table 4.2, are used to condition the sparse data streams from a sensor tag prior to feature extraction. Our main aim is to identify whether the additional features extracted using the interpolated data stream achieve a significant performance improvement over the features possible from a raw sensor data stream in the context of body-worn sensor tags. We also evaluate the mean time taken to generate a feature vector when interpolation is utilised as a measure of the cost involved in the additional processing of sensor tag data stream (see Table 4.2).

We utilise two state-of-the-art discriminative classification algorithms successfully used in previous activity recognition studies: i) a Support Vector Machine (SVM) [63]; and ii) Conditional Random Fields (CRF) [25]. We utilise linear SVM (LSVM) [123], non-linear SVM with Radial Basis Function (RBF) kernel (NSVM) [124] and the linear chain CRF implementation proposed in [46] because we are interested in applications that require real-time HAR. We train activity recognition models based on LSVM, NSVM and CRF to recognise activities: i) sitting-on-bed; ii) lying-on-bed; iii) ambulating; and iv) sitting-on-chair, using the datasets described in Section 2.4.

We have observed in Chapter 2, that the segment size influences classification performance. The segment size also influences the feature extraction performance as it increases the number of data points to be processed. Therefore, we also evaluate different segment sizes for feature extraction and different interpolation methods.

We evaluated HAR the performance based on the mean F-score for multi-class classification. F-score is the harmonic mean of the metrics *precision* (P) and *recall* (R) and is calculated as $F\text{-}score = 2.P.R/(P+R)$ where precision (P) and recall (R) are used as per the standard definitions. In contrast with accuracy, the F-score provides a better view of the classifier performance, especially for datasets with imbalanced class distributions as in our case, because the F-score is not biased towards the majority class [125]. We also present our final results using accuracy, precision, recall and specificity for completeness.

We obtained activity recognition performance using the 10-fold cross-validation strategy. We placed each trial (a continuous recording of a broadly scripted activity sequence from a single participant) randomly in tandem. Then we subdivided the dataset into 10 portions (folds) where each fold constitutes complete activity sequences, as required by the CRF classifier. The first eight folds are used as the training data to obtain the validation performance for parameter selection. Then testing performance is obtained by training the classifier using the first 9 folds and testing on the 10^{th} fold. This process is repeated ten times by rearranging the folds by shifting circularly by one. The set of hyper-parameters resulted in the highest mean F-score for validation is selected for the final model.

Using the sensor tag data stream, three sets of features can be obtained: i) features from raw acceleration signals (ACC); ii) features from the information the RFID platform ($RFID$); and iii) features from interpolated acceleration signals (INT). In this experiment, we present HAR performance with respect to combinations of these feature sets.

4.5.1 Evaluation of Interpolation Methods

We initially conduct an experiment to evaluate the HAR performance when using features available from unprocessed data streams and features that can be calculated with interpolated data streams. The main goal of this experiment is to identify a suitable interpolation method as well as evaluate the cost of interpolation.

For this experiment, we only consider *HOA-Room1* and *HOA-Room2* datasets (see Section 2.4.2). As explained in Section 2.4.2, these datasets were collected from 14 older participants. The dataset *HOA-Room1* consists of four RFID antennas and *HOA-Room2* contains three RFID antennas.

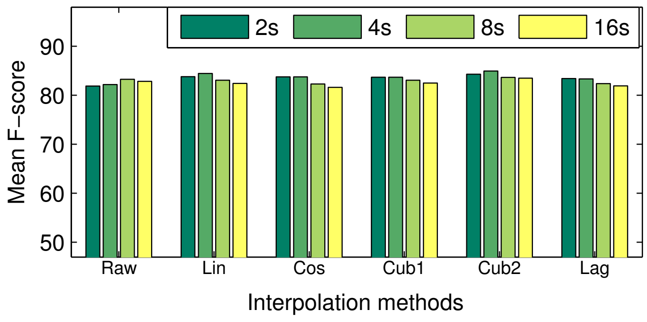
Table 4.3 Features for the evaluation of interpolants

Raw Acceleration (*ACC*)	**RFID Platform** (*RFID*)	**Interpolated Acceleration** (*INT*)
a_f	RSSI	80 FFT coefficients
a_l	aID	Signal Energy (SE)
a_v	miW	Spectral Density
θ		Spectral Entropy (PSE)
α		AR Coefficients
Δv_z		
Δd_z		

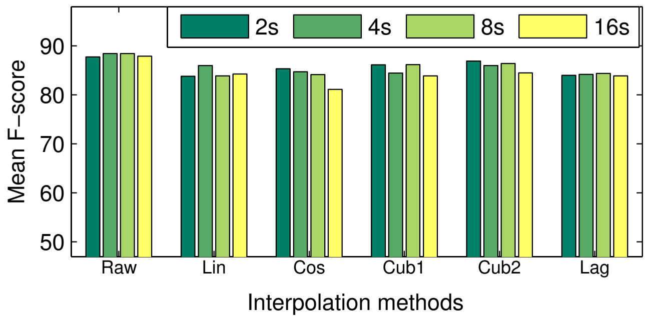
Table 4.3 lists the selected features used for this evaluation and these features are described in detail in Section 4.3.1 and Section 4.3.2. During the experiments, we have utilised all the features listed in Table 4.3 for each interpolant and we utilise features possible from raw acceleration and features from RFID platform when the unprocessed (Raw) data stream is considered.

Features from Raw and Interpolated Acceleration Signals

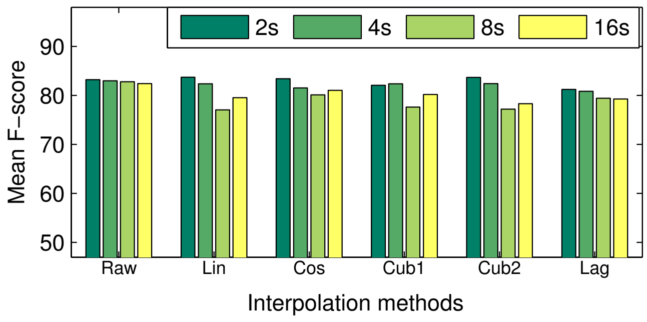
We obtained activity recognition performances for each interpolant (see Table 4.2), based on segment sizes $\delta t \in \{2,4,8,16\}$ for each classifier. Figure 4.5 and Figure 4.6 present activity recognition performances for the two datasets with the three classification models. Table 4.4 presents the highest performing classifiers for each interpolation method for both datasets, i.e. *HOA-Room1* and *HOA-Room2*. From these results, we can observe that, in 4 out of 6 instances, performance (i.e. mean F-score) of *Cub*2 is higher but statistically insignificant compared with other interpolation methods (excluding *Raw*). This general consistency is because the *Cub*2 interpolant can provide a better approximation for acceleration signals when compared with *Lin* and *Cos* interpolants. Furthermore, noisy acceleration data have a lower influence on *Cub*2 compared with *Cub*1 ($N = 4$) and *Lag* ($N = 5$) because *Cub*2 ($N = 3$) require a lower number of data points. Given the sparse nature of the data stream, interpolants that require higher numbers of data points, such as *Cub*1 and *Lag*, result in the inclusion of temporally distant sensor observations for successful interpolation (see Section 4.4). This also results in poor approximations of missing acceleration data as the temporally distant observations are most likely to belong to different activities. Therefore, *Cub*2 is able to produce more informative features for HAR. On the other hand, *Lag* depicted the lowest performance among all the interpolants. The main reason is that *Lag* is more influenced, compared with others, by data stream sparsity as it requires five data points. Additionally, if the number of data points, N, required for interpolation is $N < 5$, then sensor observations are augmented as stated in the algorithm in Algorithm 1.



(a) LSVM

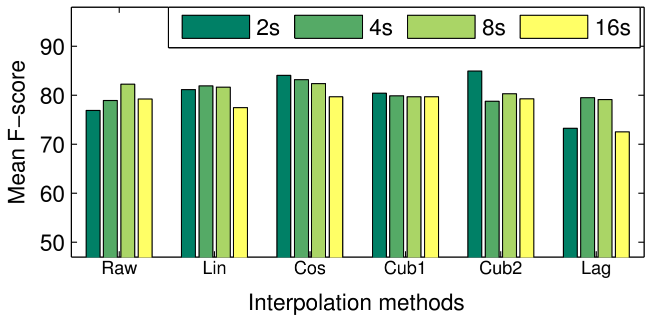


(b) NSVM

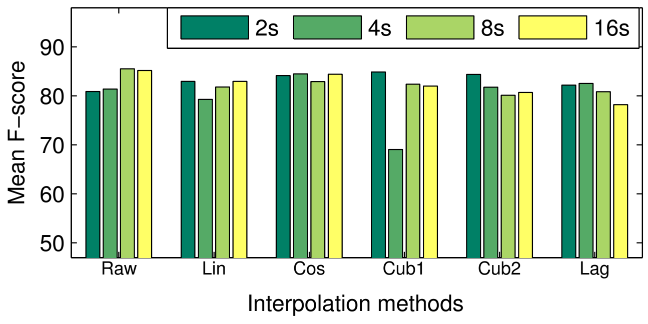


(c) CRF

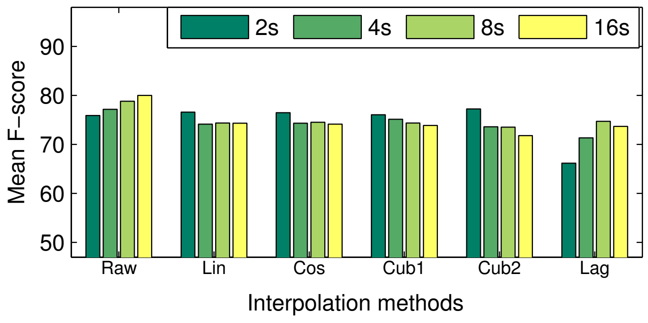
Fig. 4.5 Activity recognition performance for *HOA-Room1* (Raw: $ACC \cup RFID$; other interpolated methods: $ACC \cup RFID \cup INT$)



(a) LSVM



(b) NSVM



(c) CRF

Fig. 4.6 Activity recognition performance for *HOA-Room2* (Raw: $ACC \cup RFID$; other interpolated methods: $ACC \cup RFID \cup INT$)

Table 4.4 Activity recognition performance for both datasets[*]

	LSVM	**NSVM**	**CRF**
HOA-Room1			
Raw	83.27±2.37 (8)	**88.45±1.68** (8)	83.22±3.52 (2)
Lin	84.44±1.90 (4)	85.98±1.25 (4)	***83.73±2.40*** (2)
Cos	83.78±1.09 (4)	85.33±3.69 (2)	83.42±2.21 (2)
*Cub*1	83.70±3.41 (2)	86.17±1.83 (8)	82.39±2.90 (4)
*Cub*2	***84.96±1.23*** (4)	*86.91±3.00* (2)	83.67±1.89 (2)
Lag	83.43±1.57 (2)	84.39±2.19 (8)	81.25±2.66 (2)
HOA-Room2			
Raw	82.28±3.04 (8)	**85.53±2.86**(8)	**79.99±4.76** (16)
Lin	81.92±4.32 (4)	82.97±3.02 (16)	76.61±4.24 (2)
Cos	84.05±6.21 (2)	84.48±2.76 (4)	76.48±3.95 (2)
*Cub*1	80.42±6.27 (2)	*84.86±3.57* (2)	76.08±6.21 (2)
*Cub*2	***84.97±3.47*** (2)	84.36±3.34 (2)	*77.26±3.62* (2)
Lag	79.52±4.33 (4)	82.56±5.90 (4)	74.70±2.83 (2)

[*] Results present the F-score [mean±SD]; the segment size δt for each value is shown in brackets; the highest mean performance for each classifier and room configuration is shown in bold face; the highest mean performance for each classifier for interpolation methods are shown in italics.

Although sensor observation augmentation is necessary to reduce interpolation errors, augmenting a higher number of sensor observations may significantly change the activity patterns represented in the acceleration data. Therefore, increasing the number of augmented sensor observations, as in the case of *Lag*, adversely affect extracted features, subsequently resulting in further decreasing HAR performances.

Interpolation Cost

We consider the time required to interpolate and extract features together in contrast with treating them separately because interpolation is necessary to condition the data stream before extracting additional features (see Table 4.3). We obtain the mean time to generate a feature vector for a single segment based on different segment sizes because the number of data points to be considered for feature extraction increases linearly with the segment size. We have investigated the interpolation methods summarised in Table 4.2. Results in Figure 4.7 are obtained using Matlab (version: 8.2.0.701) on a machine running the Windows 8.1 operating system with 8 GB Random Access

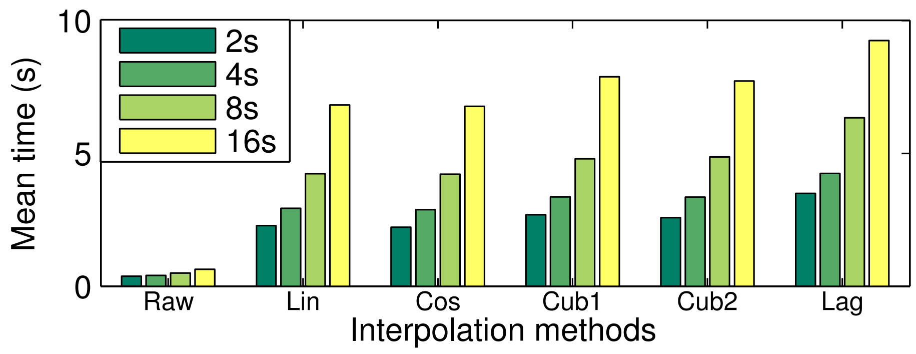


Fig. 4.7 Mean time taken to extract a feature vector using a single segment for each preprocessing method in Table 4.2 for different segment sizes.

Memory and x-64 processor with 4 cores running at 2.40 GHz (Intel Core i7-3630QM @ 2.4 GHz).

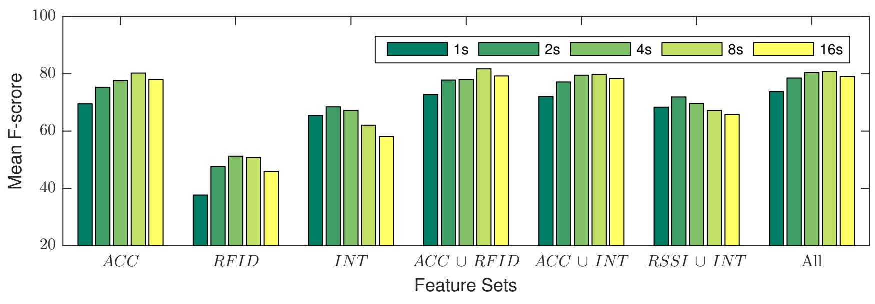
Figure 4.7 shows the mean time taken to generate a feature vector for a single segment using each interpolation method in Table 4.2. As expected, the time taken to generate a feature vector using interpolated acceleration is at least five-fold greater than that of using *Raw* data. Furthermore, increasing the segment size linearly increases the mean time for generating a feature vector, and this is associated with the linear increase in samples with the segment size.

These results (see Figure 4.7) also depict that the mean time taken by each interpolant depends on the complexity of the interpolant. For instance, *Lag* showed the highest mean time taken as it required the most number of data points, i.e. $N = 5$.

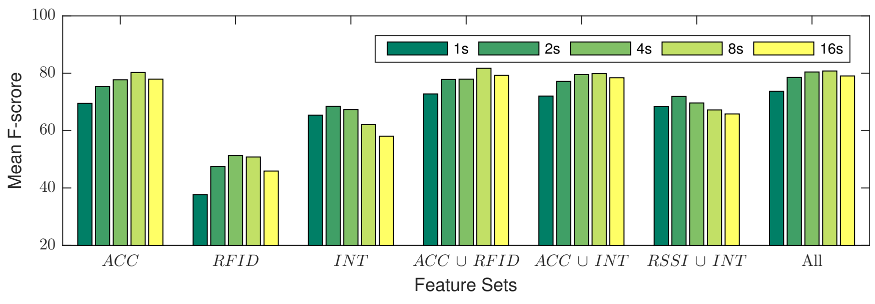
4.5.2 Activity Recognition from Different Feature Sets

Given the significant cost involved with interpolating the acceleration signals, as demonstrated in Section 4.5.1, it is interesting to evaluate the HAR performance using above three sets of features. This experiment is conducted using the $Cub2$ interpolation method as it showed better performance (see Section 4.5.1) among the other interpolates listed in Table 4.2. We utilise all datasets, i.e. *HYA*, *HOA-Room1*, *HOA-Room2* and *FOA*, described in Section 2.4. The dataset *HYA* has been collected using ten healthy young participants using an RFID infrastructure with four RFID antennas. The *FOA* dataset has been collected from frail hospitalised older people and an RFID infrastructure with three RFID antennas. The features used for this experiment are listed in Table 4.5. Based on these features, we evaluated the HAR performance for seven feature set permutations.

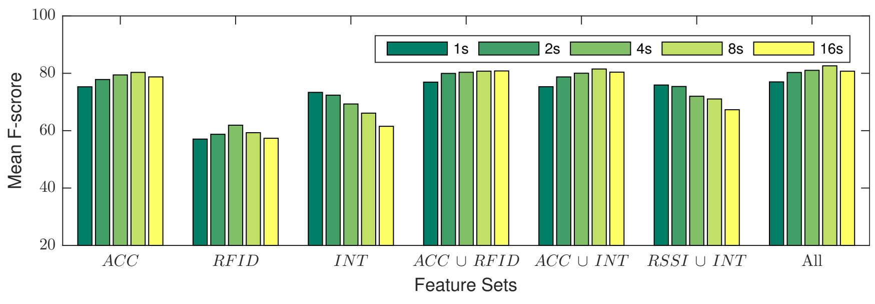
Figures 4.8, 4.9, 4.10 and 4.11 present the activity recognition performance for the datasets *HYA*, *HOA-Room1*, *HOA-Room2* and *FOA* respectively. When the feature sets



(a) LSVM

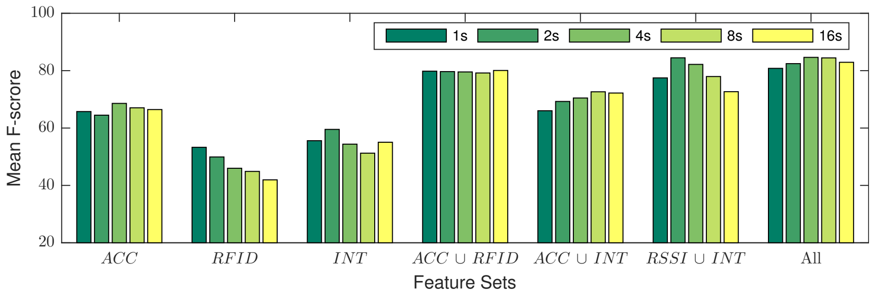


(b) NSVM

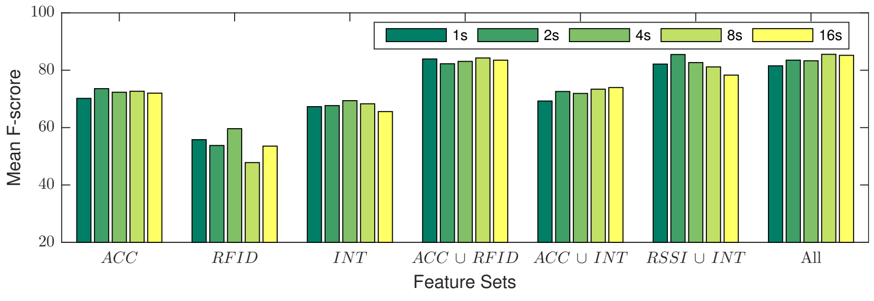


(c) CRF

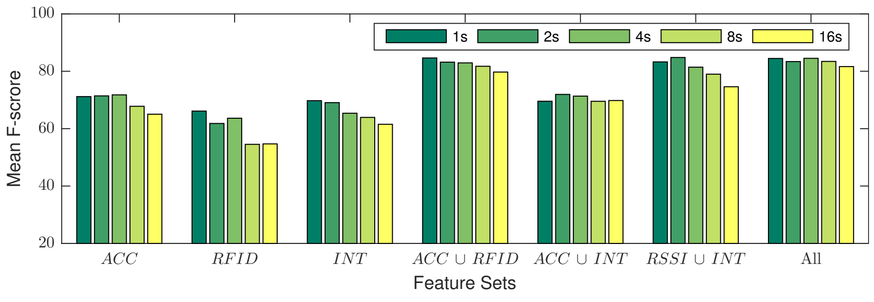
Fig. 4.8 Activity recognition performance for *HYA*



(a) LSVM

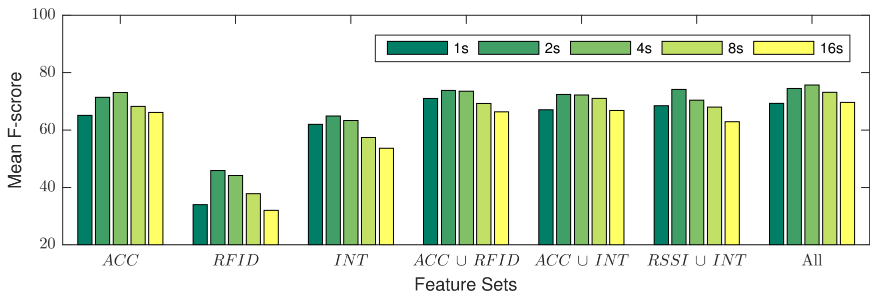


(b) NSVM

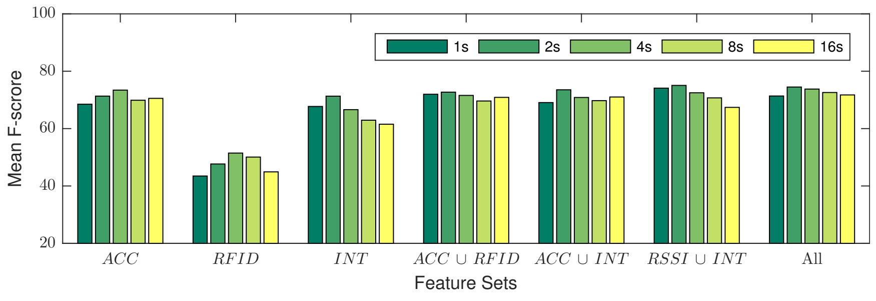


(c) CRF

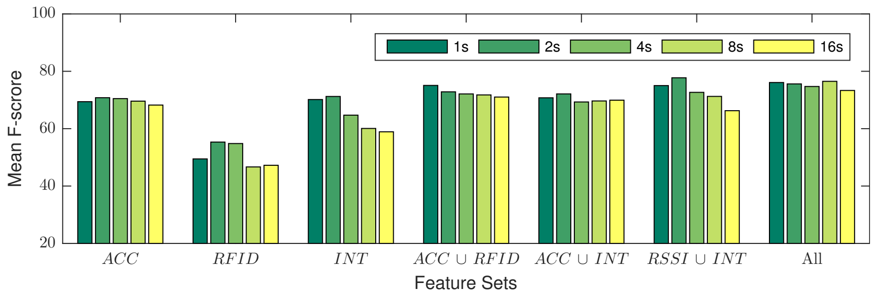
Fig. 4.9 Activity recognition performance for *HOA-Room1*



(a) LSVM

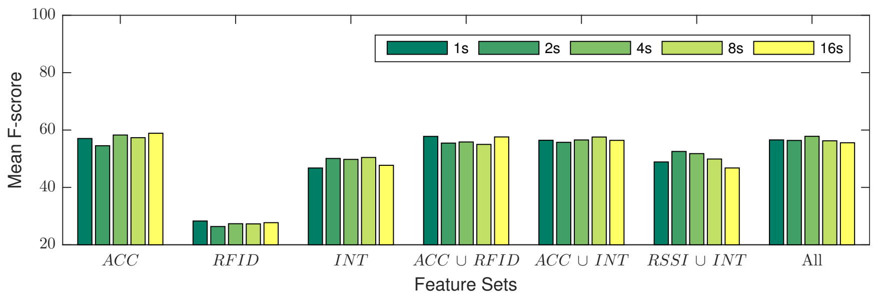


(b) NSVM

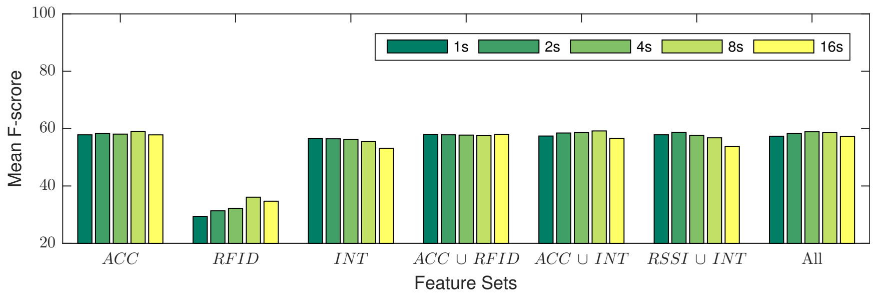


(c) CRF

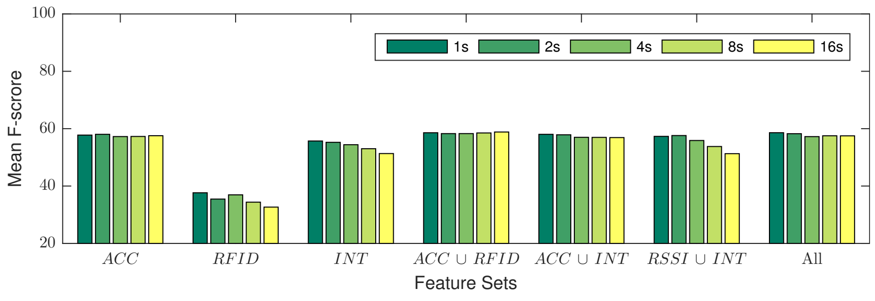
Fig. 4.10 Activity recognition performance for *HOA-Room2*



(a) LSVM



(b) NSVM



(c) CRF

Fig. 4.11 Activity recognition performance for *FOA*

Table 4.5 Features for activity recognition evaluation

Raw Acceleration (*ACC*)	**RFID Platform** (*RFID*)	**Interpolated Acceleration** (*INT*)
a_f	$mean(RSSI_a), a \in A$	$SE(s), s = \{a\_f, a\_l, a\_v, a\_r\}$
a_l	$median(RSSI_a), a \in A$	$SE\_\{band\}(s), s = \{a\_f, a\_l, a\_v, a\_r\}$
a_v	$std(RSSI_a), a \in A$	$PSE(s), s = \{a\_f, a\_l, a\_v, a\_r\}$
θ	$min(RSSI_a), a \in A$	$ARC\_5(s), s = \{a\_f, a\_l, a\_v, a\_r\}$
α	$max(RSSI_a), a \in A$	
β	$range(RSSI_a), a \in A$	
$mean(s), s = \{a_f, a_l, a_v, a_r, \theta, \alpha, \beta\}$	$skewness(RSSI_a), a \in A$	
$median(s), s = \{a_f, a_l, a_v, a_r, \theta, \alpha, \beta\}$	$kurtosis(RSSI_a), a \in A$	
$std(s), s = \{a_f, a_l, a_v, a_r, \theta, \alpha, \beta\}$	$MPM(RSSI_a), a \in A$	
$min(s), s = \{a_f, a_l, a_v, a_r, \theta, \alpha, \beta\}$	miC	
$max(s), s = \{a_f, a_l, a_v, a_r, \theta, \alpha, \beta\}$		
$range(s), s = \{a_f, a_l, a_v, a_r, \theta, \alpha, \beta\}$		
$skewness(s), s = \{a_f, a_l, a_v, a_r, \theta, \alpha, \beta\}$		
$kurtosis(s), s = \{a_f, a_l, a_v, a_r, \theta, \alpha, \beta\}$		
$MPM(s), s = \{a_f, a_l, a_v, a_r, \theta, \alpha, \beta\}$		
$hist(s), s = \{a_f, a_l, a_v, a_r\}$		
$\Delta v_x, \Delta v_y, \Delta v_z, \Delta v_r$		
$\Delta d_x, \Delta d_y, \Delta d_z, \Delta d_r$		

ACC, *RFID* and *INT* are considered, we can see a similar performance between *ACC* and *INT* across all datasets and classifiers. The feature set *RFID* depicted the lowest HAR performance. When feature set combinations are considered, in most instances, increases in HAR performance with respect to individual feature set performances can be observed. In particular, the $ACC \cup RFID$ feature set depicted higher performance than *ACC* or *RFID* when considered individually. A similar observation is made for the $ACC \cup INT$ and $RFID \cup INT$. Among all the datasets, *FOA* generally depicted a lower HAR performance.

The above observations can be closely analysed using Table 4.6, which present the performance for feature set combinations for each classifier. From this table, we can see that only a marginal performance improvement can be achieved by using the features available from the interpolated acceleration (using *Cub*2). For instance, in the *HYA* dataset, NSVM achieved the highest performance at 84.78% using all the features. In this instance, the next best performance (83.08%) was achieved by NSVM with the $ACC \cup RFID$ feature set. In the case of the *FOA* dataset, the highest performance of 58.98% was achieved by NSVM using only time-domain features from the accelerometer.

When each dataset and classier combinations is considered, in nine out of 12 instances a feature set that has features possible from interpolated acceleration signals have shown the highest performance. Furthermore, from Table 4.6, it is observed that for each dataset, the highest performance was achieved by a feature set having *INT*. This indicates that features possible from interpolated acceleration signals provide

Table 4.6 Activity recognition performance for feature set permutations [*]

	LSVM	**NSVM**	**CRF**
YHA			
ACC	80.26±3.30 (8)	82.01±4.95 (8)	80.32±4.21 (8)
RFID	51.24±3.80 (4)	62.48±7.25 (4)	61.91±7.25 (4)
INT	68.46±6.90 (2)	74.54±4.94 (2)	73.36±4.10 (1)
ACC ∪ *RFID*	*81.73±3.16* (8)	83.08±3.90 (4)	80.83±4.47 (16)
ACC ∪ *INT*	79.85±4.22 (8)	82.59±4.61 (8)	81.50±4.22 (8)
RFID ∪ *INT*	71.92±5.63 (2)	77.64±4.74 (2)	75.92±3.54 (1)
All	80.79±3.92 (8)	**84.78±3.29** (4)	*82.61±3.97* (8)
HOA-Room1			
ACC	68.59±9.28 (4)	73.57±7.32 (2)	71.79±6.66 (4)
RFID	53.29±4.91 (1)	59.61±5.40 (4)	66.14±7.48 (1)
INT	59.53±6.08 (2)	69.39±6.08 (4)	69.75±6.17 (1)
ACC ∪ *RFID*	80.07±4.15 (16)	84.28±3.02 (8)	84.65±4.30 (1)
ACC ∪ *INT*	72.64±7.53 (8)	73.97±7.17 (16)	71.94±5.68 (2)
RFID ∪ *INT*	84.48±3.19 (2)	85.48±3.11 (2)	*84.82±2.97* (2)
All	*84.64±2.52* (4)	**85.55±3.81** (8)	84.52±3.13 (4)
HOA-Room2			
ACC	73.03±9.87 (4)	73.42±10.12 (4)	70.79±14.73 (2)
RFID	45.88±6.96 (2)	51.47±7.06 (4)	55.33±12.43 (2)
INT	64.90±10.50 (2)	71.31±9.94 (2)	71.23±15.33 (2)
ACC ∪ *RFID*	73.79±11.46 (2)	72.71±9.88 (2)	75.08±13.62 (1)
ACC ∪ *INT*	72.38±9.27 (2)	73.53±9.34 (2)	72.12±12.82 (2)
RFID ∪ *INT*	74.15±12.71 (2)	*75.06±10.82* (2)	**77.74±13.31** (2)
All	*75.71±8.87* (4)	74.50±12.55 (2)	76.50±11.91 (8)
FOA			
ACC	*58.89±9.08* (16)	58.98±12.04 (8)	58.04±10.68 (2)
RFID	28.29±6.50 (1)	36.08±6.80 (8)	37.65±5.81 (1)
INT	50.44±10.16 (8)	56.51±9.99 (1)	55.70±9.80 (1)
ACC ∪ *RFID*	57.79±9.59 (1)	57.94±9.89 (16)	*58.84±9.43* (16)
ACC ∪ *INT*	57.57±9.73 (8)	**59.19±12.49** (8)	58.03±10.85 (1)
RFID ∪ *INT*	52.54±10.11 (2)	58.70±10.81 (2)	57.61±9.80 (2)
All	57.82±10.98 (4)	58.90±11.80 (4)	58.59±10.81 (1)

[*] Results present F-score [mean±SD]; the segment size δt for each value is shown in brackets; the highest mean performance for each dataset is shown in bold face; the highest mean performances for each classifier for each dataset are shown in italics.

information which are not captured by time-domain features from raw acceleration signals.

4.6 Discussion

In this chapter, we have looked at the features that can be extracted from the sensor tag data stream. In particular, using an example motion sequence, i.e. a participant getting out of bed, we showed that the sensor tag captures motion patterns not only using acceleration signals but also based on the RSSI, which varies with the sensor wearer with respect to a fixed RFID infrastructure. We have also presented how these motion patterns were captured by providing details of the features. Furthermore, we proposed an online sensor data augmentation algorithm to mitigate the issues that arise when interpolants are used to interpolate intermittent data streams from sensor tags. Online interpolation enabled the readily computation of the features requiring data streams with regular sampling rates. We implemented five interpolation methods and evaluated the performance in terms of HAR as well as mean time to obtain a feature vector to be used in real-time HAR. Finally, we evaluated combinations of feature sets that are possible from the raw acceleration signal, interpolated acceleration signal and features available from the RFID platform.

In this chapter, we proposed four novel features. We utilised rotational motion on coronal (α) and transverse (β) planes as features. Previously, researchers have only considered the trunk rotation on the sagittal plane (θ) [62, 61]. However, based on the human motion analysis (see Section 4.2), we observed that the getting out of a bed motion not only involves rotation of the trunk on the sagittal (θ) plane, but also coronal (α) and transverse (β) planes. Therefore, we approximated these rotational motion using the information from the accelerometer. In addition, we utilised maximum prior to a minimum (MPM) as a feature. As explained in Section 4.3.1, this feature can capture posture transitions. Furthermore, the feature *miC*—antenna co-occurrences—was also introduced in this thesis. This feature captures the possible location of a participant within an RFID based monitoring area, such as a hospital room. *miC* was inspired by *miW* (mutual information weighted sensor observation count).

Three sets of features can be considered from sensor tag data streams. Features that can be directly calculated from acceleration information, which captures time-domain patterns present in motion patterns as activities that are performed by the sensor tag wearer. The acceleration data needs to be interpolated to obtain the frequency-domain information and other complex time-domain features such as the ARC but additional preprocessing requires a considerable amount of time compared with calculating features from the raw acceleration signal. The final set of features is from the information from the RFID platform. Although these features are calculated without preprocessing, unlike features from acceleration signals, these features depend on the RFID deployment configuration. There are two main findings from this study. First, when only accelerations from sensor tags are considered, features obtained using interpolation marginally improve the activity recognition performance compared with time-domain features from the raw acceleration data. Secondly, features based on information from an RFID platform have a similar performance improvement for activity recognition and achieve comparable results with activity recognition models built using additional features possible from interpolation. This indicates that features from the RFID platform can successfully substitute the additional features possible from interpolated acceleration data, such as FFT based features, to achieve similar or better activity recognition performance. On the other hand, RFID deployment agnostic activity recognition models can be learnt using features based only on acceleration data, while enjoying the advantages provided by sensor tags but with a considerable real-time prediction delay ($> 400\%$ compared with *Raw*).

Previously, as discussed in Section 2.2.3 several activity recognition studies have conducted using a single wearable devise. Shinmoto Torres et al. [46] have recognised activities on *HOA-Room1* and *HOA-Room2* data sets using CRF classifier. Their they have utilised features a_f, a_v, a_l, $sin(tan^{-1}(\frac{a_f}{a_v}))$, *RSSI*, *aID*, Δt, where Δt is the time difference between sensor observations. Using this method, they were able to achieve F-scores of $65.1 \pm 11.5\%$ for *HOA-Room1* and $71.6 \pm 20.2\%$ for *HOA-Room2*. In contrast, as shown in this chapter, using an extended set of features have shown to improve the activity recognition performance of CRF classifier achieving F-scores of $84.8 \pm 3.0\%$ for *HOA-Room1* and $77.7 \pm 13.3\%$ for *HOA-Room2*. Apart from the study in [46], in this chapter, some of the features used in other studies [62, 83, 35, 44] have been utilised. The work in [62] utilised an empirical algorithm that mainly uses trunk tilt and vertical displacement to recognise posture transitions (sit-to-stand and stand-to-sit) and three daily activities (sitting, standing + walking and lying). Using nine older participants, the method proposed in this study achieved a mean sensitivity of 93.6% and mean specificity of 95.1% for activities. The approach in [83], have utilised frequency domain features such as FFT coefficients and spectral entropy to recognise ten daily activities such as sitting standing, walking, jogging and driving car, and obtained an accuracy of 92% using data collected from two healthy participants. Auto regressive coefficients, tilt angle and signal magnitude area has been used in [35] that achieved an overall accuracy of 97.9% using a data set collected from six healthy participants. Muhammad et al. [44] evaluated the activity recognition performance of features mean, standard deviation, spectral energy, spectral entropy, and correlation between axes using 6 participants. They achieved an accuracy of 85.98%. Although the study presented in this chapter depicted a good activity recognition performance, a direct comparison cannot be made with the existing studies due to the differences in number of activities, sensing devise and experiment setting.

In conclusion, in this chapter, we have presented an evaluation of features that can be extracted from the sensor tag. The main focus of this thesis is on monitoring hospitalised older people. It is expected that given a hospital ward, all the rooms have a similar layout and hence a similar RFID infrastructure deployment. In this context, using features based on information from the RFID platform does not pose a significant impact on activity recognition models. Also, considering the impact of delays with respect to features possible from interpolated acceleration signals on real-time HAR, the remaining chapters of this thesis consider the features extracted from unprocessed acceleration data and the features based on information from the RFID platform for HAR.

Chapter 5

Data Stream Segmentation on Activity Boundaries

5.1 Introduction

Current research on activity recognition using body-worn sensors predominantly relies on battery powered devices. In Chapter 2, we observed that passive sensor enabled RFID tags could be used as an alternative to battery powered sensors. Despite the clear advantages, data streams from passive sensors have two unique characteristics, namely *sparsity* and *noise*, which adversely affect the activity recognition task.

In human activity recognition, streaming sensor data are segmented to provide contextual information to extract features that are descriptive of the current activity [62, 17, 49]. It is common to use fixed sized segmentation methods, such as fixed sample and fixed time segmentation, to segment activity data streams. Nevertheless, we can observe two main issues when fixed size segmentation methods are used to segment sparse data streams from sensor tags. These issues are illustrated in Figure 5.1.

First, the presence of sensor observations from previous activities in a data stream segment can over influence feature information pertaining to the current activity in a data stream segment due to the sparse nature of the data stream [49, 46]. Usually, fixed size segmentation methods assume that the data stream has a fixed sampling rate. Therefore, given a fixed time segment, activity information is uniformly distributed across the sensor data segment. However, in the case of sparse data streams, the above assumption does not always hold true. For instance, as illustrated in Figure 5.1, fixed sample segment S3 and fixed time segment T3 for activity A5 is dominated by sensor observations from activity A4. This may cause the activity recognition performance to deteriorate.

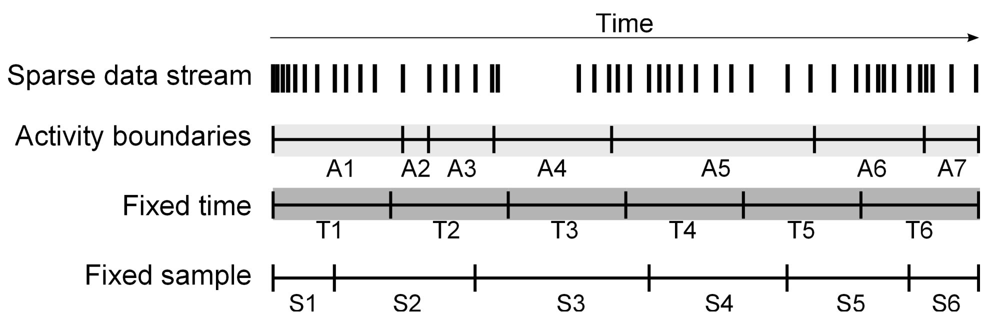


Fig. 5.1 Illustration of different segmentation methods on sparse data streams

Secondly, fixed size segmentation methods cause segment boundaries to misalign with the actual activity boundaries. Segmenting a sensor data stream on activity boundaries leads to improved activity recognition performance [17, 46, 49]. Although this is an issue common to both data streams with and without fixed sampling rates, the resulting adverse effect is larger for sparse data streams, particularly when fixed sample segmentation methods are used. For example, in Figure 5.1 where A2 and A3 are within T2 and S2 imply that using fixed size segmentation methods will lead to missing the transition between activity A2 and A3.

Our overarching goal in the context of this thesis is to recognise ambulatory movements leading to falls in a hospitalised older population. Typically, these movements are activity transitions, such as getting out of the bed and getting out of the chair. Since the activity data stream is sparse, the data stream carries little information on activity transitions. For instance, typically an activity transition takes less than a 2 s duration to complete. Therefore, directly recognising ambulatory movements is challenging.

In this chapter, we focus on segmenting the sparse data stream obtained from a sensor tag worn over the sternum at approximate activity boundaries in real time and later propose an approach to recognise ambulatory movements. Specifically, we propose two data stream segmentation methods based on detecting natural activity boundaries from sensor data streams to overcome the limitations in using conventional data stream partitioning methods (Section 5.3). These schemes are simple, inexpensive, bear no assumptions on sampling rates and rely only on the received sensor observations and are therefore suitable for real-time applications. Then, we propose an ambulatory movement recognition algorithm to identify activity transitions to address the issues posed by inadequate sensor observations (sparsity) to recognise ambulatory movements directly (bed-exits, chair-exits, and walking) (Section 5.5). This algorithm relies on the recognised activities from activity prediction models (Section 5.4) and subsequently, re-evaluating the predictions to identify ambulatory movements. Our algorithm filters predictions from learnt activity prediction models to reduce false alarms resulting from misclassifications caused mainly by noisy sensor

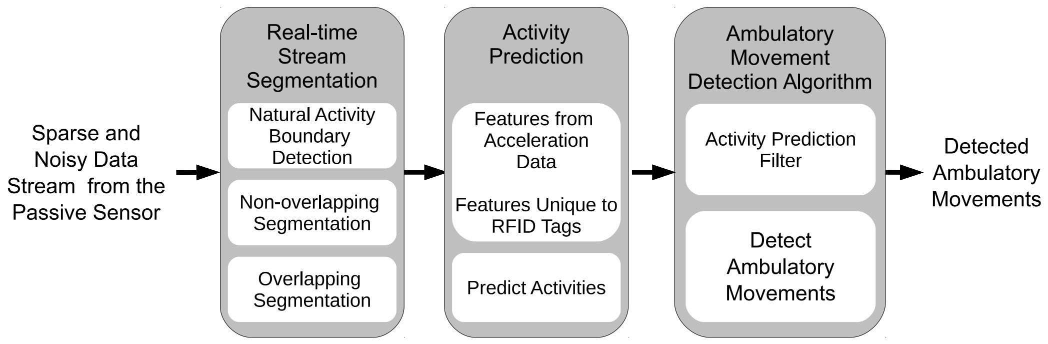

Sparse and Noisy Data Stream from the Passive Sensor
Real-time Stream Segmentation
Natural Activity Boundary Detection
Non-overlapping Segmentation
Overlapping Segmentation
Activity Prediction
Features from Acceleration Data
Features Unique to RFID Tags
Predict Activities
Ambulatory Movement Detection Algorithm
Activity Prediction Filter
Detect Ambulatory Movements
Detected Ambulatory Movements

Fig. 5.2 Ambulatory monitoring framework

measurements. We evaluated our proposed approach using data gathered from a study with ten volunteer participants (Section 5.7). The results presented in this chapter have been published in the IEEE Sensors Journal [30].

5.2 Ambulatory Monitoring Framework

As shown in Figure 5.2, our proposed ambulatory monitoring framework consists of three stages:

1. Real-time segmentation of sparse data streams on approximate activity boundaries that not only leads to extraction of features that are not influenced by previous activities but also ensures that we can make a prediction for each activity.

2. Prediction of activities having sufficient sensor observations, such as in-bed, standing, walking and in-chair, by extracting features based on segments obtained from the first stage increases in activity recognition performance by only predicting activities with sufficient information. Thus activity misclassifications of ambulatory movements, such as transferring out of bed, due to limited sensor observations can be reduced.

3. Re-evaluation of activity predictions to mitigate possible instances of misclassifications and subsequent false alarms caused primarily because of noise in the data stream. Detecting activities such as bed-exits and chair-exits as transitions between predicted activities which otherwise cannot be determined directly.

We elaborate on the proposed framework in detail in the following sections.

5.3 Real-Time Stream Segmentation

The sensor wearer's trunk depicts various levels of inclination or tilt on sagittal (θ) and coronal planes (α) (see Chapter 4). These trunk inclinations can be estimated using the instantaneous acceleration data a_f, a_l and a_v for each sensor observation as $\theta \approx tan^{-1}(a_f/\sqrt{a_l^2 + a_v^2})$ and $\alpha \approx tan^{-1}(a_l/a_v)$.

We represent the i^{th} sensor observation at time t_i, $t_i > t_{i-1}$, $i \in \mathbb{N}$ on the sparse data stream using the pair (t_i, s_i), where $s_i = [a_f, a_l, a_v, RSSI, aID]$ is a 5-tuple obtained from the sensor and the sequence of collected data $\{(t_i, s_i)\}_{i \geq 1}\, i \in \mathbb{N}$ is a non-uniform time series. Given two consecutive sensor observations, s_{i-1}, and, s_i, we define an Activity Boundary Score (ABS) for the sensor observation, s_i, as in Eq.(5.1).

$$ABS_i = |\theta_i - \theta_{i-1}| + |\alpha_i - \alpha_{i-1}| \tag{5.1}$$

The ABS captures the magnitude of changes in θ and α irrespective. During activity transitions (e.g. sitting-on-bed to standing), a sudden increase in ABS, which we refer to as *trunk tilt peaks*, is observed due to rotational movements of a person's trunk. Therefore, trunk tilt peaks are indicative of activity transitions and hence possible activity boundaries.

However, not all trunk tilt peaks correspond to activity boundaries. Our preliminary experiments revealed the feasibility of using the standard deviation of ABS, ABS_{sd}, in a dataset to select trunk tilt peaks that are more likely to be associated with activity boundaries. We defined a model with the condition:

$$(ABS_{i-1} < \lambda ABS_{sd}) \wedge (ABS_i \geq \lambda ABS_{sd}) \tag{5.2}$$

that detects a leading edge of the trunk tilt peak and defines an activity boundary at time t_i where λ is the *segmentation parameter* that controls the sensitivity of the segmentation approach and can be found using cross validation, as discussed in Section 5.6.

Figure 5.3 illustrates activity boundary scores, detecting activity boundaries when $\lambda ABS_{sd} = 0.25$ and the actual activities using our experimental dataset described in Section 5.6. It is noteworthy that a majority of the sensor observations within detected boundaries are from a single activity (class) even though activities have been fragmented (multiple boundaries within the duration of a single activity). Therefore, we can expect the data stream segments from our approach to contain information related to a specific activity as opposed to multiple activities.

The proposed activity boundary detection method can be used for efficient real-time sensor data stream segmentation because it is computationally simple and relies

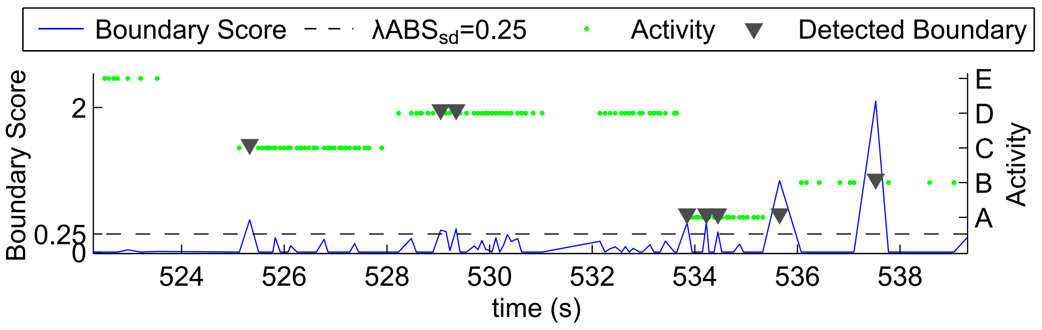


Fig. 5.3 Trunk tilt based activity boundary detection for segmentation (A: sitting-on-bed; B: lying-on-bed; C: standing; D: walking; E: sitting-on-chair)

Algorithm 2 Non-overlapping segmentation

```
Require: $\lambda$, $\delta t_{min}$, $\delta t_{max}$, $(t_i, s_i)_{i \geq 1}$
 1: buffer.clear()
 2: for $i = 0$ to $T$ do
 3:     if isEmpty(buffer) then // start a new segment
 4:         $t_{start} \leftarrow t_i$
 5:     end if
 6:     buffer.add($(t_i, s_i)$)
 7:     if (isActivityBoundary($t_i$,$\lambda$) and ($t_i - t_{start} > \delta t_{min}$)) or ($t_i - t_{start} \geq \delta t_{max}$) then
 8:         $seg_i \leftarrow$ buffer
 9:         buffer.clear()
10:         output $seg_i$ and continue
11:     end if
12: end for
```

on processing individual raw sensor observations, s_i. Furthermore, it is clear from Figure 5.3 that activity boundaries that are detected in close proximity can be merged, and we achieve this by defining a temporal constraint (minimum time interval between boundaries) on segmentation sizes. Using our activity boundary detection method, we define two segmentation methods: i) Non-overlapping; and ii) Overlapping.

Non-overlapping Segmentation (NS) In the non-overlapping segmentation method presented in Algorithm 2, the data stream is partitioned into blocks based on the detected activity boundaries. Therefore, a sensor data stream segment is defined as the sensor observations from a detected activity boundary to the subsequently detected activity boundary. This method may result in not generating a segment for a longer duration, particularly while the sensor wearer is lying on bed. Furthermore, activity boundaries can be detected in within small interval for activities such as walking. As a solution, maximum δt_{max} and minimum δt_{min} segment sizes in terms of time (temporal constraints) can be defined.

Algorithm 3 Overlapping segmentation

```
Require: $\lambda$, $\delta t_{min}$, $\delta t_{max}$, $(t_i, s_i)_{i \geq 1}$
 1: buffer.clear()
 2: for $i = 0$ to $T$ do
 3:     $\delta t \leftarrow 0 \quad k \leftarrow i$
 4:     buffer.add($(t_i, s_i)$)
 5:     while $t_i - t_k < \Delta t_{max}$ do
 6:         if isActivityBoundary($t_k$,$\lambda$) then
 7:             if $t_i - t_k < \delta t_{min}$ then
 8:                 $\delta t \leftarrow \delta t_{min}$
 9:             end if
10:             break
11:         end if
12:         $\delta t \leftarrow t_i - t_k \quad k \leftarrow k - 1$
13:     end while
14:     $seg_i \leftarrow$ buffer.last($\delta t$)
15:     buffer.retainRecent($\Delta t_{max}$)
16:     output $seg_t$ and continue
17: end for
```

In Algorithm 2, sensor observations are collected until either: i) an activity boundary is detected where the segment size is larger than δt_{min}; or ii) the segment size is δt_{max} (line 7 in Algorithm 2). For NS, the delay for is bounded by δt_{max} and taking this into consideration a suitable value for δt_{max} can be selected. Then, δt_{min} specifies the period at which possible activity boundaries near to each other are merged, and a suitable value for δt_{min} can be selected by considering activity transition durations.

The segment size or the level of segmentation can be altered by varying the segmentation parameter (λ) in the boundary detection method. Higher values of λ will result in identification of boundaries that span multiple activities and these will be amalgamated into one activity and subsequently deteriorate the classification performance. Therefore, selection of an appropriate value for λ for the non-overlapping segmentation method is important.

Overlapping Segmentation (OS) The overlapping segmentation (OS) method illustrated in Algorithm 3 generates a segment for each sensor observation, (t_i, s_i), and therefore incurs no delays in the subsequent recognition of the associated activity. In OS, a segment is defined as the sequence of sensor observations from previous activity boundary to the current sensor observations. If the previous activity boundary has been detected prior to δt_{max} from t_i, then sensor observations for the interval $t_i - \delta t_{max}$ to t_i has been considered for the segment. Furthermore, δt_{min} is used to constrain the minimum size of a segment.

Table 5.1 Features extracted from acceleration data

Feature	References
Acceleration signals (a_f, a_l, a_v)	[62, 61, 17]
Vertical Velocity Δv_z, Vertical Distance Δd_z	[62]
Resultant Velocity v_r	[69]
Trunk tilt angle on sagittal plane (θ)	[62, 69, 61]
Trunk tilt angle on coronal plane (α)	

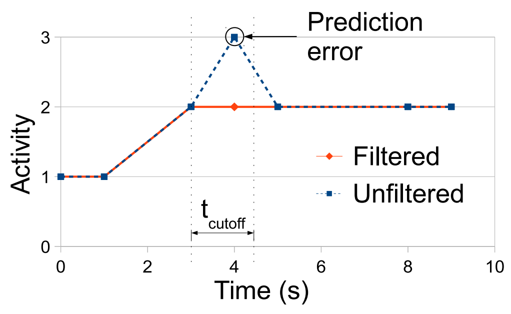
5.4 Activity Prediction

We utilised a subset of features described in Chapter 4. The features based on the acceleration signal are presented in Table 5.1. We have also considered features readily available from RFID tags; RSSI and antenna identifiers (aID) (see Section 4.3.2). For the overlapping segmentation (OS) method, we use the antenna ID (aID) of the most recent sensor observation and the mean of the RSSI values for all the readings from antenna aID within a given segment as features. For the non-overlapping segmentation method (NS), the aID of the antenna with frequent readings within the segment is selected and the mean RSSI is calculated accordingly. As discussed in Section 4.3.2, researchers in [46] have also incorporated activity contextual information using Mutual Information (MI) based features. We have also utilised the miW feature and obtained $|\mathscr{A}|$ number of features, where $\mathscr{A}$ is the set of antennas in the RFID deployment. As in [46], we have also used time differences between segments as a feature. In total we have considered $n = 11 + |\mathscr{A}|$ features from a segment. These features are arranged to form a vector $x \in \mathbb{R}^n$ to describe the activity represented by the segment.

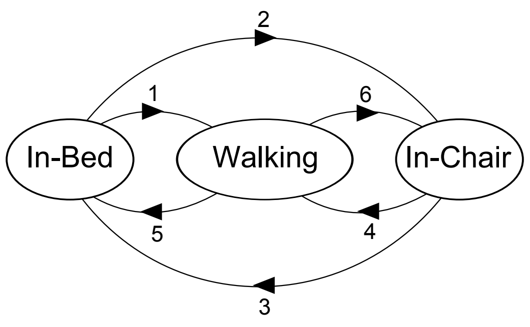
We selected five machine learning algorithms that have been successfully used in AR research: i) Naïve Bayes (NB) [17]; ii) Conditional Random Fields (CRF) [46] iii) Random Forest (RF) [126]; iv) Linear Support Vector Machine (LSVM); and v) Non-linear Support Vector Machine using Radial Basis Function (RBF) kernel (NSVM) [63]. These classifiers have been discussed in Section 2.2.4 and these algorithms differ in how they model the classification problem.

In this study, we have utilised the NB implementation provided in the Matlab (R2014a) environment. As we are interested in real-time prediction, we use the linear chain CRF proposed in [46]. We used libraries, LIBLINEAR [123] and LIBSVM [124], for LSVM and NSVM classifiers. In these libraries, multi-class classification is achieved based on the one-versus-one strategy. We use the RandomForest classifier implementation for Matlab “randomforest-matlab”, which is a ported from the RF implementation for R.[1]

[1]http://code.google.com/archive/p/randomforest-matlab/



(a)



(b)

Ambulatory movement	Transition	Corresponding uninterested movement	Transition
Bed-exit	1, 2	Bed-entry	3, 5
Chair-exit	3, 4	Chair- entry	2, 6
Into-walking	1, 4	Out-of-walking	5, 6

(c)

Fig. 5.4 (a) Illustration of the effect of the filtering parameter; (b) State model; (c) State transitions

5.5 Ambulatory Movement Detection

We propose an ambulatory movement detection algorithm (see Figure 5.2) to detect bed-exits, chair-exits and walking by re-evaluating activity predictions. Our approach solves the problem of highly sparse activity transition information encapsulated in the sensor data stream due to the passive nature of the sensor.

Direct use of activity predictions for ambulatory movement detection can lead to poor performance due to activity prediction errors leading to subsequent false alarms. This may cause alarm fatigue among caregivers and can result in the rejection of the technology [127]. In order to address the activity prediction errors, methods of filtering such as determining prediction consistency using subsequent predictions [18, 62] or the use of Kalman filter [60] have been utilised. These methods depend on more than one correctly predicted activity label. However, when the data stream is sparse and has larger gaps, these methods can result in delays.

Our preliminary studies revealed that most of the activity misclassifications occur around activity transitions such as sitting-on-bed to standing and the first change in an activity prediction is typically the correct prediction for the current activity. We incorporated our findings to moderate the consequences of activity misclassifications on detecting ambulatory movements by defining the prediction filtering function given in Eq.(5.3).

$$f(y_i;t_{cutoff}) = \begin{cases} y_k & ;\text{if } t_i - t_k \leq t_{cutoff} \\ y_i & ;\text{otherwise} \end{cases} \tag{5.3}$$

The parameter t_{cutoff} determines the degree of filtering (see Figure 5.4a), y_i is the current prediction at time t_i, y_k is the previous activity prediction at time t_k, $k < i$, where $y_i \neq y_k$. As illustrated in Figure 5.4a, the filtering function outputs the same activity for a predefined time interval, t_{cutoff}, immediately after a change in the activity prediction stream is observed and thus prevents further changes in activity predictions. The filter leads to fewer erroneous inferences of ambulatory movements. Subsequently, the filtered activity prediction stream is utilised to identify multiple ambulatory movements. A suitable value for the filtering parameter t_{cutoff} can be selected based on activity durations.

In an ideal setting, state changes between *in-bed* and *in-chair* should not be possible as a person needs to walk between the bed and the chair. However, it is possible to misclassify the activities in waking state as those of *in-bed* or *in-chair* and consequently miss an ambulatory movement. Therefore, we have also considered transitions between states *in-bed* and *in-chair* as described in Figures 5.4b and 5.4c, because detecting a bed-exit or chair-exit activity is regarded as transitioning into any state other than *in-bed* or *in-chair*, respectively.

5.6 Experiments

The experiment is conducted using the healthy young adult (HYA) dataset described in Section 2.4.1. This dataset was collected from ten healthy young volunteers aged between 23 and 30 years (mean 26.4 ± 2.12). Each volunteer wore a WISP over the garment at the sternum level and performed three randomly selected activity scripts. In this experiment we consider the five ground truth labels: i) sitting-on-bed; ii) lying-on-bed; iii) standing; iv) walking; and v) sitting-on-chair.

5.6.1 Activity Recognition Performance

In this study, we obtain precision, recall (sensitivity) and specificity. We evaluate performance mainly using the F-score, which is the harmonic mean of precision and recall since our aim is to improve recall (i.e. reduce false negatives) without deteriorating precision (i.e. minimise false positives) in the context of an imbalanced dataset. We further present our results using G-mean, which is the geometric mean of recall and specificity.

The evaluation is carried out using the 10-fold cross-validation strategy and all the parameters are selected based on the validation results. Results are shown as the mean $\pm$ standard deviation (SD). Statistical significance is measured using a two-tailed two-sample t-test at 5 % significance level.

From our preliminary experiments, we observed that a participant took approximately 2 s for a transition and spent at least 4 s in each activity. Considering this we have selected $\delta t_{min} = 1\,s$ (i.e. transition time$/2$) and $\delta t_{max} = 5\,s$ (i.e. $\delta t_{min} +$ activity duration).

5.6.2 Recognising Ambulatory Movements

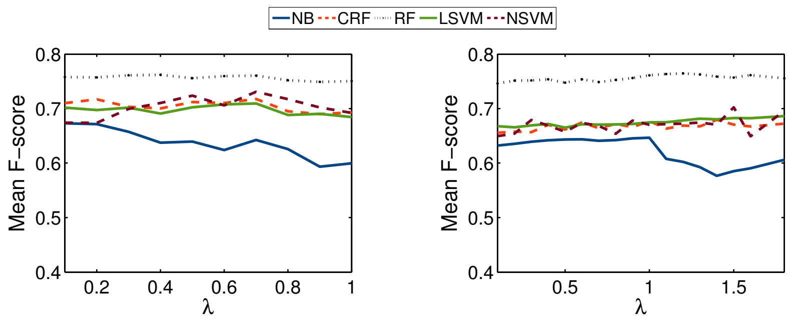
To evaluate the ability of our framework to recognise ambulatory movements shown in Figure 5.4, we define a TP as follows. If there is a corresponding ground truth within evaluation period, δt, after an ambulatory movement is detected (e.g. a bed-exit), the identified ambulatory movement is considered as a TP, otherwise as an FP. Similarly, TN and FN ambulatory movements are identified based on detection of a corresponding uninterested ambulatory movement (e.g. bed-entry) defined by the transitions into a respective state as shown in Figure 5.4b. We utilised the above definitions of TP, TN, FP, FN to obtain the F-score to analyse the performance.

Since a participant's minimum transition time between activities was approximately $2\,s$ and at least $4\,s$ was spent in each activity, it takes at least $8\,s$ for a participant to return to a given activity and hence we took the evaluation time, Δt, to be $8\,s$.

5.7 Results

5.7.1 Activity Recognition Performance

Initially, we investigate a suitable range for the segmentation parameter λ for the NS method, because overly sized segments with NS lead to missing ambulatory movements (see Section 5.3). We have segmented the data stream with $\lambda \in \{i/10\}, i = 1 \cdots 15$, and selected a range that retains more than 95% of each type of ambulatory movement in the segmented data stream with respect to the ground truth. Selection of 95% is based on the fact that we will be able to capture 95% of the ambulatory movements if an activity prediction model is able to predict activities with 100% accuracy. Recognising 95% of ambulatory movement is still a significant achievement compared with reported results for detecting a single ambulatory movement, particularly bed-exits [14, 15, 61].



(a) NS (b) OS

Fig. 5.5 Activity recognition performance (mean F-score) for segmentation methods

From this initial experiment we identified that the NS method was able to retain 95% of each activity for $\lambda \leq 1.0$, which we use in subsequent evaluation.

Figure 5.5 shows the performance (mean F-score) of classifiers with respect to the segmentation parameter, λ. Based on this figure, appropriate values for the segmentation parameter, λ, can be selected for each combination of the segmentation method and classification algorithm. Although the best performance for each classifier was found at different values of λ, the variation in performance observed was $< 8\%$ for each classifier because our segmentation approach successfully partitions sensor observations into segments related to a single activity (see Figure 5.3). However, our results show the importance of selecting an appropriate segment size (i.e. segmentation parameter) for each classifier.

The performance of the best activity prediction models based on the mean F-score for each segmentation method is presented in Table 5.2. Activity prediction models with RF have significantly ($p < 0.05$) outperformed all the other classifiers with both segmentation methods. This result is consistent with findings in [126] where the RF classifier outperformed other classifiers for HAR using acceleration sensor data. The main reason for RF to perform better is the fact that it uses a number of de-correlated DTs internally (Section 2.2.4) and hence it is more resilient to noisy features generated from the WISP data stream. It is important to note that both CRF and NSVM depicted significantly higher ($p < 0.05$) performance when using NS.

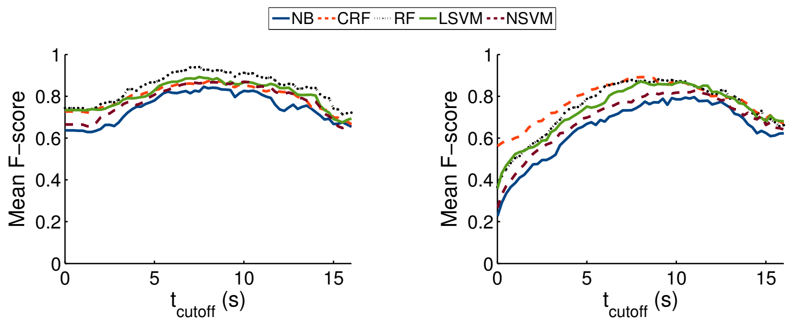
5.7.2 Ambulatory Movement Detection

Ambulatory movement detection is conducted using the best activity recognition models listed in Table 5.2. Figure 5.6 highlights the importance of filtering activity predictions prior to detecting ambulatory movements; performance without filtering is shown at $t_{cutoff} = 0$, where all the classifiers for both segmentation methods depicted their lowest performance. Increasing filtering (i.e. increasing t_{cutoff}), raises ambulatory

Table 5.2 Performance of best activity prediction models in Figure 5.5 based on the mean F-score for the two segmentation methods

	NB	CRF	RF	LSVM	NSVM
NS	$\lambda = 0.1$	$\lambda = 0.7$	$\lambda = 0.4$	$\lambda = 0.7$	$\lambda = 0.7$
F-score	67.29±3.31	71.78±2.46	**76.19±1.22**	70.95±2.66	73.09±1.89
G-mean	78.03±2.49	81.88±1.72	**84.72±1.03**	80.74±1.79	82.68±1.44
OS	$\lambda = 1.0$	$\lambda = 1.4$	$\lambda = 1.2$	$\lambda = 1.5$	$\lambda = 1.8$
F-score	64.66±3.92	67.96±4.68	**76.46±1.01**	70.24±1.63	69.58±3.44
G-mean	77.15±2.71	79.00±3.54	**84.79±0.94**	81.02±0.88	80.46±2.44

* Model parameters - NS:{CRF($\lambda = 10^{-2.2}$); RF(b=500); LSVM(c=4); NSVM(c=2^3, γ=2^2)}; OS:{CRF($\lambda = 10^{-3.5}$); RF(b=1000); LSVM(c=4); NSVM(c=2^6, γ=2^0)}



(a) NS (b) OS

Fig. 5.6 Performance of the ambulatory monitoring algorithm for different segmentation methods with filtering parameter, t_{cutoff} using activity prediction models given in Table 5.2

movement detection performance but further increments in t_{cutoff} deteriorated the F-score since over filtering removes ambulatory movements (i.e. activity transitions) from the activity prediction stream.

From Figure 5.6, we can see that RF has achieved the highest mean performance for NS. Other classifiers except NB depicted similar performances. The results in Figure 5.6 show that in the case of OS, CRF clearly outperformed the other classifiers on ambulatory monitoring, despite CRF being significantly outperformed by RF for activity recognition (see Table 5.2). Since we identify ambulatory movements based on transitions between activities, this observation indicates that although CRF misclassifies activities, CRF correctly predicts longer continuous sequences of activities and results in fewer false activity transitions; consequently decreasing the recognition of false ambulatory movements. The main reason for this observation is that CRF considers the sequential nature of activities during inferencing and the real-time CRF

Table 5.3 Performance of the ambulatory monitoring framework for NS with the RF based activity prediction model and OS with the CRF based activity prediction model.

	Bed-Exit		Chair-Exit		Walking		Mean	
	NS (RF)	OS (CRF)	NS (RF)	OS (CRF)	NS (RF)	OS (CRF)	NS (RF)	OS (CRF)
Accuracy	**92.59±8.66**	90.60±8.57	**93.26±7.38**	91.93±7.23	92.00±5.70	**92.54±2.68**	92.62	91.69
Sensitivity	**94.64±9.96**	93.14±9.63	**93.89±6.45**	92.64±6.35	**93.31±6.16**	93.19±2.02	93.95	92.99
Specificity	**90.60±9.12**	87.96±9.66	**92.64±8.66**	91.21±8.49	90.68±6.04	**91.90±4.24**	91.31	90.36
Precision	**100.00±0.00**	97.78±4.68	**89.08±8.81**	75.73±15.67	**94.50±4.54**	87.82±10.42	94.53	87.11
F-score	**96.98±5.77**	95.03±5.04	**91.22±6.34**	82.48±9.89	**93.80±4.28**	90.09±5.46	94.00	89.20
G-mean	**92.51±8.73**	90.41±8.57	**93.24±7.42**	91.90±7.27	91.96±5.71	**92.52±2.72**	92.57	91.61

[*] Each metric is presented with (mean ± SD) and the metric with the highest mean for each ambulatory movement is in bold face (filtering parameter t_{cutoff} for: NS-7.5 s; OS-8 s)

implementation used in this study benefits from the relatively longer activity sequences produced by OS compared with NS. Furthermore, LSVM has outperformed NSVM using NS in ambulatory movement detection, despite NSVM being the better performing classifier in terms of mean F-score, as shown in Table 5.2. In general, these results are a consequence of the different locations at which prediction errors occurred in their respective activity prediction streams and demonstrated that the best activity prediction model does not always yield the best model for recognising ambulatory movements.

Table 5.3 shows the performance for each ambulatory movement considered using the prediction models for RF with NS and CRF with OS. The NS (with RF) based approach clearly provides the highest mean F-score for all ambulatory movements. For chair-exit, NS depicts a significantly higher ($p < 0.05$) F-score than OS. It is also important to note that high precision for NS (>89%) compared with OS (>75%) for each ambulatory movement is indicative of fewer false positives and consequently lower false alarms. However, both segmentation methods yield higher mean sensitivity (or recall) values indicating that the proposed ambulatory monitoring framework was able to capture over 92% of the ambulatory movements recorded in the data stream and thus less than 8% of ambulatory movements were missed on average.

In contrast with NS, the relatively lower performance of OS is because of the limited sensor observations related to the current activity available for generating features when the current sensor observation is closer in time to the previous activity boundary. OS contain partial activity information, i.e. only samples up to the time of the current sample from the last activity boundary or Δt_{min} at the start of segments, as opposed to the complete information related to an activity available through segments with NS. The limited number of samples in segments at the start of activities affects the quality of information in features and subsequently the learnt activity prediction model and predictions. For example, features such as vertical displacement obtained by integrating acceleration vectors are greatly influenced by the number of sensor observations related to an activity within a segment.

5.8 Discussion

In this chapter, we have proposed two data stream segmentation method for activity recognition using sparse data streams available from sensor tags. We also proposed a simple and efficient approach to recognising ambulatory movements.

Despite the challenging nature of the data stream from sensor tags (i.e. sparsity and noise), our proposed ambulatory monitoring framework was able to successfully recognise multiple ambulatory movements (bed-exit, chair-exit and walking) in real time (mean F-score: NS 94% and OS 89%). In terms of the segmentation approaches, they have their strengths and weaknesses; NS performs better (overall) but is less responsive, and OS is highly responsive but depicts a lower performance. Therefore, depending on the application context, a suitable segmentation approach can be selected. Nevertheless, for bed-exit recognition, OS is preferable as it performs as well as NS while being more responsive than NS (see Table 5.3).

We proposed the use of the Activity Boundary Score (ABS) to identify possible activity boundaries. Although we only considered the trunk orientation of the sensor wearer to calculate the ABS, different motion parameters can be used depending on the application scenario, available sensors and sensor attachment position. For instance, in a gesture recognition scenario as in [48] that uses sensors attached to arms, both acceleration and rotational motion from an accelerometer and a gyroscope may be used to calculate the activity boundary score and subsequently segment the data stream.

The major advantages of our framework in terms of practical usage and specially for acceptance of the technology by caregivers and clinicians are: i) the use of a lightweight batteryless sensor that is low cost and maintenance free; ii) highly accurate recognition of multiple ambulatory movements with very low misses and false alarms; and iii) low latency to alarm (maximum delay with NS is bounded by the maximum segment size constraints Δt_{max} while OS will generate an alarm immediately). Recognition of multiple ambulatory movements will deliver a comprehensive fall prevention mechanism and provide caregivers with the ability to select appropriate ambulatory movements to monitor based on falls risk assessments of patients carried out daily as a part of best practice guidelines for fall prevention [9].

More significantly, the batteryless ambulatory monitoring system has demonstrated performance comparable with previous studies based on battery powered devices attached to the waist [34] or strapped to the chest [62]. Although movement sensor alarm systems that consider bed-exits and chair-exits [128] have been developed, we have only found research studies that have reported results for bed-exit alarm systems [14, 15, 61]. These approaches have also been evaluated using similar study groups, i.e. healthy adults. Table 5.4 presents the performance of previous bed-exit movement

Table 5.4 Performance of previous bed-exit movement alarms

Bed-exit alarm system	Sensitivity	Specificity	Precision	Participants' age
Hilbe et al. [14]	96%	96%		18-60
Bruyneel et al. [15]	91%	100%	100%	37±9 and 45±11
Ranasinghe et al. [61]	90%	94%		26.4±2.12
Proposed framework	94%	91%	100%	26.4±2.12

alarm approaches where we can see that the proposed framework performs comparably or better.

However, this study is not without limitations. Results for activity recognition demonstrate that the existing features for acceleration and RFID tag data may be inadequate to discriminate the activities successfully as shown by the relatively low F-score (see Table 5.2). Therefore, future work should consider more features to discriminate these activities. Furthermore, our evaluation is based on data collected from healthy young adults (mean age 26.4±2.12), which is not representative of the target group of older people. Even though body motions of older frail patients in performing activities follow that of young people, their activity durations may be different and hence our framework needs to be evaluated with data collected from older people. During experiments (Section 5.7.2), one walking activity was not captured due to the passive nature of the sensor; mainly the RFID tag was not adequately powered to sample the accelerometer. This limitation can be addressed by: i) incorporating an appropriate antenna design that is tailored to maximise energy harvesting while the sensor is worn by a person as recently demonstrated by our group in [24]; and ii) using multiple emitters to power tags in the presence of a single receiver as described in [129]; or iii) using hybrid powered sensor tags as recently demonstrated in [130].

In summary, in this chapter we have looked at the use of existing classification algorithms to recognise activities from sparse data streams using features extracted based on segments obtained from approximate activity boundaries in real time. As our main goal within this thesis is ambulatory monitoring, we also proposed an ambulatory movement recognition algorithm that re-evaluates the predictions from the activity classifiers to recognise ambulatory movements. In fact, the segmentation is the first step in machine learning based activity recognition. In Chapter 4, we looked at features from sensor tags. In the next two chapters, we look at novel classification approaches to classify streaming sparse sensor data in real time.

Chapter 6

Sequential Support Vector Machine for Sparse Data Streams

6.1 Introduction

In previous chapters, we utilised classical machine learning algorithms to recognise activities in real-time using passive sensor enabled RFID tags and subsequently recognise ambulatory movements which can leads to falls. We have also observed in Chapter 2 that the data streams from these sensors are sparse, and they are a non-uniformly sampled time series. In Section 4.4, we investigated the use of interpolation techniques on acceleration data available from the sensor to obtain a uniform time series and subsequently extract features to recognise activities. There we showed that this interpolation bears significant cost in real-time processing, although we can achieve good performance using acceleration data alone.

Previously, bed-egress detection system, named Bed Exit Alarm System (BEAS) has been proposed in [25]. BEAS employed linear chain Conditional Random Fields (CRF) to classify segments of sensor observations and subsequently identify bed-egress events as traditions from lying-on-bed or sitting-on-bed to out-of-bed. However, due to the linear chain CRF implementation used, this method does not support real-time inferencing.

Sequence learning algorithms, such as linear chain CRF learns relationships between the i^{th} and $(i+1)^{th}$ elements in a sequence; in other words, these algorithms model a sequence as a first order Markov chain [31]. Hence, regular sequential learning algorithms' reliance on a single previous sensor observation (which might belong to an activity in the distant past) may result in poor recognition performance. Therefore, modelling the sequential nature of activities using such data streams to recognise activities should consider the non-uniform nature of the underlying data stream.

On the other hand, researchers have also looked at stochastic modeling of RFID data streams using techniques such as particle filters [131] and the Partially Observable Markov Decision Process (POMDP) [132] to address the uncertainty in RFID tag observations, mainly due to the random access nature of ISO 18000-6C protocol [26]. In contrast to our goal, which is to assign a class label to each sensor observation from the sensor tag, their goal is to track RFID tags, i.e. to ascertain the location of a given tag at a given point of time when it cannot be observed by an RFID infrastructure.

In this chapter, we present a sequence learning algorithm, based on the support vector machine (SVM) framework, suitable for sparse accelerometer and RFID data from sensor tags and propose a monitoring framework that combines a novel sequence learning algorithm to recognise bed-exit movements in real time. In particular, the proposed sequence learning algorithm considers recent past sensor observations and their activity labels to model the sequential nature of activities present in sparse data streams (Section 6.3). The proposed framework utilises accelerometer and RFID data while exploiting only time domain features to support rapid feature calculations (Section 6.4). We further propose four types of features specifically to capture the sequential nature of bed-exit motion from recent activity label history (Section 6.5). The implementation of the sequence learning algorithm training is implemented following the Pegasos algorithm [90], which solves the SVM using stochastic sub- gradient method. Finally, we evaluate the performance of our approach using a dataset collected from 14 older volunteers (Section 6.6). The work presented in this chapter is published in the IEEE Journal of Biomedical and Health Informatics [32].

6.2 Bed-Exit Monitoring Framework

Human motion analysis of a participant described in Section 4.2 provides the basis for formulating the bed-exit monitoring framework. There, we identified that participants may lift their trunk to a vertical position using their dominant arm when they are lying in the supine posture and then turn their body facing the edge of the bed to place their legs on the ground. On the other hand, they can roll over to the edge of the bed and then use the weight of the legs and arm to raise their trunk to a vertical position. When the sit-to-stand posture transition is considered, participants first bend forward and then stand up. Thus, bed-exit activity involves a sequence of movements which are reflected in the sensor observations (see Figure 4.2). In this movement sequence, we are interested in transitions from *in-bed* to *out-bed*. Therefore, we formulate bed-exit recognition as:

First predicting activity label sequence, *in-bed* or *out-bed*, in real time for a sequence of sensor observations;

Then determining changes from *in-bed* to *out-bed* in the predicted activity label sequence to recognise bed-exit movements.

For this formulation, we have considered the entire bed-exit motion including possible variations observed to capture the contextual information of a bed-exit (see Section 6.4). The remaining sections of the paper discuss the proposed framework in detail.

6.3 Bed-Exit Sequence Prediction

Prediction of a sequence of activity labels (*in-bed* or *out-bed*) from a sensor observation sequence can be modeled as a sequence learning problem. To this end, we follow the deterministic approach proposed for sequential learning in [133] and propose the following sequence learning algorithm suitable for sparse data streams based on the Support Vector Machine (SVM) classifier. As opposed to considering a fixed number of elements in a sequence, we consider elements in a fixed time interval to model the sequential relationship; this approach mitigates past observations that belong to an activity in the distant past from adversely affecting the prediction of the current activity. In the following, we describe the two main steps of the algorithm, i.e. inferencing and training.

6.3.1 Representation and Inference

We denote the i^{th} sensor datum at time t_i, $t_i > t_{i-1}$, $i \in \mathbb{N}$, using the pair (t_i, s_i), where $s_i = [a_f, a_v, a_l, RSSI, aID]$. Because of the irregular nature of the data collection, the sequence of collected data $\mathbf{X} = \{(t_i, s_i)\}_{i \geq 1}$, $i \in \mathbb{N}$ is a non-uniformly sampled time series. We denote the corresponding sequence of activities for each sensor observation using $\mathbf{Y} = \{y_i\}_{i \geq 1}$, $y_i \in \mathscr{C}$, where $\mathscr{C}$ is all the possible set of class labels. In this chapter we consider only two class labels, $\mathscr{C} = \{in\text{-}bed, out\text{-}bed\}$. We use subscripts to denote subsequences; for instance $\mathbf{X}_{[a,b]}$ denotes the sensor data stream segment for time interval $[a,b]$, where $b > a$. The notation $\langle \cdot, \cdot \rangle$ is used to represent the vector inner product.

Given a sensor observation sequence, $\mathbf{X}$, the inference procedure assigns the class label, y_i to each corresponding sensor observation, (t_i, s_i) considering correlations between most recent sensor observations and labels for a fixed duration of δt; i.e. considering $Y_{[t_i - \delta t, t_i]}$ and $\mathbf{X}_{[t_i - \delta t, t_i]}$. In other words, when determining the class label for the i^{th} sensor observation (t_i, s_i), a sensor observations for fixed interval δt, i.e. $X_{[t_i-\delta t, t_i]}$, and the assigned class labels for those sensor observations excluding the label for the i^{th} sensor are used. Use of previous class labels in the sequence essentially captures the sequential nature. Since we are interested in real-time prediction, we specifically focus on the greedy inference procedure discussed in [133]. We assume here that the predictions already made are frozen, and a score function exists that calculates a score for assigning the class label y for the i^{th} sensor observation:

$$S_i(\mathbf{w}, \mathbf{X}, \mathbf{Y}, y) = \langle \mathbf{w}, \phi(\mathbf{X}_{[t_i-\delta t, t_i]}, \mathbf{Y}_{[t_i-\delta t, t_{i-1}]}, y) \rangle \tag{6.1}$$

where $\mathbf{w} \in \mathbb{R}^d$ is a learnt model also known as parameter vector and $\phi : \mathbf{X}_{[t_i-\delta t, t_i]} \times \mathbf{Y}_{[t_i-\delta t, t_{i-1}]} \times y \mapsto \mathbb{R}^d$ is the feature mapping function that captures the sequential information in a single feature vector.

The inference procedure determines successive class labels, y_i, based on maximising the score function Eq.(6.1); i.e. $\arg\max_{y \in \mathscr{C}} S_i(\mathbf{w}, \mathbf{X}, \mathbf{Y}, y)$. In fact, as we have frozen the previous predictions, the prediction function can be represented as a recursive function:

$$f_i(\mathbf{w}, \mathbf{X}) = \arg\max_{y \in \mathscr{C}} \langle \mathbf{w}, \phi(\mathbf{X}_{[t_i-\delta t, t_i]}, f_{[t_i-\delta t, t_{i-1}]}(\mathbf{w}, \mathbf{X}), y) \rangle. \tag{6.2}$$

where $f_{[a,b]}(\mathbf{w}, \mathbf{X})$ represents the predictions from time a to time b.

The feature function, ϕ, in our sequence prediction algorithm, is significant as it essentially captures patterns in the data stream. This feature function can be considered as a concatenation of individual features extracted considering the considered class label, y, observed information $X_{[t_i-\delta t, t_i]}$ and previous labels $Y_{[t_i-\delta t, t_{i-1}]}$. We specifically considered a feature function of the form $\phi(\mathbf{X}_{[t_i-\delta t, t_i]}, \mathbf{Y}_{[t_i-\delta t, t_{i-1}]}, y) = [\phi_I(\mathbf{X}_{[t_i-\delta t, t_i]}, y), \phi_P(\mathbf{Y}_{[t_i-\delta t, t_{i-1}]}, y)]$ where ϕ_I calculates features based on the considered class label, y, with observed information $\mathbf{X}_{[t_i-\delta t, t_i]}$ (Section 6.4), and ϕ_P considers the label sequence $\mathbf{Y}_{[t_i-\delta t, t_{i-1}]}$ and y (Section 6.5) to calculate features.

6.3.2 Training

The goal of the classifier training is to find the model w that minimises the classification error. According to the prediction function Eq.(6.2), we can observe that the score for a given sensor observation with the correct class label $S_i(\mathbf{X}, \mathbf{Y}, y_i)$ must be greater than the score of all other class labels $S_i(\mathbf{X}, \mathbf{Y}, y), y \in \mathscr{C} \setminus y_i$, i.e. $S_i(\mathbf{X}, \mathbf{Y}, y_i) > S_i(\mathbf{X}, \mathbf{Y}, y)$. In the SVM setting, $S_i(\mathbf{X}, \mathbf{Y}, y_i)$ should be at least greater than a margin, which is typically set to 1 [134]. Consequently, the updated constraint for minimising the error for SVM training is $S_i(\mathbf{X},\mathbf{Y},y_i) \geq S_i(\mathbf{X},\mathbf{Y},y)+1$.

Notably, as indicated by the nature of the feature function, two instances of output vectors, $\phi(\mathbf{X}_{[t_i-\delta t,t_i]},\mathbf{Y}_{[t_i-\delta t,t_{i-1}]},y)$ and $\phi(\mathbf{X}_{[t_j-\delta t,t_j]},\mathbf{Y}_{[t_j-\delta t,t_{j-1}]},y)$, where $i \neq j$, can be considered independent of each other. This characteristic enables formulating efficient learning algorithms as described in [133]. In this setting, labelled sensor data stream segments from multiple sequences can be considered for learning, unlike in other sequence learning algorithms.

The primal form of soft margin SVM formulation for the proposed sequence learning algorithm followed the formulation of soft margin SVM presented in the work entitled "Large Margin Methods for Structured and Interdependent Output Variables" [134]. Specifically, we have utilised the slack rescaling method in [134] to incorporate different costs for misclassifications. The classification model, w, is learnt by minimising the following constrained convex objective function Eq.(6.3).

$$\begin{aligned}
\min_{\mathbf{w},\xi_i} \quad & \frac{\lambda}{2}\|\mathbf{w}\|^2+\frac{1}{M}\sum_{i=1}^{M}\xi_i \\
\text{subject to} \quad & \forall i\ \langle \mathbf{w},\phi_i(y_i)\rangle - \langle \mathbf{w},\phi_i(y_r)\rangle \geq 1-\frac{\xi_i}{\Delta(y_i,y_r)} \\
& \forall i\ \xi_i \geq 0 \\
\text{where} \quad & \phi_i(y)=\phi(\mathbf{X}_{[t_i-\delta t,t_i]},\mathbf{Y}_{[t_i-\delta t,t_{i-1}]},y) \\
& y_r = \operatorname*{arg\,max}_{y\in\mathscr{C}\setminus y_i}\langle \mathbf{w},\phi_i(y)\rangle
\end{aligned} \tag{6.3}$$

In Equation Eq.(6.3), M is the number of samples in the training dataset, λ is the regularisation parameter which is treated as a model parameter, and $\Delta(y_i,y_r)$ is a cost function which assigns different costs based on the nature of misclassifications. Any margin violations are treated as positive errors that are represented by ξ_i; as a result the value of the objective function is increased, which subsequently updates the classification model to reduce the objective value.

We consider cost functions of the form:

$$\Delta(y_i,y_r)=\begin{cases} 1 & ;if\, y_i = in\text{-}bed \wedge y_r = out\text{-}bed \\ c & ;otherwise \end{cases} \tag{6.4}$$

where c is the relative cost of misclassifying a *out-bed* instance as an *in-bed* instance and c is also considered as a model parameter.

We use the Pegasos algorithm [90] to solve the optimisation problem in Eq.(6.3) directly, using its primal form. Unlike other methods such as LaRank [133] which optimises SVMs in dual form, optimising in primal form is attractive where there are a large number of training samples, as in our case. When using the Pegasos algorithm, two additional parameters need to be specified; i) number of iterations, T and ii) mini-batch size, k. In total, the proposed bed-exit sequence prediction algorithm requires four model parameters, i.e. λ, c, T and k. These model parameters are selected using cross-validation as described in Section 6.6.

6.4 Features based on Sensor Observation Sequences

As discussed in Section 2.3.2, we have two information sources in our data stream: i) acceleration signals; and ii) information from the RFID platform. Using these two information sources, we showed in Chapter 4 that a number of features can be extracted. Features based on acceleration used in this chapter are presented in Table 6.1, whereas features based on RFID information are listed in Table 6.2. These features are calculated based on a segment of sensor observations for i^{th} sensor observation, $\mathbf{X}_{[t_i-\delta t, t_i]}$ and we set $\delta t = 2\,\text{s}$ to calculate these features. Although we have summarised these features in this section, they have been described in detail in Chapter 4.

6.4.1 Features based on Acceleration

We obtain nine features using the i^{th} sensor observation ($\{1, \cdots, 9\}$ in Table 6.1). These features include acceleration signals, estimated orientation of the sensor, and approximated acceleration signals on anteroposterior (a_x), mediolateral (a_y) and dorsoventral (a_z) axes. We extract 37 features based on a sensor data stream segment $\mathbf{X}_{[t_i-\delta t, t_i]}$ ($\{10, \cdots, 46\}$ in Table 6.1). We have considered common time domain statistical features from acceleration signals and approximated orientation. We also consider the feature *MPM* (Maximum Prior to Minimum) for each acceleration axis and each angle ($\{19, 20, 21\}$ and $\{44, 45, 46\}$ in Table 6.1), as they are indicative of posture transitions which are discussed in Chapter 4. We have also used the change in displacements (Δd) and velocities (Δv) in x, y and z directions relative to the body. We further utilise the change in resultant velocity (v_r).

6.4.2 Features based on the Information From RFID Platform

We obtain eight types of features using information from the RFID platform. We calculate the time-domain statistical features of the RSSI for a segment ($\{1, 2, 3\}$

Table 6.1 Features from acceleration signals

Feature #	Feature
1	a_f - Frontal acceleration
2	a_l - Lateral acceleration
3	a_v - Vertical acceleration
4	θ - $\measuredangle$ on Sagittal plane [62, 69, 46, 61]
5	α - $\measuredangle$ on Coronal plane
6	β - $\measuredangle$ on Transverse plane
7	a_x - Anteroposterior acceleration (using θ, a_f and a_v)
8	a_y - Mediolateral acceleration (using α, a_l and a_v)
9	a_z - Dorsoventral acceleration (using θ, α, a_f, a_l and a_v)
10, 11, 12	$mean(a_f)$, $mean(a_l)$, $mean(a_v)$ [17, 46, 63]
13, 14,15	$max(a_f)$, $max(a_l)$, $max(a_v)$ [63]
16, 17, 18	$min(a_f)$, $min(a_l)$, $min(a_v)$
19, 20, 21	$MPM(a_f)$, $MPM(a_l)$, $MPM(a_v)$ - Maximum Prior to Minimum
22, 23, 24	$corr(a_f,a_l)$, $corr(a_f,a_v)$, $corr(a_l,a_v)$ [17]
25, 26, 27	$mean(a_x)$, $mean(a_y)$, $mean(a_z)$
28, 29, 30	Δv_x, Δv_y, Δv_z [62]
31, 32, 33	Δd_x, Δd_y, Δd_z [62]
34,	Δv_r [69]
35, 36, 37	$mean(\theta)$, $mean(\alpha)$, $mean(\beta)$
38, 39, 40	$max(\theta)$, $max(\alpha)$, $max(\beta)$
41, 42, 43	$min(\theta)$, $min(\alpha)$, $min(\beta)$
44, 45, 46	$MPM(\theta)$, $MPM(\alpha)$, $MPM(\beta)$ - Maximum Prior to Minimum

in Table 6.2). To capture patterns using relative RSSI, we consider a $3\delta t = 6\,\text{s}$ segment, $\mathbf{X}_{[t_i-3\delta t, t_i]}$, subdivided into three equal sub-segments based on time ($\mathfrak{S}_j = \mathbf{X}_{[t_i-j\delta t, t_i-(j-1)\delta t]}, j = 1\dots3$). Using these sub-segments, we extract mean RSSI values and features comparing the relative magnitudes of sub-segments ($\{4,5\}$ in Table 6.2).

The RFID antenna facing the sensor is most likely to both power and collect the response from the sensor tags. We use this fact to obtain an approximation of the sensor wearer's location using the feature *ant* $\in \{0,1\}^{|\mathscr{A}|}$, where $\mathscr{A}$ represents the set of RFID reader antennas in the dataset. The feature *ant* indicates the antennas present in a sensor observation segment. We further use *miW* and *miC*, which are based on mutual information obtained from antenna ID.

6.5 Features based on Label Sequences

These features are designed specifically to capture the transition of activities during a bed-exit motion. We specifically consider four features (ϕ_{pj}, $j = \{1\cdots4\}$) from previous labels $\mathbf{Y}_{[t_i-\delta t, t_{(i-1)}]}$ and the current class label y (see Table 6.3). For calculating this set of features, we set $\delta t = 2\times4 = 8\,\text{s}$ to obtain sufficient information relating to transition of activities as we have identified that a participant took approximately $2\,\text{s}$ to get out of the bed while seated during our preliminarily experiments.

Table 6.2 Features from the RFID platform

Feature #	Feature
1	$mean(RSSI_a),\, a \in \mathscr{A}$ [87]
2	$max(RSSI_a),\, a \in \mathscr{A}$ [87]
3	$min(RSSI_a),\, a \in \mathscr{A}$ [87]
4 †	$mean(RSSI_{a_j})\ a \in \mathscr{A}, j = 1 \cdots 3$
5 †	$mean(RSSI_{a_j}) > mean(RSSI_{a_{j+1}})\ a \in \mathscr{A}, j = 1,2$
6	$ant_a = \mathbf{1}_{[a \in S_i(aID)]},\, a \in \mathscr{A}$ [46]
7	$miW \in \mathbb{R}^{\lvert\mathscr{A}\rvert}$ [46]
8	$miC \in \mathbb{R}^d,\, d = {}_{\lvert\mathscr{A}\rvert} C_2$

† $RSSI_{a_j}$: RSSI values received for antenna a for the sub-segment $\mathfrak{S}_j$

The relationships between y and 3 previous class labels is considered as a feature, $\phi_{p1} \in R^{3|\mathscr{C}|}$. We partition the output vector from ϕ_{p1} into 2 and, depending on y, the previous 3 class labels are assigned to corresponding partitions. Given the last three class labels, $Y_{\{i-3,i-1\}}$, feature ϕ_{p1} is calculated as

$$\phi_{p1} = \begin{cases} \{\mathbf{0}_3, Y_{\{i-3,i-1\}}\} & ; if\ y = out\text{-}bed \\ \{Y_{\{i-3,i-1\}}, \mathbf{0}_3\} & ; otherwise \end{cases} \tag{6.5}$$

where $\mathbf{0}_3$ is a 1×3 zero vector and $\{\cdot,\cdot\}$ represents the concatenation of vectors.

The time-weighted count of labels is also used as a feature ($\phi_{p2} \in R^{|\mathscr{C}|}$). We assign the highest weight to the class label at time t_i and the weights are decreased linearly to 0 at time $t_i - \delta t$. Weight at time t, $t_i - \delta t \leq t \leq t_i$, is given by $\tau_t = (t - t_i + \delta t)/\delta t^2$ such that $\int_{t_i-\delta t}^{t_i} \tau_t = 1$. The value of the feature for the jth class $\phi_{p2}(j)$ is obtained as

$$\phi_{p2}(j) = \sum_{k=1}^{|Y'|} \mathbf{1}_{[Y'_k = j]} \tau_{t_k}\ , j = \{in\text{-}bed, out\text{-}bed\}\,. \tag{6.6}$$

Using this approach, emphasis is given to the recent class labels with respect to the ith observation.

Given a sufficiently small interval δt, typically, participants do not transition between activities more than once, i.e. they do not get out from and get into a bed in quick succession. Therefore, we consider the number of changes in activity labels as a feature (ϕ_{p3}). As there are two classes in our setting, i.e. *in-bed* and *out-bed*, this feature can be represented by $\phi_{p3} \in R^{d'}$ where $d' = {}_{|\mathscr{C}|}P_2$.

Table 6.3 Features from label sequences

Feature #	Feature
1	$\phi_{p1} \in R^{3\lvert\mathscr{C}\rvert}$ - Relationship between y and 3 previous class labels
2	$\phi_{p2} \in R^{\lvert\mathscr{C}\rvert}$ - Time-weighted label count
3	$\phi_{p3} \in R^{d'}$, $d' = {}_{\lvert\mathscr{C}\rvert}P_2$ - Number of changes in activity labels
4	$\phi_{p4} \in R^{\lvert\mathscr{C}\rvert}$ - Last activity transition time relative to $t_i - \delta t$

We also consider the time that the last activity transition is observed relative to $t_i - \delta t$ as a feature ($\phi_{p4} \in R^{\lvert\mathscr{C}\rvert}$). Based on the transition, i.e. *in-bed to out-bed* or *out-bed to in-bed*, the transition time is assigned to the corresponding position of the output vector from ϕ_{p4}.

6.6 Experiments

The overarching goal of the bed-egress motion analysis framework is to recognise the most number of bed-egress movements with fewer false recognitions as much as possible to avoid alarm fatigue. It is important to exercise immediate interventions while eliminating false bed-exit recognitions; therefore, we also evaluate the bed-exit movement recognition in terms of latency.

We conducted experiments using the *HOA* dataset collected from 14 older volunteers aged between 66 and 88 years as described in Section 2.4.2. This dataset contains data from two RFID configurations which are referred to as *HOA-Room1* and *HOA-Room2*. During the data collection, participants wore the sensor at sternum level over the garment and performed activities listed in two activity scripts. These activity scripts contained activities such as lying on the bed, sitting on the bed and getting out of the bed and walking to the chair. The researcher present during the data collection recorded 5 ground truth labels: i) sitting-on-bed; ii) lying-on-bed; iii) standing; iv) walking; and v) sitting-on-chair. In this study, the class labels sitting-on-bed and lying-on-bed are considered as *in-bed* and the other class labels were considered to be *out-bed*. Data from *HOA-Room1* configuration contained 45922 of *in-bed* and 6335 *out-bed* samples. In the case of *HOA-Room2* these values were 21782 and 864 respectively.

In order to evaluate the bed-exit recognition performance, we defined a true positive (TP) as a bed-exit movement recognised 5 s prior to when the participant was known to be *out-bed* or when the participant was known to be *out-bed* (i.e. they are already out of bed). All other bed-exit movements while the participant was *in-bed* were incorrect and hence counted as false positives (FPs). The missed bed-exit movements are treated as false negatives (FNs). We evaluated bed-exits after a δt duration from the start of a trial to provide sufficient information to our inference algorithm. We present precision (positive predictive value) and recall (true positive rate) of bed-exit movements along with the values of TPs, FPs and the actual number of bed-exit movements [135]. In order to evaluate the latency, we considered the time taken from the actual bed-exit movement (i.e. based on the ground truth) to the predicted bed-exit movement as the delay. Delays for bed-exit movements that are predicted prior to the actual bed-exit movement are taken as zero.

Parameter selection for the sequence learning classifier was performed using G-mean, which is the geometric mean of recall (sensitivity) and specificity (true negative rate) that are obtained using the standard confusion matrix for binary classification [136]. Since bed-exit movements are determined as changes from *in-bed* to *out-bed*, optimising the sequence learning classifier on sensitivity increases the bed-exit recognition performance.

We use leave-one-patient-out cross-validation to evaluate the performance of our framework. Denoted a set of data collected from the i^{th} participant as D_i, the whole dataset is defined as $D = \cup_{i=1}^{n} D_i$, where n is the number of participants in the dataset. Given a participant, p, D is partitioned into three subsets: i) $D_p \in D$—test set; ii) $D_j \in D \setminus D_p$—validation set; and iii) $D \setminus (D_p \cup D_j)$—training set, where j is a randomly selected participant. First, classifiers are trained with different model parameter permutations using the training sets and obtain the G-mean by evaluating the trained models using the corresponding validation sets for all participants. Then the set of model parameters that maximise the mean G-mean measure is selected to train the classifiers to obtain the testing performance. Finally, we present the results obtained using the test set D_p, using a classifier trained using $D \setminus D_p$ based on the selected set of model parameters. This evaluation scheme is close to practical usage of the framework because initially training is carried out using data collected by a subset of the target population and then the trained framework will be used to recognise the bed-exit movements of the other participants who are unknown to the framework.

6.7 Results

6.7.1 Sequence Prediction

In this section, we present the results for predicting *in-bed* and *out-bed* labels using the proposed sequence prediction algorithm. Table 6.4 lists the label prediction performance for the proposed approach and the previously published approach, Bed Exit Alert System (BEAS) [25]. Since we need real-time predictions, for BEAS, we

Table 6.4 *In-bed* and *Out-bed* label prediction performances

	HOA-Room1		*HOA-Room2*	
	Proposed	Baseline	Proposed	Baseline
Accuracy	94.0±3.5	92.4±3.8	96.1±2.4	93.4±5.2
Sensitivity	90.6±8.1	84.5±7.9	96.3±6.3	74.2±7.7
Specificity	94.7±4.5	93.8±3.5	96.2±3.0	94.8±3.9
G-mean	92.5±4.1	89.0±4.9	96.2±3.3	83.7±3.4

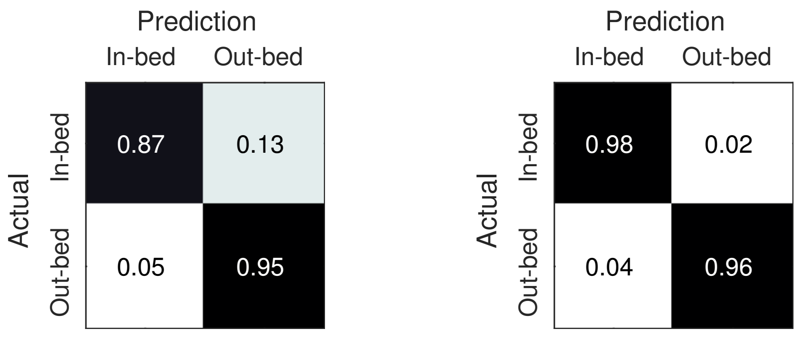


Fig. 6.1 Normalised confusion matrices for the proposed sequence learning algorithm

used the linear chain CRF algorithm that supports real-time inferencing proposed in [46] as opposed to the batch-inferencing version used previously in [25]. The BEAS uses three class labels, sitting-on-bed, lying-on-bed and out-bed. We compare here the performance of predicting the out-bed class label of the baseline approach with the proposed approach. From the results in Table 6.4, we can see that the proposed method outperforms the baseline method in all measures for both data sets. More importantly, the proposed sequential learning method showed statistically significantly higher ($p < 0.05$) performance in terms of sensitivity and G-mean measures for *HOA-Room2*. Figure 6.1 illustrates the normalised confusion matrices for predicting class labels for two datasets. From this, we can clearly observe that the sequence prediction algorithm predicted *out-bed* for both datasets with a higher accuracy.

6.7.2 Bed-Exit Movement Recognition

Table 6.5 shows the bed-exit movement recognition results for TPs, FPs, precision and recall for *HOA-Room1* and *HOA-Room2* using the proposed framework and the previously published approach, Bed Exit Alert System (BEAS) [25] for comparison. From these results (Table 6.5), we can observe that the proposed framework exhibits a significantly lower number of FPs compared with BEAS but it depicts a higher

Table 6.5 Bed-exit recognition performances (alarming performances)

Participant	Gender	Actual	Proposed Framework				BEAS proposed in [25]			
			TP	FP	Precision	Recall	TP	FP	Precision	Recall
HOA-Room1 (T : 1000, k : 100, λ : 10^{-8}, c : 0.7)*										
1	M	11	5	0	100.0	45.5	5	39	11.4	45.5
2	F	7	4	2	66.7	57.1	7	14	33.3	100.0
3	F	9	4	1	80.0	44.4	4	6	40.0	44.4
4	F	4	4	0	100.0	100.0	4	2	66.7	100.0
5	F	6	6	0	100.0	100.0	5	7	41.7	83.3
6	F	14	6	1	85.7	42.9	6	78	7.1	42.9
7	M	10	5	1	83.3	50.0	8	48	14.3	80.0
8	M	7	1	1	50.0	14.3	3	116	2.5	42.9
9	F	14	5	1	83.3	35.7	9	14	39.1	64.3
Total		82	40	7	**85.1**	48.8	51	324	13.6	**62.2**
HOA-Room2 (T : 100, k : 100, λ : 10^{-2}, c : 5.2) *										
10	F	11	11	0	100.0	100.0	10	22	31.3	90.9
11	F	13	11	1	91.7	84.6	9	16	36.0	69.2
12	M	8	8	2	80.0	100.0	8	20	28.6	100.0
13	F	10	8	2	80.0	80.0	10	92	9.8	100.0
14	F	10	7	3	70.0	70.0	10	20	33.3	100.0
Total		52	45	8	**84.9**	86.5	47	170	21.7	**90.4**

* Model parameters

number of TPs. We conclude that the proposed framework outperforms BEAS [25], as there is a lower number of FPs and consequently fewer false alarms.

When two room configurations are considered, we can see a similar performance in terms of precision. However, *HOA-Room1* depicts considerably lower recall compared with *HOA-Room2*. Based on these observations, we can see that the *HOA-Room2* configuration performs better across the two room settings.

We specifically investigated FPs and misses (i.e. number of missed bed-exit events or false negatives, $Actual - TP$) in both room configurations. In the case of *HOA-Room1*, there are 7 FPs and 42 misses. We identified that FPs are caused by sensor observations received from antenna A2 oriented towards the chair (see Figure 2.7b). These readings are from very weak signals—indicated by lower RSSI—and more likely to be due to reflections. These FPs can be addressed by reducing the RFID reader's receiver sensitivity on the antenna ports; in this case reducing the receiver sensitivity of A2's port. It is observed that all the misses are caused by insufficient sensor observations after a bed-exit. Out of 42, 24 misses have occurred in user trials that ended with a bed-exit and antenna A1, facing a standing person, failed to energise the sensor tag and capture readings. Therefore, we further evaluated the *HOA-Room1* bed-exit performance excluding the sensor data for the final 4 s for each trial. We

Table 6.6 Bed-exit recognition performance excluding final 4 s from each trial for *HOA-Room1*

Participant	Actual	Proposed Framework				BEAS proposed in [25]			
		TP	FP	Precision	Recall	TP	FP	Precision	Recall
1	6	5	0	100.0	83.3	4	39	9.3	66.7
2	4	3	2	60.0	75.0	4	14	22.2	100.0
3	4	1	1	50.0	25.0	3	6	33.3	75.0
4	3	3	0	100.0	100.0	3	2	60.0	100.0
5	5	5	0	100.0	100.0	4	7	36.4	80.0
6	9	6	1	85.7	66.7	6	78	7.1	66.7
7	6	5	1	83.3	83.3	6	48	11.1	100.0
8	5	1	1	50.0	20.0	3	116	2.5	60.0
9	10	5	1	83.3	50.0	6	14	30.0	60.0
Total	52	34	7	82.9	65.4	39	324	10.7	75.0

believe this would better reflect real-world use of the proposed method where a person ambulating is likely to be recorded and classified, albeit with a delay; these results are shown in Table 6.6. From these results we can see that the number of missed bed-exits for *HOA-Room1* has reduced by 24, thereby increasing recall from 48.8% to 65.4%. Remaining misses can be mitigated by improving the sensor tag design, in particular, by improving power harvesting by improving the antenna design, as indicated in [24] and using a hybrid-powered WISP, which is assisted by excess harvested power stored in a supercapacitor to prevent brownouts and improve reliability [130].

In contrast with *HOA-Room1* deployment, which is designed to irradiate the entire room, *HOA-Room2* deployment is designed to illuminate the areas where a patient is more likely to spend time, such as around the bed and the chair. In *HOA-Room2* deployment, there are only 8 FPs and 7 misses. All the FPs have resulted when data is not available for $> 2s$ period. In such an instance, the classifier incorrectly predicts an *out-bed* events. When more sensor observations are available, the correct class label was predicted. The misses are also caused by unavailability of sensor observations, particularly after a bed-exit.

6.7.3 Bed-Exit Movement Recognition Delays

Figure 6.2 presents the distribution of bed-exit recognition delays for the proposed framework, and the BEAS proposed in [25]. The vertical lines in Figure 6.2 indicate the 90th percentile in respective distributions.

From Figure 6.2 we can see that for the proposed framework >90% of the alarm instances are within the first 8 s for *HOA-Room1* and within the first 6 s for *HOA-Room2*. Generally, *HOA-Room2* performs better than *HOA-Room1*, as alarm delays

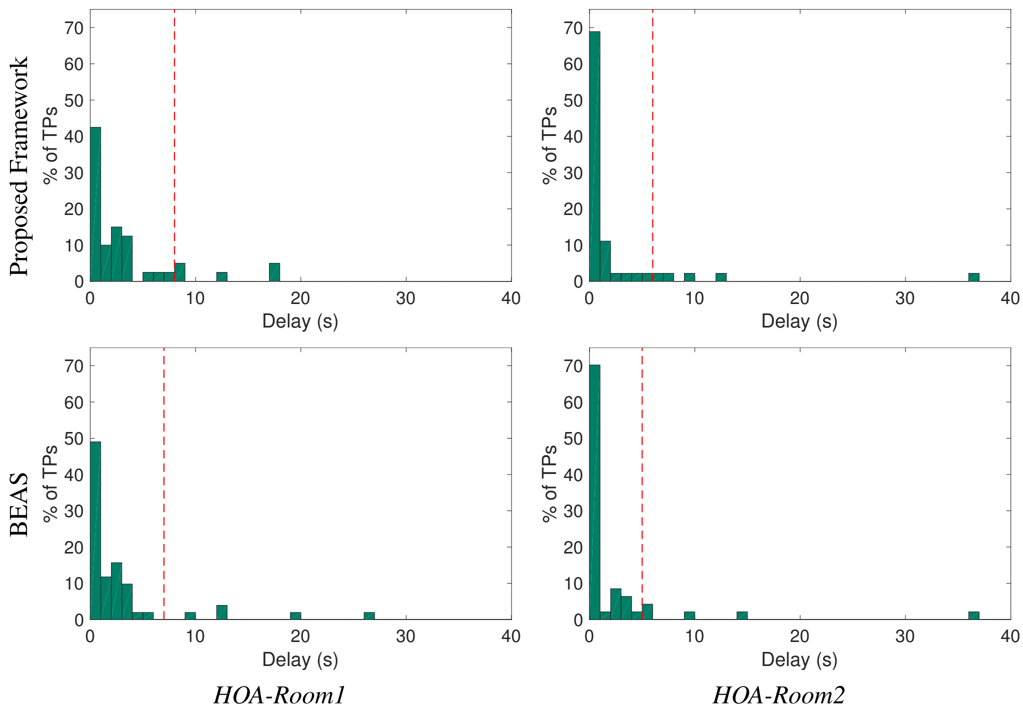


Fig. 6.2 Distribution of bed-exit movement recognition delays. Vertical lines indicate the 90th percentile of distributions.

are concentrated near zero, where there are virtually no delays. The main reason for this is the rich set of location information available from the RFID infrastructure in *HOA-Room2* due to its configuration to monitor targeted areas.

Although there are higher FPs and lower TPs for BEAS, we can see that BEAS has performed slightly better than the proposed framework in terms of delays. It is observed that for BEAS, $>90\%$ of alarm instances are within the first 7 s for *HOA-Room1* compared with 8 s and the first 5 s for *HOA-Room2*, and compared with 6 s for our proposed framework.

6.8 Discussion

In this chapter, we have devised a novel sequence classification algorithm that is suitable for sparse data streams. Based on the sequence classification algorithm, we have proposed a framework to analyse bed-exit movements using a sparse acceleration and RFID data stream from a sensor tag, i.e. W^{2}ISP. Our framework not only relies on information from a triaxial accelerometer but also considers information from the RFID infrastructure to recognise bed-exit movements in real time successfully while depicting a considerable performance improvement to that of the previously published method BEAS in [25].

Table 6.7 Performance of previous bed-egress movement alarms

Bed-egress recognition approach	Precision	Recall	Participants' age (years)
Hilbe et al. [14]		96%	18-60
Bruyneel et al. [15]	100%	91%	37 ± 9 and 45 ± 11
Najafi et al. [62]*		93	66 ± 14
Godfrey et al. [69]*		83	77.2 ± 4.3
Proposed framework	85%	86%	66-86

* a sit-to-stand posture transition was considered as a bed-egress.

Recently, several bed-egress movement sensor alarms have been reported in the literature. Most of these studies [14, 15, 62, 69] have been evaluated with healthy adult participants as opposed to older people. Hilbe et al. [14] and Bruyneel et al. [15] used pressure sensors attached to hospital beds. Najafi et al. [62] and Godfrey et al. [69] used wearable sensors strapped to participants for activity recognition and evaluated their approaches on data collected from older people, as in this chapter. Although movement sensor alarms were not the focus of these studies, they presented results for detecting sit-to-stand posture transitions that can be considered similar to a bed-egress movement. Table 6.7 summaries the results of previous bed-egress movement recognition approaches with the results obtained for *HOA-Room2*. Although a direct comparison with other studies cannot be made due to the differences in data sets and experimental settings, we can clearly see that the proposed framework performs comparably. We have not considered specificity reported in previous studies since the activities considered as true negative could not be clearly established across the studies. Furthermore, our focus has been to evaluate precision and recall as motivated in Section 6.6.

There are two major advantages arising from our approach. First, our proposed framework for bed-exit recognition is highly responsive and reduces time delays to alarm so that caregivers are given a better opportunity to intervene while patients are undertaking unsupervised risky activities. In contrast, previous methods needed pre-processing of sensor data (such as filtering) [61] or a waiting time for sensors to reach a steady state (≈ 2 min in [15]) prior to generating an alarm. Furthermore, approaches to handling activity label prediction inaccuracies in [62] and [18] rely on initiating notifications (i.e. alarms) after detecting a change in predictions (i.e. from *sitting-on-bed* to *standing*) and a subsequent *fixed evaluation* period (i.e. 10 s) to assess the consistency of the predicted activity label for subsequent sensor observations; this approach will incur a notification delay equal to the fixed evaluation period. Instead, we notify (i.e. alarm) at the *first instance of detecting a change in predictions* (e.g.

from *sitting-on-bed* to *standing*). Our method requires no pre-processing as the novel sequence prediction algorithm accurately predicts bed-exit motion sequences using time domain features. Secondly, although the proposed bed-exit sequence prediction algorithm was used to predict bed-exit motion sequences, it can also be used in other sequence prediction problems where the data stream is sparse, as in our case. However, different features may be required, depending on the classification problem and data available from the data stream.

In summary, in this chapter we have looked at sequence learning from sparse data streams to recognise activities in real time and highlight that better data analysis methods can improve the activity recognition performance by alleviating the effects of noise and the sparse nature of data streams from sensor tags. The improved performance in activity monitoring can be seen as significant in facilitating the deployment of passive sensor enabled RFID tags in real-world applications.

Chapter 7

Segmentation-Free Activity Classification Using a Structured Predictor

7.1 Introduction

In the previous chapters, we have utilised classical machine learning algorithms to predict activity labels for sensor observations from the passive sensor enabled RFID tag (sensor tag). In Chapter 6, we have looked at a sequence learning algorithm to predict the activity labels by considering the sparse nature of the data stream from the sensor tag.

The machine learning method proposed in Chapter 6, as well as most of the previous activity recognition methods listed in Chapter 2, require the data stream to be segmented, typically using fixed size segmentation methods, such as fixed-time or fixed-sample, prior to extracting features to be used with machine learning classification models [17, 63, 46]. Furthermore, application of sequence learning methods, such as HMM still require segmenting the data stream prior to calculating features [83, 75]. As highlighted in Chapter 5, such data stream segmentation methods result in sensor observations related to previous activities to be considered in the data stream segment pertaining to the current activity. This results in calculation of features that are not representative of the activities which can degrade the activity recognition performance. The sparse and variable sampling rate of the sensor data stream from sensor tag aggravates this issue because for a sensor data stream segment, majority of the sensor observations can be from the previous activity despite they represent a smaller duration within the segment.

In Chapter 5, we have proposed data stream segmentation methods to segment the data stream on natural activity boundaries, and this is evaluated using the dataset collection from healthy young participants (*HYA*). The proposed data stream segmentation methods were based on heuristics of the sensor wearers' trunk rotational motion between consecutive sensor observations and alleviates issues discussed above related to fixed size segmentation methods. Nevertheless, explicitly recognised segments using the natural boundaries needs to be presented for a machine learning classifier to recognise the respective activities. Data collected from older participants depicts a higher sparseness, i.e. they have larger inter-sensor observation time differences as shown in Table 2.2. Consequently, the method proposed in Chapter 5 cannot be successfully applied to data streams with higher sparsity due to its reliance on the change in the values of consecutive sensor observations.

In this chapter, we propose a real-time sensor data stream classification method that does not require any explicit segmentation prior to classification, hence it overcomes the issues related to explicit segmentation posed by previous HAR approaches. In particular, we propose a structured predictor to predict a structure for a data stream subsequence, thereby utilising the structure to assign class labels to the sensor observation stream. The predicted structure on a given subsequence of sensor observations consists of two parts (see Section 7.2). Given a sensor data subsequence, the two-part structure includes the location of the boundary of the parts and the activity class label for Part 1. Taking advantage of the structured output, we propose six ways to measure dissimilarity between output structures, which are utilised during the training phase of the proposed structured predictor. Finally, we show the performance of the proposed structured predictor based data stream classification approach by conducting experiments with three datasets, described in Section 7.6.

7.2 Representation and Inferencing

We denote the i^{th} sensor data at time t_i, $t_i > t_{i-1}$, $i \in \mathbb{N}$, using the pair (t_i, s_i), then the sequence of collected data $\mathbf{X} = \{(t_i, s_i)\}_{i \geq 1}$, $i \in \mathbb{N}$ is a time series. We use the notation $\mathbf{X}_{[a,b]}$ to denote the sensor data stream segment for the time interval $[a,b]$, where $b > a$.

Our goal is to assign an activity label or a class label for each and every sensor observation. Given a subsequence of sensor observations, this task can be modelled as a structured prediction problem. As opposed to regular prediction problems which have real-value output in case of regression and categorical outputs in case of classification, the output of a structured prediction problem contains some form of a complex structure, such as trees, graphs, sequences and other structures.

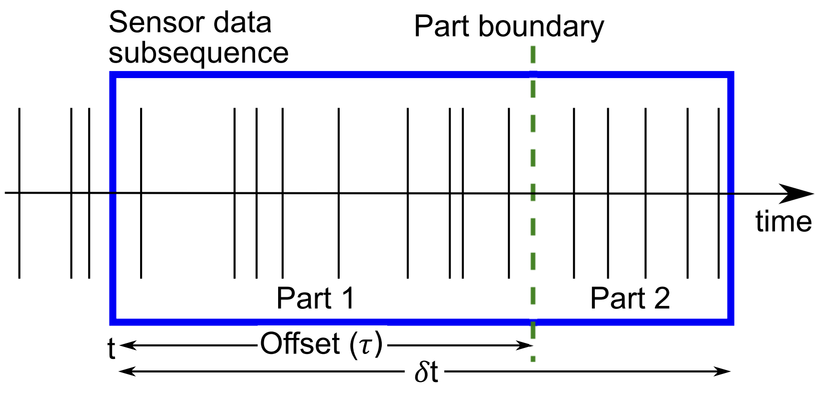


Fig. 7.1 The two-part structure is defined on a given sensor data subsequence. Part 1 is assumed only to have sensor observations from a single activity and Part 2 can contain sensor observations from all the activities. This part can be represented by the pair (τ, c), where τ is the time offset to the part boundary and c is the class label associated with Part 1.

Given a subsequence of sensor observations $\mathbf{X}_{[t,t+\delta t]}$ of time length δt, we assume that $\mathbf{X}_{[t,t+\delta t]}$ has a two-part structure as illustrated in Figure 7.1. We denote the time offset from t to the boundary of the parts as τ and consider that Part 1, i.e. $X_{[t,t+\tau]}$, which has sensor observations towards the beginning of the subsequence, belongs to a single activity. Part 2, i.e. $\mathbf{X}_{[t+\tau,t+\delta t]}$, contains the remaining sensor observations, and we assume that they may belong to different activities or the same activity, as in Part 1. We denote this two-part structure using the pair $Y = (c, \tau)$, where c is the class label associated with Part 1.

In the context of real-time predictions, the proposed approach proceeds as follows: sensor observations are buffered as and when they become available. When the contents of the buffer reach a pre-defined time length, δt, then the output structure, Y, for the buffer content is predicted. The output, Y, is represented by the pair (τ, c) where τ is the boundary offset and c is the activity label for Part 1. Then, the sensor observations pertaining to Part 1 are assigned the class label c and Part 1 is removed from the buffer while retaining sensor observations relating to Part 2. This process repeats until data the stream continues. The proposed structured predictor can determine the output structure that includes both the part boundary offset and the activity label of Part 1, and therefore, the requirement for explicit segmentation prior to classification is eliminated.

Based on the above definitions, our structured prediction problem is to predict the output structure Y for the presented input sequence of sensor observations $\mathbf{X}_{[t,t+\delta t]}$ of length δt. This task can be cast into learning a score function $f : \mathbf{X} \times Y \mapsto \mathbb{R}$, such that given a subsequence, X_j, the two-part structure Y_j can be obtained using:

$$Y_j = \underset{Y \in \mathscr{Y}_j}{\arg\max}\, f(\mathbf{X}_j, Y) \tag{7.1}$$

where $\mathscr{Y}_j$ is the possible set of structures for $\mathbf{X}_j$. Essentially, Eq.(7.1) represents the classification function and it is important to note that, as in many structured prediction problems, we assume the information in a given subsequence $\mathbf{X}_j$ is sufficient to predict its structure Y_j.

For f, we consider functions that are linear in some feature representation ϕ [133, 137]. Such a linear function allows representing the classification model using a vector. Therefore, as mentioned in Section 2.2.4, inferencing using such a linear function is efficient as the score for prediction can be obtained efficiently using the inner-product between the classification model ($\mathbf{w}$) and the feature representation ϕ (see Eq.(7.2)).

$$f(\mathbf{X}, Y; \mathbf{w}) = \langle \mathbf{w}, \phi(\mathbf{X}, Y) \rangle \tag{7.2}$$

In Equation Eq.(7.2),classification model $\mathbf{w}$ is learnt following the training procedure described in Section 7.3. Similar to the method described in Chapter 6, the feature representation captures the relationships between the input $\mathbf{X}_i$ and an output structure Y into a vector. This has been explained in detail in Section 7.4.

In Equation Eq.(7.1), generating the set $\mathscr{Y}_j$ is important to predict a suitable structure, Y_i, accurately. A given subsequence, $\mathbf{X}_j$, can be partitioned in a number of ways to obtain a comprehensive $\mathscr{Y}_j$. However, this approach will hinder the runtime performance as the number of evaluations are linear to the $|\mathscr{Y}_j|$, where $|\cdot|$ denote the cardinality of a set. Given $\mathbf{X}_j$ has a time length of δt, we create a set of time offsets $\mathscr{T}_j$ for partitioning, as in Eq.(7.3),

$$\mathscr{T}_j = \left\{ \tau_j^k : \tau_j^k = t_{shift} \times k,\ \tau_j^k <= \delta t \right\} \tag{7.3}$$

where $k \in \mathbb{N}^+$ and t_{shift} is the time difference between two partitioning positions. As can be seen from Eq.(7.3), the number of partitions can be controlled using t_{shift}. Additionally, the output structure also contains the class label $c_j \in \mathscr{C}$ of the first part, where $\mathscr{C}$ represents the set of class labels considered in this study. During inferencing, we consider all possible pairs of class labels, $\mathscr{C}$ and partition offsets $\mathscr{T}_j$.

7.3 Training

The goal of the classifier training is to find the model w that minimises the classification error. We take a similar approach to that has been taken in Chapter 6 to train the classifier. According to the prediction function Eq.(7.1), we can observe that the score for a given sensor observation with the most similar structure, $f(\mathbf{X}_j, Y_j)$ must be greater than the score of all other structures $f(\mathbf{X}_j, Y), Y \in \mathscr{Y}_j \backslash Y_j$, i.e. $f(\mathbf{X}_j, Y_j) > f(\mathbf{X}_j, Y)$.

Using the generic framework for structured prediction proposed in [134], we can learn the function f, particularly, the classification model $\mathbf{w}$. As also explained in Section 6.3.2, in the SVM training setting, $f(\mathbf{X}_j, Y_j)$ should be at least greater than a margin, which is typically set to 1 [134]. Consequently, the constraint $f(\mathbf{X}_j, Y_j) \geq f(\mathbf{X}_j, Y) + 1$ is used for minimising the classifier training error.

As we focused on predicting the structure of a given subsequence, we created the training dataset from data streams by first segmenting the data streams using a δt sized sliding window with t_{shift} slide length and then assigning the structure based on the ground truth. Here δt is the length of the buffer. Given a training dataset containing subsequences $\mathbf{X}_j$ and the corresponding structure Y_j, $D = (\mathbf{X}_j, Y_j)_{j=1}^{M}$ where M is the length of the dataset, as described in [134], we learn the classification model $\mathbf{w}$ by solving the primal form of the constrained convex optimisation problem in Eq.(7.4).

$$
\begin{aligned}
\min_{\mathbf{w}, \xi_j} \quad & \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{M}\sum_{j=1}^{M} \xi_j \\
\text{subject to} \quad & \forall j \; \langle \mathbf{w}, \phi(\mathbf{X}_j, Y_j)\rangle - \langle \mathbf{w}, \phi(\mathbf{X}_j, Y_r)\rangle \geq 1 - \frac{\xi_j}{\Delta(Y_j, Y_r)} \\
& \forall j \; \xi_j \geq 0 \\
\text{where} \quad & Y_r = \underset{Y \in \mathscr{Y}_j \backslash Y_j}{\arg\max} \langle \mathbf{w}, \phi(\mathbf{X}_j, Y)\rangle
\end{aligned}
\tag{7.4}
$$

In Equation Eq.(7.4), λ represents the regularisation parameter, $\Delta(Y_j, Y_r)$ is the cost function and ξ_js are positive slack variables, which are introduced to capture margin violations during training. We follow the same procedure to generate the set $\mathscr{Y}_j$, as described for inferencing in Section 7.2. The cost function, $\Delta(Y_j, Y_r)$ essentially measures the dissimilarity between structures Y_j and Y_r, and we specifically used the slack rescaling approach described in [134] as this behaves better when compared with the other alternative, margin rescaling. In this study, different ways of measuring dissimilarity are considered, and they are described in Section 7.5.

The optimisation problem Eq.(7.4) can be converted to an unconstrained optimisation problem as in Eq.(7.5) and can be solved using Stochastic Gradient Descent

(SGD) approaches.

$$\min_{\mathbf{w}} \quad \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{M}\sum_{j=1}^{M} max\left(\Delta(Y_j, Y_R)(1 + \langle \mathbf{w}, \phi(\mathbf{X}_j, Y_r) - \phi(\mathbf{X}_j, Y_j)\rangle), 0\right) \tag{7.5}$$

We use the Pegasos algorithm [90] to solve the unconstrained optimisation problem in Eq.(7.5) directly using its primal form. We specifically consider the mini-batch version of Pegasos, given in Eq.(7.6), to reduce the training time as we have a large number of training samples.

$$\min_{\mathbf{w}} \quad \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{k}\sum_{j=1}^{k} max\left(\Delta(Y_j, Y_R)(1 + \langle \mathbf{w}, \phi(X_j, Y_r) - \phi(X_j, Y_j)\rangle), 0\right) \tag{7.6}$$

In Equation Eq.(7.6), k is the mini-batch size and the Pegosos algorithm iterated T times and drawing k number of training data points uniformly with replacement from the training dataset. We use the learning rate $\eta = 1/(T\lambda)$ to attain the same convergence rate specified in [90].

Unlike other methods, such as LaRank [133] which optimise SVMs in dual form, optimising in primal form is attractive where there are a large number of training samples, as in our case.

7.4 Feature Representation

Based on the output structure that consists of two parts shown in Figure 7.1, we extract features considering: i) subsequence sensor observations; and ii) location of the boundary between Part 1 and Part 2 (i.e. the boundary offset). These features are then correlated with the class associated with Part 1 of the output structure.

7.4.1 Features from Sensor Subsequences

Based on the output structure, we can consider three subsequence; i) sensor observations for the entire subsequences—$\mathbf{X}_{[t,t+\delta t]}$; ii) sensor observations for Part 1—$\mathbf{X}_{[t,t+\tau]}$; and iii) sensor observations for Part 2—$\mathbf{X}_{[t+\tau,t+\delta t]}$. For each subsequence, we extract statistical features using the acceleration signal and RSSI information available from the sensor tag employed in this study.

Features from Acceleration

Acceleration signals are obtained relative to the reference frame of the sensor, and we considered the acceleration on the frontal axis (a_f), lateral axis (x_l) and the vertical

Table 7.1 Features from acceleration signals

	Feature
1	$mean(x)$
2	$mode(x)$
3	$median(x)$
4	$range(x)$
5	$var(x)$—variance
6	$std(x)$—standard deviation
7	$mean(\|(x - mean(x))\|)$—mean absolute deviation
8	$median(\|(x - median(x))\|)$—median absolute deviation
9	$max(x)$
10	$min(x)$
11	$MPM(x)$—Maximum Prior to Minimum

x represents the sequence of values for each variable.

Table 7.2 Features from the RFID platform

	Feature
1	$mean(RSSI_a)$ [87]
2	$std(RSSI_a)$
3	$max(RSSI_a)$ [87]
4	$min(RSSI_a)$ [87]
5	$MPM(RSSI_a)$

$RSSI_a$ represents the RSSI values for sensor observations captured by antenna a

axis (a_v). From the acceleration signals, as described in Chapter 4, rotational angles on the sensor wearer's reference frame on the sagittal (θ), coronal (α) and transverse (β) planes are approximated. Consequently, we consider 11 statistical features listed in Table 7.1, considering the aforementioned six variables. Thus we have extracted 66 features from a sequence of acceleration signals.

Features from RSSI

Statistical features listed in Table 7.2 are extracted using RSSI available from the RFID platform (Section 2.3.2). As described in Chapter 4, given the set of antennas in the RFID platform as $\mathscr{A}$, features from RSSI are extracted considering sensor observations captured by each antenna $RSSI_a$ where $a \in \mathscr{A}$.

7.4.2 Features based on the Part Boundary

We use the location of the boundary between Part 1 and Part 2 as a feature. Specifically, we use an indicator vector to represent the location. As mentioned in Section 7.2, the boundary $\tau \in \mathscr{T}$ is defined by $\mathscr{T} = \left\{\tau^k : \tau^k = t_{shift} \times k,\ \tau^k <= \delta t\right\}$. A zero vector of length $\delta t / t_{shift}$, i.e. $v = \{0\}^{\delta t / t_{shift}}$, is created, and the k^{th} element of the vector is assigned a value of 1, considering the location of the boundary, i.e. $v_k = 1$.

7.5 Measuring Dissimilarity

As mentioned earlier, the output structure is represented by the pair (τ, c). In this structured prediction problem, we measure the dissimilarity between the correct output, Y_j and another Y_l based on two aspects, the class label associated with Part 1, c and the location of the part boundary, τ. Specifically, we consider a cost function in the form presented in Algorithm 4. In line 2 of Algorithm 4, $M\mathscr{M}(a,b)$ is a square matrix $\mathscr{M} \in \mathbb{R}^{|\mathscr{C}| \times |\mathscr{C}|}$ containing costs for misclassification of the class a as the class b. The function $almt(a,b)$ (line 4 in Algorithm 4) measures the alignment of the part boundary. As evident from this algorithm, we first evaluate the class labels for Part 1, and if the class labels are equal, then the boundary alignment is considered. If the sensor observations in Part 1 of the considered structures belong to different classes, then a comparison on the part boundary cannot be considered as different activities usually have different lengths.

Algorithm 4 Cost function

```
Require: $Y_j$ and $Y_l$
1: if $Y_j.c \neq Y_l.c$ then
2:     return $\mathscr{M}(Y_j.c, Y_l.c)$
3: else
4:     return $almt(Y_j.\tau, Y_l.\tau)$
5: end if
```

7.5.1 Cost for Misclassification of Part 1

We have considered three approaches to obtain the matrix $\mathscr{M}$ that defines the cost for assigning the class label b instead of the correct class label a for part 1.

As the first approach, we assign equal costs for misclassification by using a matrix of ones as $\mathscr{M}$; thus $\mathscr{M}^1 = \{1\}^{|\mathscr{C}| \times |\mathscr{C}|}$ We denote this first approach as $\mathscr{M}^1$.

The second approach, referred to as $\mathscr{M}^2$, considers different penalisations for addressing imbalance [125]. The cost for misclassifying a is obtained by considering the ratio of classes in the training dataset, and the same cost is assigned irrespective of the other class. Denoting m as the number of training samples and m_a as the number of training samples for class a, we obtain $\mathscr{M}^2$ as in Eq.(7.7).

$$\begin{aligned}\mathscr{M}^{2'}(a,\cdot) &= m/m_a \\ \mathscr{M}^2(a,\cdot) &= \mathscr{M}^{2'}(a,\cdot)/min(\mathscr{M}^{2'}(a,\cdot))\end{aligned} \tag{7.7}$$

The third approach is inspired by the nature of the optimisation problem Eq.(7.4) for training the classifier. In the objective function Eq.(7.4), the constraint only considers the correct output structure Y_j and the output structure, Y_r, resulting in the highest score except Y_j to obtain the loss. Therefore, in order to define the cost of misclassifying class a as class b, we use the ratios between the two classes in the training dataset. We denote this approach as $\mathscr{M}^3$ and it is obtained as given in Eq.(7.8).

$$\mathscr{M}^3(a,b) = \frac{m_b}{m_a \times min(\frac{m_a}{m_b}, \frac{m_b}{m_a})} \tag{7.8}$$

7.5.2 Cost Based on the Part Boundary

We have considered two forms of the function $almt(a,b)$ to measure the dissimilarity between alignments of the boundary offset.

The first approach, referred to as $almt^1$, uses the absolute difference between the given boundary offsets, a and b.

$$almt^1(a,b) = \frac{|a-b|}{\delta t} \tag{7.9}$$

In Equation Eq.(7.9), δt is used to scale the output value to the range $[0,1]$ so that more emphasis is given to the misclassification instead of the alignment.

In the second approach, the square of the first approach is used, and it is denoted as $almt^2(a,b)$.

$$almt^2(a,b) = \left(almt^1(a,b)\right)^2 = \left(\frac{a-b}{\delta t}\right)^2 \tag{7.10}$$

7.6 Experiment and Statistical Analysis

We have conducted experiments using the two datasets. We utilised the dataset collected from 14 older volunteers aged between 66 and 88 years, as described in Section 2.4.2. This dataset contains data from two RFID configurations, *HOA-Room1* and *HOA-Room2*. We also utilised the data collected from 24 hospitalised older volunteers, *FOA*, described in Section 2.4.3. During the data collection, participants wore the sensor at sternum level over the garment and performed activities listed in two activity scripts. These activity scripts contained activities such as lying on the bed, sitting on the bed, get out of the bed and walk to the chair. The researcher present during the data collection recorded five ground truth labels: i) sitting-on-bed; ii) lying-on-bed; iii) standing; iv) walking; and v) sitting-on-chair. For this study, standing and walking classes were combined into ambulating as there were a small number of sensor observations. Therefore, we considered four class labels: i) sitting-on-bed; ii) lying-on-bed; iii) ambulating; and iv) sitting-on-chair.

As mentioned in Section 7.5, there are six combinations of dissimilarity measurements can be obtained and we evaluated these six combinations. We compared our method with two linear multi-class classifiers, as we are interested in linear decision functions to reduce the training time with larger datasets and to reduce the runtime complexity during inferencing. We utilised linear SVM provided by LibLinear [123] and our implementation of a multi-class linear classifier similar to that presented in [90]. The main difference between these two classifiers is that LibLinear uses the one-against-one approach to achieve multi-class classification while our implementation of multi-class SVM uses a score function and obtains the class label similar to Eq.(7.1). For both these classifiers, we extracted features for a sensor subsequence described in Section 7.4.1, for a segment of δt which is sided by t_{shift}.

During the evaluation, we set $\delta t = 5\,s$ and $t_{shift} = 0.5\,s$. This is based on the general observation from the datasets that each participant took approximately 2 s to perform an activity transition such as sitting-on-bed to get-out-of-bed; thus the buffer will be able to capture adequate motion information to determine its structure. For the proposed structured predictor and the implemented multi-class baseline SVM we set the number of iterations $T = 1000$ and mini-batch size $k = 100$. The remaining parameter λ and the cost parameter (c) for LibLinear has been selected based on leave-one-participant-out cross-validation as described below.

We use the cross-validation procedure described in Section 6.6. Given a participant, p, D is partitioned into three subsets: i) $D_p \in D$—test set; ii) $D_j \in D \setminus D_p$—validation set; and iii) $D \setminus (D_p \cup D_j)$—training set, where j is a randomly selected participant. We trained the classifier using the training sets and obtained the F-score by evaluating the trained models using the corresponding validation sets to select a set of model parameters that maximise the mean F-score measure. Finally, we present the results obtained using the test set D_p, using a classifier trained using $D \setminus D_p$ based on the selected set of model parameters.

Table 7.3 Classification results for *HOA-Room1*

Classifier	Sensitivity	PPV	F-score	Parameters
Proposed				
$\mathscr{M}^1$+*almt*^{1}	85.8±4.4	87.7±8.5	83.8±10.0	$\lambda = 10^{-3.6}$
$\mathscr{M}^2$+*almt*^{1}	84.5±7.1	83.5±7.1	83.4±6.7	$\lambda = 10^{-0.8}$
$\mathscr{M}^3$+*almt*^{1}	84.9±8.9	84.0±7.9	83.8±8.1	$\lambda = 10^{-1.2}$
$\mathscr{M}^1$+*almt*^{2}	85.3±4.3	86.5±7.8	83.6±8.8	$\lambda = 10^{-3.2}$
$\mathscr{M}^2$+*almt*^{2}	84.9±6.9	84.1±6.2	83.8±6.0	$\lambda = 10^{-0.8}$
$\mathscr{M}^3$+*almt*^{2}	85.5±6.4	84.7±6.6	**84.5**±**6.0**	$\lambda = 10^{-1.2}$
Baseline				
LibLinear	81.2±10.0	85.0±9.7	81.7±10.5	$c = 2^{5.0}$
Milti-class SVM	80.9±8.3	85.3±9.4	81.9±8.8	$\lambda = 10^{-3.0}$

7.7 Results

The activity classification results using the aforementioned datasets from the sensor tag are presented in Table 7.3, Table 7.4 and Table 7.5. When the mean F-score is considered, the proposed real-time data stream classification approach, based on a structured predictor, depicted higher performance compared with the baseline approaches for all the three datasets *HOA-Room1*, *HOA-Room1* and *FOA*. The main reason for this performance improvement is the capability of the structured predictor to assign a class label to a part of a data stream with a higher confidence (see Section 7.2); this is in contrast with assigning a single class label for an entire sensor data segment, as commonly performed in classification approaches, such as the baseline.

When the proposed combinations of dissimilarity measures are considered (see Section 7.5), we can observe that three different combinations have depicted a higher performance for each dataset. For *HOA-Room1*, the M^3+*almt*^{2} combination depicted the highest performance while the M^1+*almt*^{1} combination showed the highest performance for *HOA-Room2*. In the case of the *FOA* dataset, the M^2+*almt*^{2} combination demonstrated the highest performance. Although the structured prediction approach depicted a good performance, the dissimilarity measures used during training affects the final classification performance. Therefore, the most appropriate dissimilarity measure needs to be selected, based on the dataset.

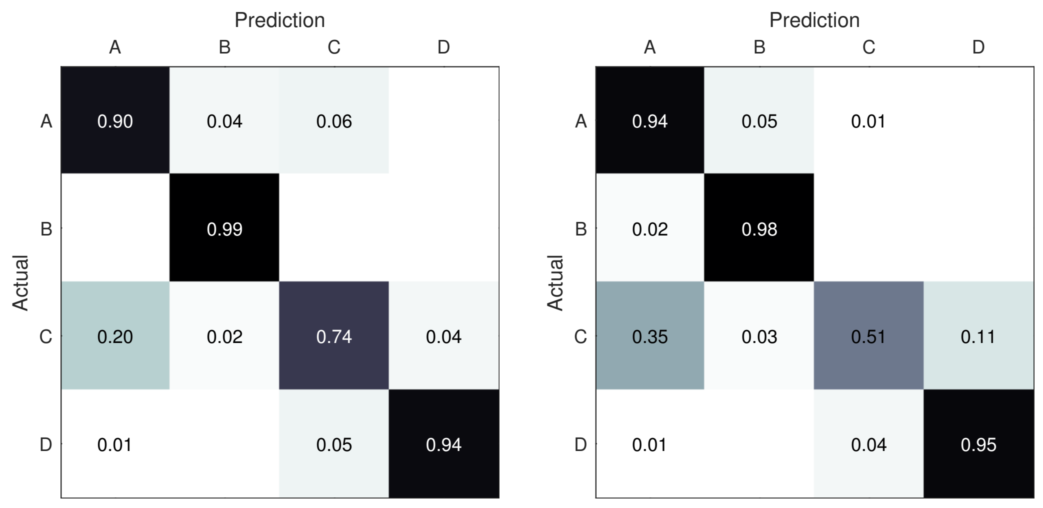
We further analysed the results based on the normalised confusion matrices for the best performing classifiers. The confusion matrices for the three datasets are shown in Figures 7.2 and 7.4. In these Figures, the class labels are represented as follows: A–sitting-on-bed; B–lying-on-bed; C–ambulating; and D–sitting-on-chair.

Table 7.4 Classification results for *HOA-Room2*

Classifier	Sensitivity	PPV	F-score	Parameters
		Proposed		
$\mathscr{M}^1$+*almt*^{1}	76.3±5.8	79.0±13.4	**75.5**±**10.7**	$\lambda = 10^{-3.2}$
$\mathscr{M}^2$+*almt*^{1}	74.6±6.6	72.1±6.2	70.6±7.0	$\lambda = 10^{-1.6}$
$\mathscr{M}^3$+*almt*^{1}	76.4±4.0	71.9±4.1	73.4±4.0	$\lambda = 10^{-2}$
$\mathscr{M}^1$+*almt*^{2}	75.9±4.3	75.4±7.7	74.6±5.4	$\lambda = 10^{-3.2}$
$\mathscr{M}^2$+*almt*^{2}	72.9±5.9	70.8±5.0	69.7±5.4	$\lambda = 10^{-0.8}$
$\mathscr{M}^3$+*almt*^{2}	74.2±11.2	70.8±11.8	71.7±11.2	$\lambda = 10^{-2.8}$
		Baseline		
LibLinear	77.2±4.9	73.1±16.6	72.4±10.5	$c = 2^2$
Milti-class SVM	78.4±2.7	75.4±15.7	74.0±9.6	$\lambda = 10^{-3}$

Table 7.5 Classification results for *FOA*

Classifier	Sensitivity	PPV	F-score	Parameters
		Proposed		
$\mathscr{M}^1$+*almt*^{1}	55.9±16.5	55.6±18.9	51.4±17.7	$\lambda = 10^{-2.6}$
$\mathscr{M}^2$+*almt*^{1}	57.4±17.1	58.5±17.0	51.7±16.2	$\lambda = 10^{-2.0}$
$\mathscr{M}^3$+*almt*^{1}	57.0±18.0	57.5±15.0	52.7±16.9	$\lambda = 10^{-2.0}$
$\mathscr{M}^1$+*almt*^{2}	56.1±13.3	58.4±16.6	52.7±15.1	$\lambda = 10^{-2.4}$
$\mathscr{M}^2$+*almt*^{2}	59.6±20.9	59.7±16.9	**56.6**±**19.2**	$\lambda = 10^{-1.8}$
$\mathscr{M}^3$+*almt*^{2}	60.1±19.8	60.7±16.8	56.2±18.3	$\lambda = 10^{-2.0}$
		Baseline		
LibLinear	57.6±13.8	62.0±16.3	54.2±15.7	$c = 2^{3.5}$
Milti-class SVM	54.7±12.5	56.2±13.2	50.0±14.0	$\lambda = 10^{-2.6}$

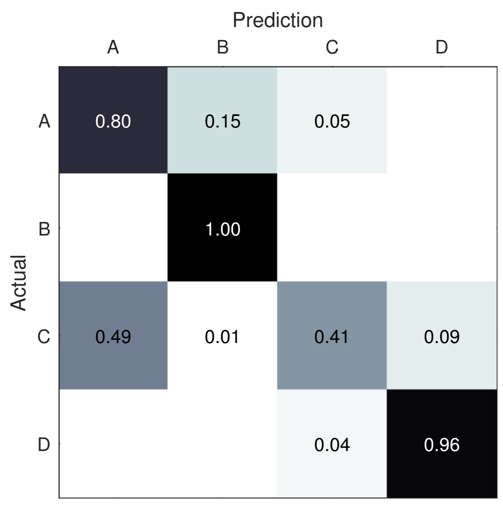


(a) $\mathscr{M}^3$+*almt*^{2} (b) Multi-class SVM

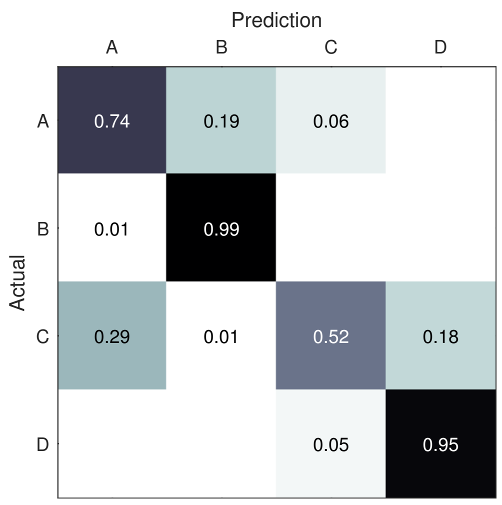
Fig. 7.2 Normalised confusion matrix for *HOA-Room1*. A: Sitting-on-bed; B: Lying-on-bed; C: Ambulating; D: Sitting-on-chair.

Figure 7.2a represents the normalised confusion matrix for the *HOA-Room1* dataset using the M^3+*almt*^{2} combination as the dissimilarity measure. Figure 7.2b is the normalised confusion matrix for Multi-class SVM, which depicted the highest performance for the *HOA-Room1* dataset among the baseline classifiers. It is interesting to note that the Multi-class SVM only classified 51% of the ambulating (C) instances correctly, while the proposed structured predictor with M^3+*almt*^{2} was able to classify 74% of them correctly. However, the proposed classifier misclassified 6% of sitting-on-bed instances as ambulating but that case for the baseline is only 1%.

Figure 7.3a represents the normalised confusion matrix for proposed classifier using the $\mathscr{M}^1$+*almt*^{1} combination as the dissimilarity measure for the *HOA-Room2* dataset. The Multi-class SVM depicted the highest performance among baseline classifiers for the *HOA-Room2* dataset, hence its confusion matrix for comparison with the proposed classifier in Figure 7.2b. From these Figures, we can see that for both classifiers, there are higher values on the main diagonal of each confusion matrix, which indicates the good classification performance. It is important to note that the proposed structured predictor had classified 49% of the ambulating (C) instances into sitting-on-bed (A). The Multi-class classifier had also misclassified 29% of ambulating (C) instances as sitting-on-bed (A). Furthermore, the Multi-class SVM had misclassified 18% of the ambulating instances as sitting-on-chair (D) while this percentage for the proposed classifier is only 9%. Apart from the classification for ambulating (C), the proposed classifier depicted higher correct classifications compared with the baseline for all the classes.



(a) $\mathscr{M}^1$+*almt*^{1}

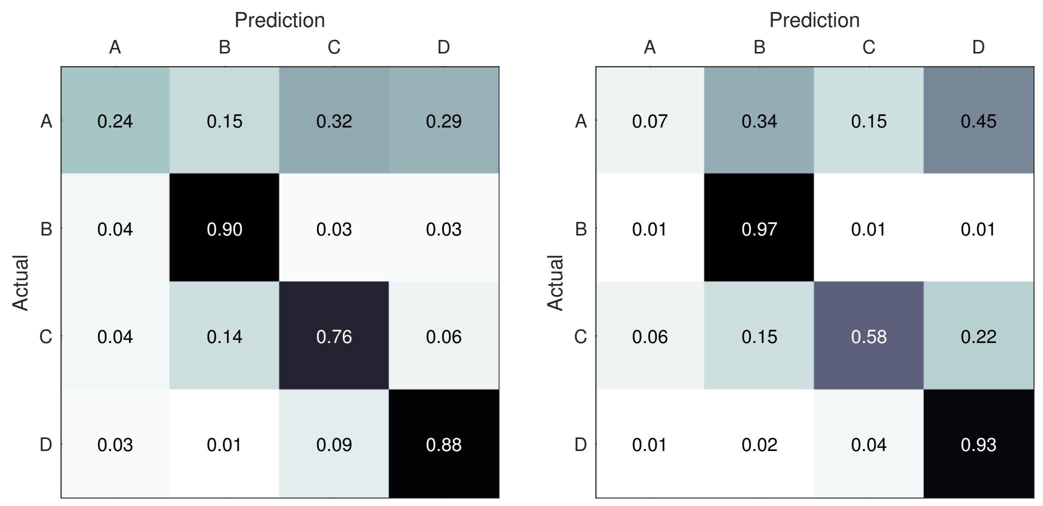


(b) Multi-class SVM

Fig. 7.3 Normalised confusion matrix for *HOA-Room2*. A: Sitting-on-bed; B: Lying-on-bed; C: Ambulating; D: Sitting-on-chair.

Confusion matrices for the *FOA* dataset using the $\mathscr{M}^2$+*almt*^{2} combination for the proposed classifier and the LibLinear classifier are illustrated in Figure 7.4a and Figure 7.4b respectively. From these Figures, we can see that both classifiers failed to classify correctly sitting-on-bed (A). The proposed structured predictor correctly classified 24% of the sitting-on-bed instances while the baseline method, LibLinear, only classifying 7% of the instances correctly. A possible reason for this confusion may be due to the shorter distance between the chair and the patient's bed. As a result, patients were also able to transfer to the chair without straightening their trunk. Furthermore, when the classification of ambulating (C) is considered, the proposed approach was able to classify 75% of the instances correctly, but LibLinear was only able to classify 58% of the ambulating (C) instances correctly.

We can observe that all the classifiers face a difficulty in classifying the ambulating (A) class. This is mainly due to two reasons. First, as discussed in Section 2.4, the sensor was attached to the garment at the sternum level over the garment. Therefore, it is difficult to distinguish ambulating or sitting-on-bed by observing the acceleration signal because, during these postures, the participants' trunks remain upright. The other compounding fact is the limited number of sensor observations for the ambulating activity in all datasets. In fact, in all the datasets, ambulation had the least number of sensor observations. Consequently, these factors attributed to the lower classification performance for ambulation.



(a) $\mathscr{M}^2$+*almt*^{2} (b) LibLinear

Fig. 7.4 Normalised confusion matrix for *FOA*. A: Sitting-on-bed; B: Lying-on-bed; C: Ambulating; D: Sitting-on-chair.

7.8 Discussion

In this chapter, we proposed a real-time classification approach, based on a structured predictor, to classify activity data streams. The structured predictor is capable of predicting a two-part structure, including the location of the boundary of the parts, as well as the activity class label for Part 1 for a given sensor data subsequence. We extracted statistical features based on acceleration signals, as well as RSSI information available from the sensor tag, considering three sensor data subsequences defined by the structure (see Section 7.4.1). Features were also extracted considering the part boundary (see Section 7.4.2). Thus, these features capture the patterns represented by the output structure in a format suitable for learning and prediction using machine learning approaches. Finally, from the experiment conducted with three data sets collected from older people, we show that the proposed structured predictor based activity stream classification method performed better when compared with the multi-class classifiers used as the baseline.

Previous, as explained in Section 2.2.3 real-time activity recognition studies had relied on explicit segmentation of the data stream to extract features [17, 48, 46, 83, 75]. Studies such as in [17, 63, 73] utilised fixed size segments to extract features for recognising activities using machine learning based classifies. The fixed size segmentation method is simple to implement suited for real-time activity recognition as it can easily be implemented using a buffer. Although Lester et al. [83] and Suutala et al. [75] utilised HMM with discriminative classifiers to model activity sequences, they relied on fixed size data stream segments to train their discriminative classifiers. Furthermore, the application of HMM, particularity the use forward-backward algorithm during inferencing, can be seen as a limitation on real-time activity recognition. In general, dynamic segmentation methods seek to segment the data stream on activity boundaries (see Chapter 5). Junker et al. [48] investigated dynamic segmentation based on the sliding-window and bottom-up algorithm, where and later used HMM to classify the activities. This segmentation method required significant number of iterations (≈ 350) over a data stream buffer to identify segments. Shinmoto Torres et al. [46] evaluated several sensor data segmentation methods, including a dynamic segmentation method for sensor data streams from sensor tags and utilised linear chain CRF to assess the performance. In contrast to previous methods, the work presented in this chapter, considers segmentation and classification as a single task and segmentation is driven by a trained model instead of using heuristics.

Two notable advantages arise from our proposed approach. First, the proposed structured predictor based classifier does not require any explicit segmentation step in real-time prediction as in other data stream segmentation methods. As mentioned in Section 7.2, Part 1 of the output structure is assumed to contain sensor observations from a single activity and so the issues discussed with the use of fixed size segmentation methods in Chapter 5 are not present in our proposed approach. Secondly, the proposed structured predictor: i) does not rely on heuristics on the dataset; and ii) does not require the sensor to be worn on a specific location, as in the natural boundary segmentation proposed in Chapter 5 and the study in [48]. The methods proposed in Chapter 5 and [48], relied on specific sensor data stream characteristic as a result of the position of the sensor to segment the data stream. When using these methods for a different activity recognition task, a suitable data stream characteristic needs to be selected. Therefore, the method proposed in this chapter can be considered as a generic approach to activity classification. However, when using the proposed method in a different activity recognition task, a set of features suitable to the sensors used need to be used. For instance, as all the datasets we evaluated are from a sensor tag, and as sensor tag data streams are sparse, we only considered time domain statistical features. However, for battery powered sensors, frequency domain features can be used instead of features from the RFID platform.

From the data stream classification perspective, the proposed approach results in a delay no longer than the buffer length, i.e. $delay < \delta t$. This is due to the structured predictor predicting the activity label of Part 1 of the output structure. Based on the definition of the output structure, sensor observations in Part 1 is older compared to the sensor observations in Part 2. Predicting the activity for Part 1 can also be viewed as modelling the sequential nature of the activities based on future information with respect to the predicted activity instead of using the prior activity information as discussed in Chapter 6. Therefore, as future work, it would be interesting to model the sequential nature of activity data streams by combining the sequential learning approach discussed in Chapter 6 and the structured predictor proposed in this chapter.

Chapter 8

Recognition of Falls Using Dense Sensing in an Ambient Assisted Living Environment

8.1 Introduction

The main focus of this thesis has been fall prevention, particularly to recognise activities leading to falls such as getting out of the bed and getting out of the chair. Recognising activities that may result in falls can be as useful in creating warnings with the aim of staff promptly attending to the patient and thereby potentially reducing the risk of a fall or providing immediate care in case of a fall [138]. Although fall prevention is a growing area of study [25, 139] and is more desirable as opposed to systems focused on fall detection; there is still the need to detect falls timely, as they occur.

Until recently, falls were reported in the literature to be mostly unwitnessed [2, 3]. However, a significant recent and study by Robinovitch et al. [140], conducted over three years, determined the main causes of falls for people in long term facilities and recorded falls from real subjects in a video for analysis. This evidence is valuable as now researchers can study different ways of actual falls occurrences focusing on improving fall prevention and detection. In fact, Robinovitch et al. identified the main causes of falls as: i) incorrect weight shifting (41%); ii) tripping or stumbling (21%); iii) hit and bump (11%); and iv) loss of support (11%).

Based on the sensing method applied, present fall detection approaches in the literature can be categorised into two categories: i) body-worn approaches; and ii) environmental sensor based approaches. In addition there are a limited number of hybrid approaches that use both body-worn sensors as well as sensors deployed in the environment [64].

The studies relying on body-worn sensors often utilise encasings of battery powered devices that use straps to secure the device to the body of the user [18, 141–143]. However, use of such devices has been reported as an inconvenience, in particular, if the target population is older people [22].

In relation to environmental sensor approaches, the use of video images for monitoring as in Qian et al. [144] can raise privacy concerns from users [58]. The functionality of methods using instrumented objects such as walking canes [145] depends on the person carrying or effectively using the targeted object, which is not the case for people suffering from dementia or delirium—a condition that is common in older people. Alternatively, multiple approaches using floor based ambient sensing technologies have been developed and studied. In Klack et al. [146], several sensors and microcontrollers were embedded in the solid floor to monitor movements and detect falls. Similarly, Braun et al. [147] introduced an arrangement of capacitive sensors on floor mats to detect a person lying on the floor. Aud et al. [148] proposed the use of foils as a sensor to detect the presence of a foot and developed a smart carpet. Although these studies have described how their proposed hardware can be used to detect falls, they have not evaluated the detection of falls. On the other hand, Werner et al. [149] and Alwan et al. [150] exploited floor vibration on a impact to detect falls. In particular, Werner et al. [149] utilised accelerometers to capture floor vibrations while Alwan et al. [150] utilised a piezo transducer. However, these studies were tested with mannequins, which fall according to gravity as opposed to following the falling motion of a human. Furthermore, SensorFloor [151], a commercially available smart floor, uses specially built radio modules and proximity sensors to detect falls by identifying the shape of the covered area. However, SensorFloor is a wired solution that needs to be connected to an electrical power outlet since the sensors are active devices and its performance on the detection of falls has not been presented.

In contrast, we instrument the floor using passive Radio Frequency Identification (RFID) tags to detect falls. As opposed to the use of battery powered sensors, these tags are batteryless and can report their unique identifier (tag ID) wirelessly and almost indefinitely, without maintenance, to an RFID reader. Previously, we proposed a smart carpet embedded with multiple paper thin RFID tags, arranged to form a 2D grid that is interrogated using an RFID reader connected to two RFID antennas [152]. Our previous approach relied on exploiting the RSSI and binary tag observation data as well as their temporal variations of all the tags from our smart carpet grid collected by both antennas to provide a description of activities occurring on the carpet. Although we have demonstrated the ability of a dense deployment of RFID tag embedded smart carpet using a broadly scripted set of simulated falls and limited walking over the smart carpet, the method is influenced by the noise in the RSSI data; for instance movement of people in the environment. This method is also not scalable to a larger monitoring area as it requires information to be collected from all the tags in an entire monitoring area to detect falls.

In this chapter we propose a novel fall detection methodology which only relies on binary tag observation information—presence or absence of tags–and carries out a comprehensive evaluation that includes a rich set of broadly scripted walking patterns in addition to the scenarios described and evaluated in [152]. In particular, we formulate tag observations as a binary image (Section 8.3), which enabled us to detect falls by focusing on a region of unobserved tags as a possible location of a fall. This formulation not only removes the constraints on scalability in terms of the monitored area in our previous approach [152] but also makes the proposed approach agnostic to noise in RSSI. We also create 7 feature types and extracted 8 individual features using only binary tag observations. We use data from a possible region indicating a fall on the smart carpet to extract these features as opposed to 758 features extracted from all tags in the smart carpet as in our previous approach in [152] (Section 8.3.2). The fall detection algorithm proposed in this chapter is based on heuristics and machine learning (Section 8.3.1). Initially, the algorithm preselects the possible falls instances based on the area of unobserved tag regions. Then, the preselected tag regions are classified using machine learning to detect falls. Thus, the proposed algorithm is scalable in terms of the monitored areas, since it relies on specific regions of interest and the related tag observations in that region. Finally, we evaluate simulated falls in a setting that included a comprehensive set of walking activities with healthy young adults (Section 8.4.1). Evaluating with a comprehensive set of walking patterns is important because a significant amount of information from a real-world deployment will be related to walking instead of falls and other activities. The work presented in this chapter has been published in the journal of Pervasive and Mobile Computing [33].

8.2 Smart Carpet

The smart carpet showcases RFID tags integrated into the floor carpet as its core component (see Figure 8.1). Passive RFID tags are batteryless and use the RF energy radiated from the RFID reader antennas to power its circuitry (energise) and return a signal encoded with a digital identification code.

The smart carpet has a three-layered structure (see Figure 8.1(b)) where an array of RFID tags are sandwiched between a commercially available 3 mm nylon carpet

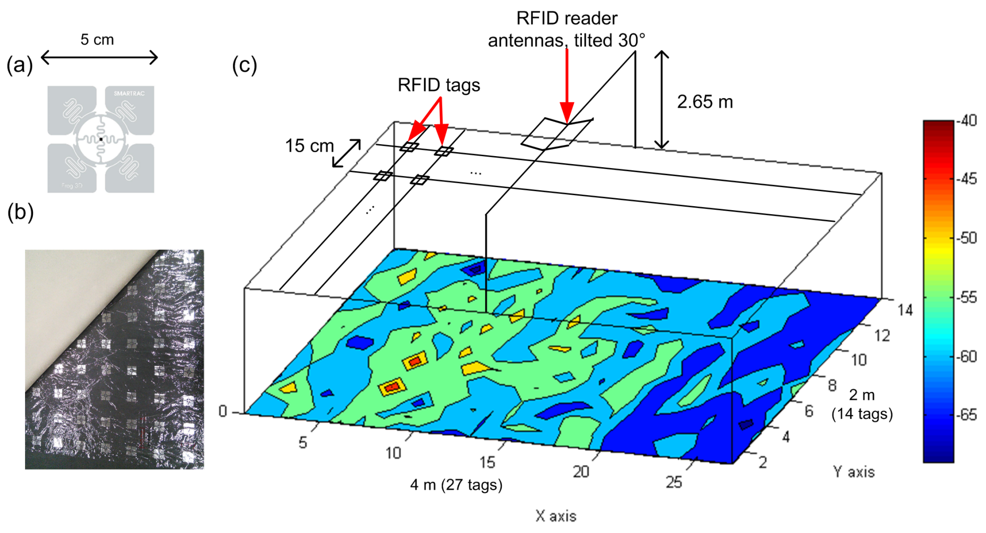


Fig. 8.1 Overall scheme of the experimental setting using a 4×2 m smart carpet: (a) Smartrack FROG 3D RFID tag indicating dimensions; (b) Smart carpet showing its three layers showing back of the carpet, the tag grid and base layer underneath; and (c) Temperature map of average RSSI readings per tag for a 4 hour trial without external interaction (empty room) demonstrating the different levels of power received from the smart carpet.

and a 3 mm anti-slip polyester sheet. The tag array is placed in between two sheets of 0.5 mm plastic to secure their position. This design allows the concealment of the tags as in a real-world deployment and protects the tags from impacts caused by footsteps as well as falls.

The RFID tag selection depends on certain conditions such as compliance with RFID communication protocols such as ISO 18000-6C [26] and the operational range that allows the maximisation of the area covered by an RFID reader antenna. Hence, we selected the commercially available Smartrack FROG 3D[1] tag (see Figure 8.1(a)) due to its operational range ($>$ 6 m when covered by 3 mm plastic) and its orientation insensitivity with respect to the RFID reader antenna.

The tag density determines the information content available for identifying falls and the time required to inventory all the tags in the smart carpet, thereby determining the latency of our approach. We placed tags in a grid, shown in Figure 8.1(b), with a 15 cm separation between tag centres on both directions. Given that the average width of a person (chest depth) is $\approx$ 24 cm [153], this layout ensures that at least one tag is occluded by a person's body even when lying on their side. Moreover, our dense sensor deployment requires 378 (27×14) tags to monitor an 8 m^2 (4 m$\times$2 m) carpet area.

[1]https://www.smartrac-group.com/frog-3d.html

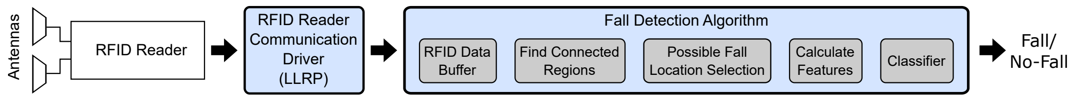


Fig. 8.2 Overview of the proposed methodology

We used two RFID reader antennas placed 2.65 m above the ground in opposing directions for data collection as shown in Figure 8.1(c). The antennas are inclined at approximately 30° to the horizontal plane, directed towards the smart carpet and connected to an Impinj Speedway R-420 (Firmware version: 5.2.1) RFID reader. Using this setup, we were able to inventory over 95% of tags within 0.5 s.

The RFID data from the smart carpet consists of two data sources: i) binary tag observation—unique tag ID of an observed tag; and ii) the antenna that observed the tag and the RSSI. However, these data sources are noisy. The tag observation noise arises as a result of the RFID reader not being able to interrogate all the RFID tags in its field in a single read cycle. The main reason for this is that the communication between the RFID reader and RFID tags is carried out using ISO 18000-6C based on the non-deterministic slotted Aloha random access protocol [26]. Furthermore, as explained in Section 2.4.4 for sensor tags, RSSI is affected by several factors such as [154, 155]: i) channel multipath propagation; ii) variations in incident power on tags; iii) interference from objects and equipment; and iv) thermal noise at the RFID reader receiver [155]. Consequently, for example, thermal noise can cause RSSI data to fluctuate in the absence of any changes in the environment. Furthermore, multipath propagations lead to changes in the RSSI, becoming dependent on the deployment and environment in which the RFID readers and tags operate, for instance, the dimensions of the corridor.

8.3 Methods

Figure 8.2 presents the overview of our proposed methodology. The RFID reader and antennas here are the main hardware components other than the proposed smart carpet. The RFID Reader Communication Driver and Fall Detection Algorithm are software components. RFID data from the smart carpet is captured by the RFID reader. The RFID Reader Communication Driver communicates with the RFID reader using an LLRP protocol [156] over a local area network and data obtained for 0.5 s intervals are forwarded to the Fall Detection Algorithm. The Fall Detection Algorithm, which is described in detail in Section 8.3.1, processes tag observation information from the smart carpet and recognises falls by first selecting possible falls locations using heuristics and subsequently classifying them using machine learning algorithms. In

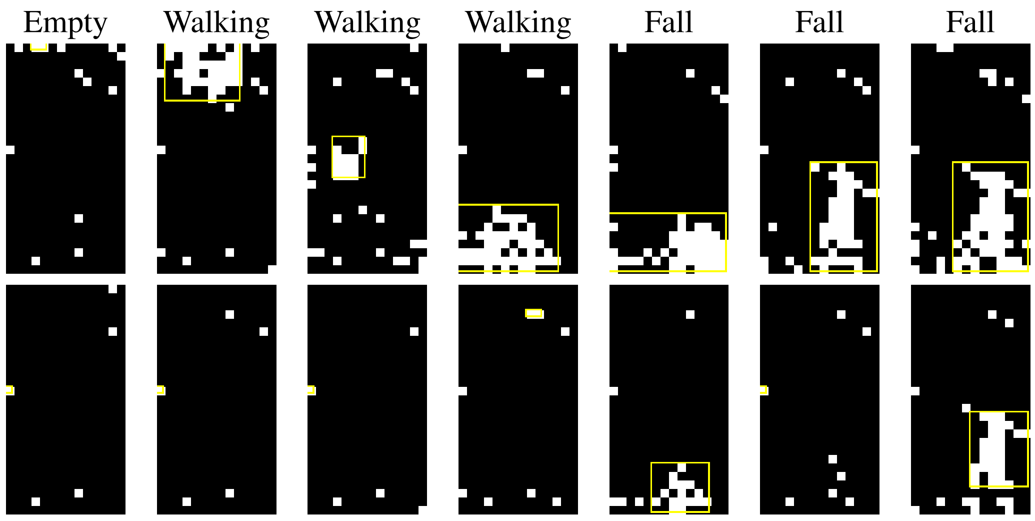


Fig. 8.3 Tag observation data as a sequence of images; top row $R_i^{0.5}$ and bottom row $R_i^{1.5}$. The rectangle in each image is the bounding box of the connected region with the maximum area.

future, we envision the fall detection algorithm to be completely integrated with the RFID reader firmware.

8.3.1 Fall Detection Algorithm

Given a set of readings for a 0.5 s period from the RFID reader $S_i^{0.5}$, we can formally represent the presence and absence of tags in a 27×14 binary matrix, $R_i^{0.5} \in \{0,1\}^{27\times 14}$. Here, presence of a tag is represented by 1 and 0 otherwise. The data represented by $R_i^{0.5}$ can be treated as a binary image. Figure 8.3 shows a sequence of binary images which represents a scenario of a person entering into the monitoring area (smart carpet) and subsequently falling.

The top sequence of images in Figure 8.3 were generated from $R_i^{0.5}$. From these images, we can see that a fall event can be identified by considering the properties of the shape of the unobserved tags while the person is lying on the ground. Furthermore, we can also see that some tags are unobserved randomly as a result of the tag observation noise described in Section 8.2. In our setting, the probability of a tag being unobserved given it is exposed to an RFID antenna ($P^{0.5}$) is at most 0.05; i.e. $P^{0.5} < 0.05$. The probability of a tag being unobserved can be reduced by considering a longer duration than 0.5 s. Therefore, we considered data for a 1.5 s period ($R_i^{1.5}$) by aggregating three consecutive sets of tag readings, $S_j^{0.5}$, $S_i^{1.5} = \bigcup_{j=i-2}^{i} S_j^{0.5}$, in order to lower the probability of a tag being not read significantly to $P^{1.5} < (P^{0.5})^3 = 1.25 \times 10^{-4}$. The sequence of images in the bottom row in Figure 8.3 shows the corresponding images

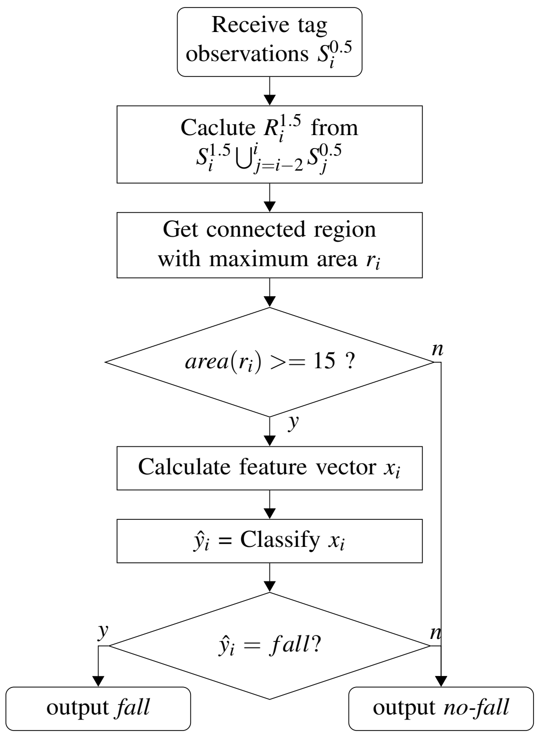

Receive tag observations $S_i^{0.5}$
Caclute $R_i^{1.5}$ from $S_i^{1.5} \bigcup_{j=i-2}^{i} S_j^{0.5}$
Get connected region with maximum area r_i
$area(r_i) >= 15$?
n
y
Calculate feature vector x_i
$\hat{y}_i$ = Classify x_i
$\hat{y}_i = fall$?
y
n
output *fall*
output *no-fall*

Fig. 8.4 Proposed fall detection algorithm that utilises binary tag observation information available from the smart carpet

to the top row in Figure 8.3 created based on $R^{1.5}$. Based on these observations, we proposed a novel algorithm to detect falls using data represented in $R_i^{1.5}$.

The proposed fall detection algorithm is illustrated in Figure 8.4. This algorithm receives $S_i^{0.5}$, which is the tag observation data for the i^{th} instance. The algorithm obtains $R_i^{1.5}$ using $S_i^{0.5}$ and past data over a 1 s period. By treating $R_i^{1.5}$ as a binary image, connected regions are obtained by using standard computer vision techniques and the region with the maximum area, r_i, only is retained. In the case of a fall, the area of r_i, $area(r_i)$, should be sufficiently large, as shown in Figure 8.3. Therefore, the algorithm makes an initial classification based on this heuristic, where, for instance, $area(r_i) < 15$ is classified as *no-fall*. The value 15 is selected considering the posture of a person lying on their side on the floor. Specifically, the average chest depth of an individual is 24 cm [153]. At least two tags will be covered across the body when a person is lying on their side on the smart carpet because the spacing between consecutive RFID tags in the smart carpet is 15 cm. Then, considering a person with a height of 1.5 m, at least 20 tags are expected to be covered while a person is lying on the ground on their side. Therefore, the value 15 is selected because $15 = 20 \times 0.75$ where we expected a minimum of 75% of the tags out of 20 tags to be covered considering the person with a smaller physical size. Our observations also confirm that for all the cases of falls $area(r_i) > 15$.

If $area(r_i) \geq 15$ then r_i is classified using a standard machine learning classifier to determine whether r_i represents a fall or not. This classification is carried out using features derived from the binary tag observation data from the connected region, r_i. This algorithm continues until data is available from an RFID reader.

It is important to note that the proposed fall detection algorithm reduces the adverse effects of noise, typically present in RFID data, in three ways. First, it only relies on binary tag observation information instead of considering noisy RSSI information, which is highly dependent on channel characteristics and the deployment environment. Secondly, only the information from a possible fall location is considered instead of utilising information from the entire smart carpet. This prevents the decision being affected by tag observation noise from other parts of the smart carpet. Thirdly, as shown in Figure 8.3, we use data for 1.5 s from the smart carpet to reduce the probability of a given tag not being observed randomly due to the nature of the random access protocol IS0 18000-6C used for interrogating RFID tags.

8.3.2 Feature Calculation

As stated previously, we rely on binary tag observation information alone for our falls classifier. Given a tag observation matrix $R_i^{1.5}$, we calculate 7 feature types that result in 8 individual features. These features are calculated based on the region with the maximum area r_i in $R_i^{1.5}$ and using only binary tag observation information for a possible falls location (r_i); therefore, they are more robust against environmental factors, unlike RSSI based features, and tag observations outside r_i. We denote the width and height of the bounding box of r_i as w_i and h_i respectively. The centre of the smart carpet where the antennas are placed is denoted as (c_x, c_y) (see Figure 8.1).

Area of r_i ($area(r_i)$) Area of r_i represents the number of tags that are occluded, possibly due to the human body.

Area of the bounding box of r_i The area is calculated as $w_i \times h_i$ and provides information relating to the nearby area of a possible fall.

Extent This represents the ratio of $area(r_i)$ to the area of the bounding box and is obtained as $\frac{a_i}{w_i \times h_i}$.

Ratio of the lengths of sides of the bounding box This is calculated as $\frac{max(w_i,h_i)}{min(w_i,h_i)}$. In case of a fall, based on the human physique, it is expected that one side of the bounding box be larger than the other.

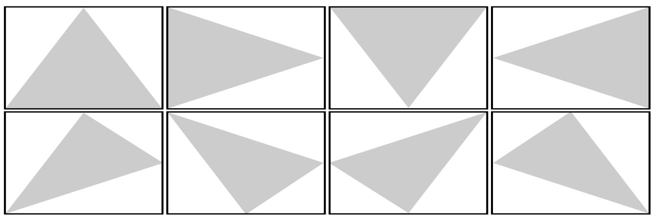


Fig. 8.5 Eight triangles within the bounding box considered for obtaining the value for the feature *triangular region*.

Centroid of r_i $((x_i, y_i))$ Centroid represents the location of a possible fall on the smart carpet. We use both x and y coordinate values (with reference to the smart carpet shown in Figure 8.1) to generate two features.

Distance from the centre of the smart carpet to the centroid (d) We obtain the Euclidean distance as $d = \sqrt{(x_i - c_x)^2 + (y_i - c_y)^2}$.

Triangular region Considering n number of triangular areas, t_j, this feature specifically relates r_i with a triangular area t_j within the bounding box. By considering n number of triangular areas, we obtain the maximum of the ratio between the area of intersection of r_i and t_j $(r_i \cap t_j)$ and $area(r_i)$; i.e. $max_{j=1}^{n} area(r_i \cap t_j)/area(r_i)$.

The feature *triangular region* specifically tries to capture the standing positions. When a participant is standing still, a large number of RFID tags are occluded as the human body attenuates the RF radiation to a significant extent and in such instances, unobserved tags are seen as a triangular region. Figure 8.5 shows the 8 triangle areas considered for calculating the *triangular region*. This feature provides information to discriminate falls from standing.

8.3.3 Machine Learning Classifier

We use four machine learning classifiers: i) Naïve Bayes—NB; ii) Linear Support Vector Machines (SVM)—LSVM; iii) Non-linear SVM with Radial Basis Function (RBF) kernel—NSVM; and iv) Random Forest—RF. All these classifiers can model i.i.d. classification problems and can perform in real time. For this study, NB implementation in Matlab 2015b was used. We utilised the linear SVM from the LibLinear package [123] for LSVM and SVM with the RBF kernel from LibSVM package [124] for NSVM. The RF classier utilised the TreeBagger in the Matlab 2015b environment.

In contrast to our previous method [152], machine learning classifiers are only used to classify a subset of $R_i^{1.5}$s. Therefore, it is important to select a subset of $R_i^{1.5}$s from the training dataset to achieve a good performance. To this end, given a training

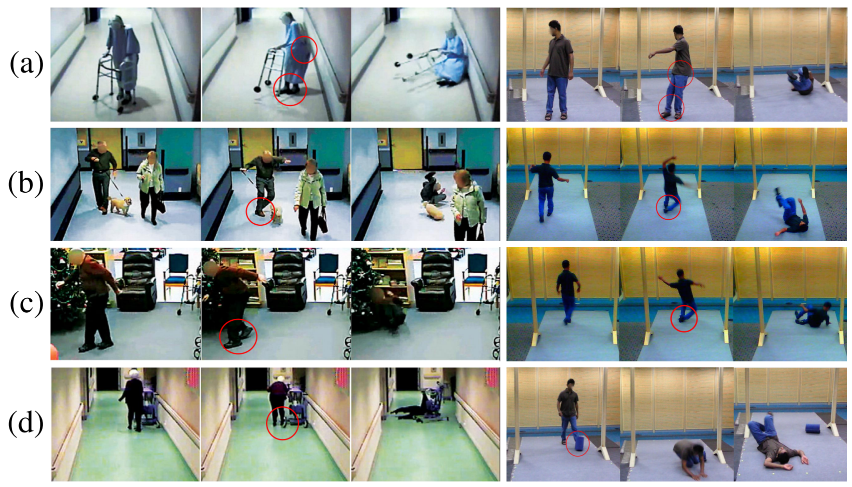


Fig. 8.6 Four types of falls as captured by Robinovitch et al. [140] to the left and as performed by young volunteers following the corresponding falls pattern in the dense sensing environment, depicted to the right. Red circles indicate the body motion that causes the fall, as reported in Robinovitch et al. [140]. (a) Fall 1: fall backwards after body rotation and incorrect weight shifting, (b) Fall 2: fall sideways after narrowing support base form by feet, (c) Fall 3: fall sideways after tripping over the other leg and (d) Fall 4: fall forwards after tripping over an object.

data set $D^{1.5} = R_i^{1.5}\, i = 1 \cdots T$, we only select $R_i^{1.5}$s with $area(r_i) > 10$. The condition $area(r_i) > 10$ ensures that there are sufficient *non-fall* instances for training.

8.4 Experiments and Analysis

8.4.1 Experiment Description

Thirteen adult participants (average age of 30.6±8.2 years old) and male to female ratio of 8:5 participated in the study. They were asked to follow broadly scripted activity routines on the smart carpet, and a researcher annotated the collected data in real time. The broadly scripted activity routines allowed us to capture a sufficient number of activities for classifier training and evaluation as well as capture natural variations of individual participants such as walking speed and stride length. We collected two datasets: i) simulated falls; and ii) walking patterns. During the evaluation, we combined both datasets into a single dataset. In total, we recorded 72 falls instances. These datasets are publicly available at the project website.[2]

Simulated Falls

Eight of the participants performed a series of broadly scripted simulated falls (back, sideways and forward) following the falls sequences as captured and described by Robinovitch et al. [140]. Specifically, the following types of falls were performed:

Fall 1: Fall backwards by rotating the body performing a weight shift as shown in Figure 8.6(a).

Fall 2: Fall sideways by narrowing the support base and incorrectly shifting weight as shown in Figure 8.6(b).

Fall 3: Fall sideways by tripping over one's leg shown in Figure 8.6(c).

Fall 4: Fall forwards by tripping over one leg with an obstacle on the path as shown in Figure 8.6(d).

Each participant fell two to four times per each type of fall over the smart carpet. A participant was free to start the fall and land anywhere in the carpet but had to fall at least once on both sides (left and right) of the wooden frame supporting the antennas. We labelled the data as a fall when the first part of the body other than the feet touched the floor rather than waiting for the person to be flat on the floor as we are interested in providing a prompt response. In addition to falls, these participants also performed a limited number of other activities including: i) walking across the smart carpet; and ii) walking into the smart carpet, squatting down to pick-up an object and leaving.

Walking Patterns

The main purpose of the walking dataset is to evaluate the fall detection algorithm with a set of comprehensive walking patterns. In a real-world deployment, falls are limited, and it is expected that a significant portion of information from the smart carpet will relate to walking activity. Ten participants walked across the smart carpet in eight paths. The first seven paths are illustrated in Figure 8.7. As the eighth path, we asked each participant to enter into the smart carpet at preferred locations, walk within the smart carpet in any preferred direction and leave the smart carpet at a preferred location. Furthermore, they were asked to stop and stand without moving at a random location on their path. In the study, none of the participants were instructed on how to walk, such as the walking speed or the stride length.

[2]Project website:http://autoidlab.cs.adelaide.edu.au/densesensing

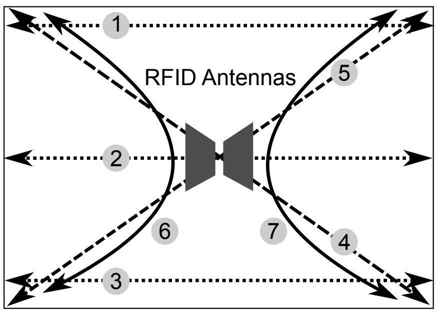


Fig. 8.7 Walking paths on the smart carpet performed by ten participants

8.4.2 Statistical Analysis

For fall detection, True Positives (TP) were defined to be at least a single prediction as a fall during an actual fall event recorded in the ground truth. All the incorrectly detected falls were treated as False Positives (FP). We considered true falls events that were missed by the fall detection algorithm as Falls Negatives (FN). For comparison between approaches, we utilised metrics:

$$Precision\,(P) = TP/(TP+FP) \tag{8.1}$$

$$Recall\,(R) = TP/(TP+FN) \tag{8.2}$$

$$F\text{-}score = 2.P.R/(P+R) \tag{8.3}$$

Evaluation of these metrics was performed using a leave-one-participant-out cross-validation procedure. Classifier training is carried out using the training data created as explained in Section 8.3.3. For parameter evaluation, given a set of training data for a participant, we randomly selected 80% from that dataset to train the classifier and obtain the classifier performance on the remaining 20% of the data set. The set of model parameters that maximises the F-score (which is obtained based on the standard binary classification performance) is selected. The final performance is obtained by analysing the entire dataset for a specific participant using the trained classification model for that participant. We report results for each participant independently as the model is trained without the data from the test participant.

8.5 Results

Table 8.1 compares the fall detection performance between the proposed algorithm with different classifiers and our previously published algorithm in [152] using the combined dataset described in Section 8.4.1. From these results, we can clearly observe that the recall for the proposed algorithm is at least 4% higher than our

Table 8.1 Overall fall detection performances

Method	TP*	FP*	FN*	Precision	Recall	F-score
NB	70	21	2	81%	97%	90%
LSVM†	70	23	2	75%	97%	85%
NSVM†	69	8	3	90%	96%	93%
RF†	70	12	2	85%	97%	91%
Previous algorithm in [152]	66	6	6	92%	92%	92%

* Precision, Recall and F-score are obtained based on the total number of TPs, FPs and FNs.

† Model parameters: LSVM-c:$2^{4.5}$; NSVM-c:2^3, g:2^2; RF-B:1000

Table 8.2 Fall detection performances

Participant #	1	2	3	4	5	6	7	8	9	10	11	12	13	Total
# of falls	11	8	0	9	8	8	0	0	10	10	8	0	0	72
Proposed algorithm														
TP	11	8	0	8	8	8	0	0	9	9	8	0	0	69
FP	1	0	0	4	1	0	0	1	0	0	1	0	0	8
FN	0	0	0	1	0	0	0	0	1	1	0	0	0	3
Previous algorithm described in [152]														
TP	9	8	0	9	6	8	0	0	10	8	8	0	0	66
FP	2	1	0	0	2	0	0	0	0	0	1	0	0	6
FN	2	0	0	0	2	0	0	0	0	2	0	0	0	6

previous algorithm. However, classifiers except NSVM depicted a higher number of FPs and consequently lower precision. Based on the F-score, the proposed algorithm with NSVM depicted a higher performance compared with other classifiers and we use this for comparison with our previous method.

We can observe that the proposed algorithm with NSVM depicts 3 more TPs and 2 more FPs when compared with our previous algorithm. Despite the increase in the number of FPs, the overall F-score has been improved. The main reason for this overall improvement is the ability of the proposed set of features to represent the information relating to falls events as shown by good classification performance, while being more immune (see Figure 8.3) to the noise present in the RFID data. Consequently, as denoted by the reduction in false negatives (misses) without adversely increasing false positives, the NSVM classifier is able to discriminate more falls instances successfully. Table 8.2 presents the results in terms of each participant. In particular, we can observe that there are 4 FPs under participant 4. To gain further insight we analysed the distribution of FPs with respect to each activity category as shown in Table 8.3. In Table 8.3, the *actual* column represents the number of episodes under each category. From Table 8.3, we can clearly see that the number of uninterested activities, i.e.,

Table 8.3 Distribution of false positives for different activities

Participant #	Empty		Walking		Standing		Pick		Total	
	Actual	FP	Actual	FP	Actual	FP	Actual	FP	Actual	FP
1	103	0	100	1	10	0	2	0	215	1
2	96	0	88	0	3	0	2	0	189	0
3	77	0	70	0	0	0	0	0	147	0
4	113	0	112	1	16	3	2	0	243	4
5	21	0	20	0	4	1	2	0	47	1
6	19	0	19	0	6	0	2	0	46	0
7	52	0	48	0	0	0	0	0	100	0
8	82	0	79	0	5	1	0	0	166	1
9	113	0	112	0	14	0	2	0	241	0
10	20	0	20	0	5	0	2	0	47	0
11	104	0	100	0	9	1	2	0	215	1
12	77	0	70	0	0	0	0	0	147	0
13	81	0	80	0	5	0	0	0	166	0
Total	958	0	918	2	77	6	16	0	1969	8

* Actual column under each category represents the continuous episodes of the particular category that consist of a sequence of sets of tag observations with the same class label.

activities except falls, (1969) is significantly higher than the number of falls (72) and the number of FPs (8) compared with the uninterested activities are negligible.

It is important to note that there are no FPs when the smart carpet is empty and while an object is picked up. However, two walking instances and six standing instances have been incorrectly classified as falls. Close observations on these FPs revealed that the FP for participant 4 while walking and the FP for participant 8 while standing occurred just after a picking-up activity, but no instance due to a picking-up activity has been misclassified as a fall. During picking-up, the occluded set of tags is different form the set of tags occluded while the participant is standing and walking. When a participant stands up quickly, tags which were previously covered respond with a delay because they are not sufficiently energised to perform communication at the first instance they are exposed. Then, together with the unobserved tags during standing and walking, a larger number of tags were unobserved and hence incorrectly classified as falls. A similar observation is made regarding the FP for participant 8; where the participant starts walking after standing still for a long period of time (>5 s).

When remaining FPs for standing are considered, they cover a large rectangular area of tags, which is similar to a fall. It is also observed that, in all these instances, misclassified regions are close to a boundary of the smart carpet. As evident from Figure 8.1(c), tags near the smart carpet boundaries receive less RF energy, mainly because they are far away from an RFID reader antenna. When a participant is near a

Table 8.4 Performance of previous fall detection approaches

Fall detection study	Accuracy	Recall (Sensitivity)	Specificity	Precision
Bianchi et al. [18]	90%			
Bourke et al. [141]		100%	100%	
Li et al. [142]		91%	92%	
He et al. [143]		92%	92%	
Qian et al. [144]	98%			
Method proposed in this chapter		96%		90%

boundary, the RF energy incident on tags near the boundary is further reduced as the human body absorbs RF energy. This causes a larger set of tags to be unobserved and subsequently unobserved regions are classified as a fall.

8.6 Discussion

In this chapter, we have presented a dense sensing method for the recognition of falls using batteryless RFID tags embedded into a carpet—a smart carpet—as opposed to instrumenting a person. In particular, we proposed a fall detection algorithm, which only relies on the binary image based on binary tag observations from the smart carpet. The novel algorithm performed better when compared with our previous algorithm [152] depicting a nearly 5% increase in performance in terms of TPs. Additionally, the proposed algorithm showed an overall F-score improvement compared with the previously proposed algorithm [152].

As mentioned in Section 8.1, fall detection have been approached using a wide array of technologies with varying success (see Table 8.4). Although, several research [146–151] have looked at embedding various sensors on a floor to capture falls and lying on the ground, none of them have conducted experiments on fall detection. These studies only demonstrate the technical feasibility of using the respective proposed technologies. On the other hand, studies in [18, 141–143] have utilised body-worn sensors (accelerometers and gyroscopes) to detect falls. Qian et al. [144] have utilised a wall mounted video camera to detect falls by identifying lying on ground. These methods have demonstrated good performance as listed in Table 8.4. Although we cannot make direct comparison with the previous methods, the prosed fall detection method presented in this chapter depicted comparable performance.

The proposed algorithm differs from our previously published algorithm [152] in four aspects. First, in our previous algorithm we have relied on information from the RSSI. As described in Section 8.2, the RSSI is noisy and is dependent on the environment. Therefore, the proposed algorithm is expected to be agnostic to the deployment environment. Secondly, the new formulation of the problem, i.e. considering tag observations as a binary image, allows us to focus only on information related to a possible fall. As described in Section 8.3.1, our proposed algorithm uses the connected region with the maximum area in the binary image to detect a fall event. In contrast, our previous algorithm considered the data from the entire smart carpet for fall detection. The proposed approach makes the fall detection algorithm robust as fall detection is not affected by noise in other parts of the smart carpet. Furthermore, this allows the monitoring area to be scaled without significantly increasing the complexity of the fall detection approach. Thirdly, the machine learning classifier is trained using only 8 features, in contrast to the 758 features used in our previous algorithm. These features capture the visual appearance of a region that may represent a possible fall and are calculated efficiently using standard computer vision techniques. Finally, as described in Section 8.3.1, initial heuristic based classification to eliminate *no-fall* events and to preselect potential falls events for classification by the learnt model reduce the processing burden further. The significantly reduced number of features and preselection using a heuristic reduce both the memory and computational complexity of our proposed fall detection algorithm.

The present research was evaluated against a set of data collected from young volunteers because it is not feasible to collect data, particularly falls, using a population of older people. The proposed approach relies on the shape and size of the unobserved tag patch as a result of rags being covered after a fall on the carpet. The shape of the unobserved tag patch is determined by the physical dimensions, such as height, and therefore it is not influenced by age or weight. In fact, the features accommodate variations of physical dimensions because they are calculated as ratios instead of raw values. Therefore, we expect feature values not to vary significantly with the people, but a detailed analysis of the effect of physical dimensions needs to be conducted in the future. Furthermore, the proposed approach needs to be evaluated with multiple people as well as with pets, because there will be multiple regions of unobserved tags. In such cases, we expect a similar performance to that demonstrated in this study because, as described in Section 8.3, the proposed approach only relies on binary tag observation information for possible falls locations instead of the data from the entire smart carpet to recognise falls. In the case of furnished environments, it is expected that stationary furniture will result in static unobserved regions on the smart carpet, which will be able to remove by comparing it with readings obtained only having furniture. In instances where furniture is mobile, such as wheeled hospital beds, the unobserved tag regions are expected to be significantly larger compared to a person lying on the ground. Furthermore, since the RFID tags in the proposed smart carpet is covered by a carpet and not directly exposed, we believe that they are less likely to get damaged by wheelchairs and wheeled hospital beds, walking sticks, and other furniture.

In conclusion, the presented approach provides high performance with an F-score of 93% and higher recall performance than previous work for the recognition of falls, as tested with young volunteers, using evidence from previous research on real older patients [140]. Our proposed method uses features only based on binary tag information and provides a response after 0.5 s of data collection and has a maximum response delay of up to 1.5 s to the occurrence of a fall event. Although the current approach is representative of and can be realised in carpeted halls, walkways and open spaces, further studies must include the use of walking aids and wheelchairs as well as expanding the monitoring environment to a home environment containing various pieces of furniture. In addition, the reduction of false positives should also be considered, without increasing the possibility of missing falls.

Chapter 9

Conclusion and Future Work

This thesis presents the use of passive RFID technology for monitoring ambulatory movements of older people. In particular, we focus on the use of an accelerometer embedded passive sensor enabled RFID tag (sensor tag) to capture movement information from hospitalised older patients to facilitate the development of wearable sensor based technological fall prevention interventions. Unlike battery powered sensors, passive sensor enabled RFID tags are expected to be unobtrusive for older people as they are small and lightweight. However, the main challenges of using passive sensor enabled RFID tags for activity recognition are sparsity and noise in the data stream. Therefore, several approaches have been proposed to monitor ambulatory movements accurately by tackling the aforementioned challenges.

Initially, in Chapter 2, we provided a background on Human Activity Recognition (HAR), common machine learning algorithms used in HAR and passive sensor enabled RFID tags. There we also presented the datasets, which have been collected from sensor tags, used in this thesis and their characteristics.

The work carried out in terms of sensor data acquisition from sensor tags has been presented in Chapter 3. A generic tag data format to identify sensor capabilities was initially proposed. Then we presented the generic middleware architecture that conforms to the EPC global architecture and an implantation to acquire sensor data from the sensor tags. Finally, performance of the middleware was demonstrated using an emulated experiment and through an application scenario.

In Chapter 4, we have proposed novel features for ambulatory monitoring based on human motion analysis. We also proposed a dynamic sensor data augmentation algorithm to facilitate the application of interpolation methods on sparse data streams. Initially, we extracted acceleration based features that require a data stream with a fixed sampling rate by interpolating the acceleration data from sensor tags using five interpolation methods with varying complexity and subsequently evaluated the activity recognition performance. There the cubic convolution interpolation method [122] depicted better activity recognition performance among other interpolants. Finally, activity recognition performances for combinations of the three feature sets from: i) the raw acceleration signal; ii) information from the RFID platform; and iii) the interpolated acceleration signal, were investigated. The inclusion of features available from the interpolated acceleration signal only resulted in marginal performance improvement over the use of the features possible from the raw data stream but at a significant computational cost.

In Chapter 5, we have proposed two real-time segmentation methods (non-overlapping and overlapping) that can segment the data streams on possible natural activity boundaries to mitigate the issues resulting from the use of sparse sensor data streams from sensor tags. The natural activity boundaries were detected based on heuristic analysis of the activity boundary score that is obtained based on the sensor wearer's trunk rotational motion. An ambulatory movement recognition algorithm to recognise multiple ambulatory movements by re-evaluating predictions from machine learning based activity classifiers was also proposed. The non-overlapping segmentation method performs better but is less responsive. The overlapping segmentation method is highly responsive but depicted lower performance. Despite the sparsity and noise present in the data streams from sensor tags, the proposed ambulatory monitoring framework was able to recognise multiple ambulatory movements successfully in real time.

We have proposed two real-time data stream classification algorithms in Chapter 6 and Chapter 7. These classifiers were trained using algorithms based on the Pegasos algorithm [90] and hence, they support online learning. The sequence learning algorithm proposed in Chapter 6 relied on the previous sensor observations for a specified interval and their predicted class labels to model the sequential nature of the sparse data stream. A data stream classification algorithm that does not require explicit segmentation was proposed in Chapter 7. Given a sensor data stream subsequence, the proposed algorithm predicts a two-part structure that divides the sensor data stream subsequence into two parts and activity associated with Part 1 that contains the initial set of sensor observations. Both sensor data stream classification algorithms were able to recognise activities successfully from data streams from sensor tags. More importantly, since these classifiers do not hold any assumptions on the nature of the data streams, they are expected to be used effectively in other data stream classification scenarios.

Finally, in Chapter 8, as opposed to using body-worn sensors, a dense sensing approach to detect falls using a smart carpet embedded with passive RFID tags was proposed. The proposed algorithm only uses binary tag information as RSSI is more susceptible the environment factors. To mitigate tag observation noise present in RFID data, the proposed fall detection algorithm first identifies a possible fall location and subsequently, the possible fall location is classified using a machine learning based classifier to recognise a fall using features extracted, which only consider the possible fall location. The proposed method provides a response after 0.5 s of data collection and has a maximum response delay of up to 1.5 s to the occurrence of a fall event.

There are three main future directions that stem from this thesis. First, this thesis used two broadly scripted activity datasets from older participants. Although the methods proposed in this thesis were able to recognise activities successfully using these datasets, future studies should consider a longitudinal monitoring study with unscripted activities to confirm the performance of the proposed methods and acceptability of sensor tags. In particular, the longitudinal monitoring study should be conducted in the day time as well as at night time. Although FOA dataset contained data collected from different rooms with similar RFID deployment configuration, cross-validation with different rooms will be important to establish a generalise performance under varying room settings.

Secondly, Chapters 6 and 7 presented two novel classification algorithms to classify sensor data streams. While the sequence learning algorithm presented in Chapter 6 considers past information, the use of data from Part 2, which arrived after the data in Part 1, for predicting the activity in Part 1 in the classification algorithm in Chapter 7 can be considered as relying on future activity information with respect to Part 1. Therefore, it is interesting to consider both previous and future information in a data stream classification algorithm to recognise activities. Such an algorithm is expected to depict better activity recognition performance when compared with the methods presented in Chapters 6 and 7.

Thirdly, modern RFID readers are capable of reporting the phase and the frequency of the received signal apart from the received signal strength. More recently, features extracted from the phase have been used in gesture recognition studies based on commercially available passive RFID tags [157, 158]. Therefore, the evaluation of features presented in Chapter 4 can be further extended by incorporating features calculated using phase information available from the RFID readers.

In summary, the contribution made in this thesis is a significant step towards human activity recognition, particularly using passive RFID technology. In this thesis, we show that despite the challenging nature of the data streams from passive sensor enabled RFID tags, human activities can be recognised accurately by utilising innovative data stream processing and classification methods. Finally, we believe the methods proposed in this thesis will facilitate the development of ambulatory monitoring systems to mitigate falls in hospitals and other older care settings.

References

[1] N. Kosse, K. Brands, J. Bauer, T. Hortobagyi, and C. Lamoth, “Sensor technologies aiming at fall prevention in institutionalized old adults: A synthesis of current knowledge,” *International Journal of Medical Informatics*, vol. 82, no. 9, pp. 743–752, 2013.

[2] E. B. Hitcho, M. J. Krauss, S. Birge, W. Claiborne Dunagan, I. Fischer, S. Johnson, P. A. Nast, E. Costantinou, and V. J. Fraser, “Characteristics and circumstances of falls in a hospital setting,” *Journal of General Internal Medicine*, vol. 19, no. 7, pp. 732–739, 2004.

[3] M. Johnson, A. George, and D. T. Tran, “Analysis of falls incidents: Nurse and patient preventive behaviours,” *International Journal of Nursing Practice*, vol. 17, no. 1, pp. 60–66, 2011.

[4] National Patient Safety Agency, “The third report from the Patient Safety Observatory: Slips, trips and falls in hospital,” National Patient Safety Agency, Tech. Rep., 2007. [Online]. Available: http://www.nrls.npsa.nhs.uk/EasySiteWeb/getresource.axd?AssetID=61390&type=full&servicetype=Attachment

[5] D. Oliver, F. Healey, and T. P. Haines, “Preventing falls and fall-related injuries in hospitals,” *Clinics in geriatric medicine*, vol. 26, no. 4, pp. 645–692, 2010.

[6] C. Abreu, A. Mendes, J. Monteiro, and F. R. Santos, “Falls in hospital settings: a longitudinal study,” *Revista latino-americana de enfermagem*, vol. 20, no. 3, pp. 597–603, 2012.

[7] K. D. Hill, M. Vu, and W. Walsh, “Falls in the acute hospital setting - impact on resource utilisation,” *Australian Health Review*, vol. 31, no. 3, pp. 471–477, 2007.

[8] K. Rapp, C. Becker, I. Cameron, H. König, and G. Büchele, “Epidemiology of falls in residential aged care: analysis of more than 70,000 falls from residents of bavarian nursing homes,” *Journal of the American Medical Directors Association*, vol. 13, no. 2, pp. 187.e1–187.e6, 2012.

[9] Australian Commission on Safety and Quality in Health Care, *Preventing falls and harm from falls in older people: best practice guidelines for Australian community care*. Sydney: Australian Commission on Safety and Quality in Health Care, 2009.

[10] I. D. Cameron, G. R. Murray, L. D. Gillespie, M. C. Robertson, K. D. Hill, R. G. Cumming, and N. Kerse, “Interventions for preventing falls in older people

in nursing care facilities and hospitals," in *Cochrane Database of Systematic Reviews*. John Wiley & Sons, Ltd, 2010.

[11] C. A. Brand and V. Sundararajan, "A 10-year cohort study of the burden and risk of in-hospital falls and fractures using routinely collected hospital data," *Quality & Safety in Health Care*, vol. 19, no. 6, p. e51, 2010.

[12] R. Visvanathan, D. C. Ranasinghe, R. L. Shinmoto Torres, and K. Hill, "Framework for preventing falls in acute hospitals using passive sensor enabled radio frequency identification technology," in *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2012, pp. 5858–5862.

[13] E. Capezuti, B. L. Brush, S. Lane, H. U. Rabinowitz, and M. Secic, "Bed-exit alarm effectiveness," *Archives of Gerontology and Geriatrics*, vol. 49, no. 1, pp. 27–31, 2009.

[14] J. Hilbe, E. Schulc, B. Linder, and C. Them, "Development and alarm threshold evaluation of a side rail integrated sensor technology for the prevention of falls," *International Journal of Medical Informatics*, vol. 79, no. 3, pp. 173–180, 2010.

[15] M. Bruyneel, W. Libert, and V. Ninane, "Detection of bed-exit events using a new wireless bed monitoring assistance," *International Journal of Medical Informatics*, vol. 80, no. 2, pp. 127–132, 2011.

[16] K.-H. Wolf, K. Hetzer, H. M. z. Schwabedissen, B. Wiese, and M. Marschollek, "Development and pilot study of a bed-exit alarm based on a body-worn accelerometer," *Zeitschrift fur Gerontologie und Geriatrie*, vol. 46, no. 8, pp. 727–733, 2013.

[17] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *Pervasive Computing*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, no. 3001, pp. 1–17.

[18] F. Bianchi, S. Redmond, M. Narayanan, S. Cerutti, and N. Lovell, "Barometric pressure and triaxial accelerometry-based falls event detection," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 6, pp. 619 –627, 2010.

[19] N. Noury, A. Galay, J. Pasquier, and M. Ballussaud, "Preliminary investigation into the use of Autonomous Fall Detectors," in *International Conference of the IEEE Engineering in Medicine and Biology Society*, 2008, pp. 2828–2831.

[20] R. Steele, A. Lo, C. Secombe, and Y. K. Wong, "Elderly persons' perception and acceptance of using wireless sensor networks to assist healthcare," *International Journal of Medical Informatics*, vol. 78, no. 12, pp. 788–801, 2009.

[21] G. Mountain, S. Wilson, C. Eccleston, S. Mawson, J. Hammerton, T. Ware, H. Zheng, R. Davies, N. Black, N. Harris, and others, "Developing and testing a telerehabilitation system for people following stroke: issues of usability," *Journal of Engineering Design*, vol. 21, no. 2-3, pp. 223–236, 2010.

[22] M. Gövercin, Y. Költzsch, M. Meis, S. Wegel, M. Gietzelt, J. Spehr, S. Winkelbach, M. Marschollek, and E. Steinhagen-Thiessen, “Defining the user requirements for wearable and optical fall prediction and fall detection devices for home use,” *Informatics for Health and Social Care*, vol. 35, no. 3-4, pp. 177–187, 2010.

[23] A. P. Sample, D. J. Yeager, P. S. Powledge, A. V. Mamishev, and J. R. Smith, “Design of an RFID-based battery-free programmable sensing platform,” *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 11, pp. 2608–2615, 2008.

[24] T. Kaufmann, D. C. Ranasinghe, M. Zhou, and C. Fumeaux, “Wearable quarter-wave folded microstrip antenna for passive UHF RFID applications,” *International Journal of Antennas and Propagation*, vol. 2013, 2013.

[25] R. L. Shinmoto Torres, D. C. Ranasinghe, Q. Shi, and A. P. Sample, “Sensor enabled wearable RFID technology for mitigating the risk of falls near beds,” in *IEEE International Conference on RFID*, 2013, pp. 191–198.

[26] EPCglobal Inc, “EPC™ Radio-Frequency Identity Protocols Generation-2 UHF RFID,” Oct. 2008. [Online]. Available: http://www.gs1.org/gsmp/kc/epcglobal/uhfc1g2/uhfc1g2_1_2_0-standard-20080511.pdf

[27] EPCglobal Inc, “Epcglobal ratified standards.” [Online]. Available: http://www.gs1.org/gsmp/kc/epcglobal/

[28] A. Wickramasinghe, D. C. Ranasinghe, and A. Sample, “WINDWare: Supporting Ubiquitous Computing with Passive Sensor Enabled RFID,” in *IEEE International Conference on RFID*, 2014.

[29] A. Wickramasinghe and D. C. Ranasinghe, “Recognising Activities in Real Time Using Body Worn Passive Sensors With Sparse Data Streams: To Interpolate or Not To Interpolate?” in *International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2015.

[30] A. Wickramasinghe and D. C. Ranasinghe, “Ambulatory monitoring using passive computational RFID sensors,” *IEEE Sensors Journal*, vol. 15, no. 10, pp. 5859–5869, 2015.

[31] C. Sutton and A. McCallum, “An Introduction to Conditional Random Fields,” *arXiv:1011.4088 [stat]*, 2010, arXiv: 1011.4088.

[32] A. Wickramasinghe, D. C. Ranasinghe, C. Fumeaux, K. D. Hill, and R. Visvanathan, “Sequence learning with passive rfid sensors for real time bed-egress recognition in older people,” *IEEE Journal of Biomedical and Health Informatics*, no. 99, 2016, [online].

[33] A. Wickramasinghe, R. L. Shinmoto Torres, and D. C. Ranasinghe, “Recognition of falls using dense sensing in an ambient assisted living environment,” *Pervasive and Mobile Computing*, pp. –, 2016.

[34] D. Karantonis, M. Narayanan, M. Mathie, N. Lovell, and B. Celler, “Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring,” *IEEE Transactions on Information Technology in Biomedicine*, vol. 10, no. 1, pp. 156–167, 2006.

[35] A. Khan, Y.-K. Lee, S. Lee, and T.-S. Kim, “A Triaxial Accelerometer-Based Physical-Activity Recognition via Augmented-Signal Features and a Hierarchical Recognizer,” *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 5, pp. 1166–1172, 2010.

[36] Z. Wang, M. Jiang, Y. Hu, and H. Li, “An Incremental Learning Method Based on Probabilistic Neural Networks and Adjustable Fuzzy Clustering for Human Activity Recognition by Using Wearable Sensors,” *IEEE Transactions on Information Technology in Biomedicine*, vol. 16, no. 4, pp. 691–699, 2012.

[37] W. Jiang and Z. Yin, “Human Activity Recognition using Wearable Sensors by Deep Convolutional Neural Networks,” in *ACM International conference on Multimedia*, 2015, pp. 1307–1310.

[38] F. J. Ordóñez and D. Roggen, “Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition,” *Sensors*, vol. 16, no. 1, p. 115, 2016.

[39] M. Stikic, K. Van Laerhoven, and B. Schiele, “Exploring semi-supervised and active learning for activity recognition,” in *IEEE International Symposium on Wearable Computers*, 2008, pp. 81–88.

[40] M. Zhang and A. A. Sawchuk, “Motion primitive-based human activity recognition using a bag-of-features approach,” in *International Health Informatics Symposium*, 2012, pp. 631–640.

[41] L. Gao, A. K. Bourke, and J. Nelson, “Evaluation of accelerometer based multi-sensor versus single-sensor activity recognition systems,” *Medical Engineering & Physics*, vol. 36, no. 6, pp. 779–785, 2014.

[42] J. Korpela, T. Maekawa, J. Eberle, D. Chakraborty, and K. Aberer, “Tree-structured classifier for acceleration-based activity and gesture recognition on smartwatches,” in *IEEE International Conference on Pervasive Computing and Communication Workshops*, 2016, pp. 1–4.

[43] N. Kern, B. Schiele, and A. Schmidt, “Multi-sensor activity context detection for wearable computing,” in *Ambient Intelligence*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2003, no. 2875, pp. 220–232.

[44] S. A. Muhammad, B. N. Klein, K. V. Laerhoven, and K. David, “A Feature Set Evaluation for Activity Recognition with Body-Worn Inertial Sensors,” in *Constructing Ambient Intelligence*, ser. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2012, no. 277, pp. 101–109.

[45] C. A. Ronao and S.-B. Cho, “Human activity recognition with smartphone sensors using deep learning neural networks,” *Expert Systems with Applications*, vol. 59, pp. 235–244, 2016.

[46] R. L. Shinmoto Torres, D. C. Ranasinghe, and Q. Shi, “Evaluation of wearable sensor tag data segmentation approaches for real time activity classification in elderly,” in *International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2013.

[47] M. Shoaib, S. Bosch, O. D. Incel, H. Scholten, and P. J. M. Havinga, “Complex Human Activity Recognition Using Smartphone and Wrist-Worn Motion Sensors,” *Sensors*, vol. 16, no. 4, p. 426, 2016.

[48] H. Junker, O. Amft, P. Lukowicz, and G. Tröster, “Gesture spotting with body-worn inertial sensors to detect user activities,” *Pattern Recognition*, vol. 41, no. 6, pp. 2010–2024, 2008.

[49] N. C. Krishnan and D. J. Cook, “Activity recognition on streaming sensor data,” *Pervasive and Mobile Computing*, vol. 10, Part B, pp. 138–154, 2014.

[50] L. Chen, J. Hoey, C. Nugent, D. Cook, and Z. Yu, “Sensor-Based Activity Recognition,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 6, pp. 790–808, 2012.

[51] O. Lara and M. Labrador, “A Survey on Human Activity Recognition using Wearable Sensors,” *IEEE Communications Surveys Tutorials*, vol. 15, no. 3, pp. 1192–1209, 2013.

[52] O. C. Ann and L. B. Theng, “Human activity recognition: A review,” in *IEEE International Conference on Control System, Computing and Engineering*, 2014, pp. 389–393.

[53] C. Liu and P. Yuen, “A boosted co-training algorithm for human action recognition,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 9, pp. 1203–1213, 2011.

[54] W.-H. Ong, L. Palafox, and T. Koseki, “Investigation of feature extraction for unsupervised learning in human activity detection,” *Bulletin of Networking, Computing, Systems, and Software*, vol. 2, no. 1, 2013.

[55] D. Cook, “Learning setting-generalized activity models for smart spaces,” *IEEE Intelligent Systems*, vol. 27, no. 1, pp. 32 –38, 2012.

[56] M. Buettner, R. Prasad, M. Philipose, and D. Wetherall, “Recognizing daily activities with RFID-based sensors,” in *International conference on Ubiquitous computing*, 2009, pp. 51–60.

[57] L. A. Schwarz, D. Mateus, and N. Navab, “Recognizing multiple human activities and tracking full-body pose in unconstrained environments,” *Pattern Recognition*, vol. 45, no. 1, pp. 11–23, 2012.

[58] G. Demiris, B. K. Hensel, M. Skubic, and M. Rantz, “Senior residents’ perceived need of and preferences for "smart home" sensor technologies,” *International Journal of Technology Assessment in Health Care*, vol. 24, pp. 120–124, 2008.

[59] F. Wagner, J. Basran, and V. D. Bello-Haas, “A Review of Monitoring Technology for Use With Older Adults,” *Journal of Geriatric Physical Therapy*, vol. 35, no. 1, pp. 28–34, 2012.

[60] C. Doukas and I. Maglogiannis, “Advanced patient or elder fall detection based on movement and sound data,” in *International Conference on Pervasive Computing Technologies for Healthcare*, 2008, pp. 103–107.

[61] D. C. Ranasinghe, R. L. Shinmoto Torres, K. Hill, and R. Visvanathan, “Low cost and batteryless sensor-enabled radio frequency identification tag based approaches to identify patient bed entry and exit posture transitions,” *Gait & Posture*, vol. 39, no. 1, pp. 118–123, 2014.

[62] B. Najafi, K. Aminian, A. Paraschiv-Ionescu, F. Loew, C. Bula, and P. Robert, “Ambulatory system for human motion analysis using a kinematic sensor: Monitoring of daily physical activity in the elderly,” *IEEE Transactions on Biomedical Engineering*, vol. 50, no. 6, pp. 711–723, 2003.

[63] J. P. Varkey, D. Pompili, and T. A. Walls, “Human motion recognition using a wireless sensor-based wearable system,” *Personal and Ubiquitous Computing*, vol. 16, no. 7, pp. 897–910, 2012.

[64] C. Doukas and I. Maglogiannis, “Emergency fall incidents detection in assisted living environments utilizing motion, sound, and visual perceptual components,” *IEEE Transactions on Information Technology in Biomedicine*, vol. 15, no. 2, pp. 277–289, 2011.

[65] Y. Nam, S. Rho, and C. Lee, “Physical activity recognition using multiple sensors embedded in a wearable device,” *ACM Transactions on Embedded Computing Systems*, vol. 12, no. 2, p. 26:1–26:14, 2013.

[66] C. Zhu and W. Sheng, “Realtime recognition of complex daily activities using dynamic Bayesian network,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 3395–3400.

[67] Y. Guo, W. He, and C. Gao, “Human Activity Recognition by Fusing Multiple Sensor Nodes in the Wearable Sensor Systems,” *Journal of Mechanics in Medicine & Biology*, vol. 12, no. 5, pp. –1, 2012.

[68] D. Wang, Z. sand Wu, J. Chen, A. Ghoneim, and M. A. Hossain, “A Triaxial Accelerometer-Based Human Activity Recognition via EEMD-Based Features and Game-Theory-Based Feature Selection,” *IEEE Sensors Journal*, vol. 16, no. 9, pp. 3198–3207, 2016.

[69] A. Godfrey, A. Bourke, G. Ólaighin, P. van de Ven, and J. Nelson, “Activity classification using a single chest mounted tri-axial accelerometer,” *Medical engineering & physics*, vol. 33, no. 9, pp. 1127–1135, 2011.

[70] R. Matsushige, K. Kakusho, and T. Okadome, “Semi-supervised learning based activity recognition from sensor data,” in *IEEE Global Conference on Consumer Electronics*, 2015, pp. 106–107.

[71] J. Wang, R. Chen, X. Sun, M. F. She, and Y. Wu, “Recognizing human daily activities from accelerometer signal,” *Procedia Engineering*, vol. 15, pp. 1780–1786, 2011.

[72] O. Thomas, P. Sunehag, G. Dror, S. Yun, S. Kim, M. Robards, A. Smola, D. Green, and P. Saunders, "Wearable sensor activity analysis using semi-Markov models with a grammar," *Pervasive and Mobile Computing*, vol. 6, no. 3, pp. 342–350, 2010.

[73] M. Zhang and A. Sawchuk, "Human Daily Activity Recognition With Sparse Representation Using Wearable Sensors," *IEEE Journal of Biomedical and Health Informatics*, vol. 17, no. 3, pp. 553–560, 2013.

[74] J. Mäntyjärvi, J. Himberg, and T. Seppänen, "Recognizing human motion with multiple acceleration sensors," in *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 2, 2001, pp. 747–752 vol.2.

[75] J. Suutala, S. Pirttikangas, and J. Röning, "Discriminative Temporal Smoothing for Activity Recognition from Wearable Sensors," in *Ubiquitous Computing Systems*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, no. 4836, pp. 182–195, dOI: 10.1007/978-3-540-76772-5_15.

[76] R. Liu, T. Chen, and L. Huang, "Research on human activity recognition based on active learning," in *International Conference on Machine Learning and Cybernetics*, vol. 1, 2010, pp. 285–290.

[77] A. Mannini and A. M. Sabatini, "Accelerometry-Based Classification of Human Activities Using Markov Modeling," *Computational Intelligence and Neuroscience, Computational Intelligence and Neuroscience*, vol. 2011, 2011, p. e647858, 2011.

[78] E. Sazonov, N. Hegde, R. Browning, E. Melanson, and N. Sazonova, "Posture and Activity Recognition and Energy Expenditure Estimation in a Wearable Platform," *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 4, 2015.

[79] M. Ermes, J. Pärkkä, J. Mäntyjärvi, and I. Korhonen, "Detection of Daily Activities and Sports With Wearable Sensors in Controlled and Uncontrolled Conditions," *IEEE Transactions on Information Technology in Biomedicine*, vol. 12, no. 1, pp. 20–26, 2008.

[80] S. Liu, R. Gao, D. John, J. Staudenmayer, and P. Freedson, "Multisensor Data Fusion for Physical Activity Assessment," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 3, pp. 687–696, 2012.

[81] B. Ariyatum, R. Holland, D. Harrison, and T. Kazi, "The future design direction of smart clothing development," *Journal of the Textile Institute*, vol. 96, no. 4, pp. 199–210, 2005.

[82] R. Y. W. Lee and A. J. Carlisle, "Detection of falls using accelerometers and mobile phone technology," *Age and Ageing*, vol. 40, no. 6, pp. 690–696, 2011.

[83] J. Lester, T. Choudhury, N. Kern, G. Borriello, and B. Hannaford, "A Hybrid Discriminative/Generative Approach for Modeling Human Activities." in *International Joint Conference on Artificial Intelligence*, vol. 5, 2005, pp. 766–772.

[84] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Annual conference on Computational learning theory*, 1998, pp. 92–100.

[85] B. Logan, J. Healey, M. Philipose, E. M. Tapia, and S. Intille, "A Long-Term Evaluation of Sensing Modalities for Activity Recognition," in *UbiComp 2007: Ubiquitous Computing*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, no. 4717, pp. 483–500.

[86] Y. Wu, R. Chen, J. Wang, X. Sun, and M. F. She, "Intelligent clothing for automated recognition of human physical activities in free-living environment," *Journal of the Textile Institute*, vol. 103, no. 8, pp. 806–816, 2012.

[87] L. Wang, T. Gu, H. Xie, X. Tao, J. Lu, and Y. Huang, "A wearable rfid system for real-time activity recognition using radio patterns," in *Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Springer, 2013.

[88] T. Miu, P. Missier, and T. Plotz, "Bootstrapping Personalised Human Activity Recognition Models Using Online Active Learning," in *IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, 2015, pp. 1138–1147.

[89] V. N. Vapnik and V. Vapnik, *Statistical learning theory*. Wiley New York, 1998, vol. 2.

[90] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: primal estimated sub-gradient solver for SVM," *Mathematical Programming*, vol. 127, no. 1, pp. 3–30, 2011.

[91] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[92] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[93] Trevor Hastie, Robert Tibshirani, and Jerome Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY: Springer New York, 2009.

[94] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[95] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[96] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[97] R. Want, "An introduction to RFID technology," *IEEE Pervasive Computing*, vol. 5, no. 1, pp. 25–33, 2006.

[98] S. Miles, “Introduction to RFID history and markets,” in *RFID Technology and Applications*. Cambridge University Press, 2008.

[99] D. C. Ranasinghe, Q. Z. Sheng, and S. Zeadally, *Unique Radio Innovation for the 21st Century: Building Scalable and Global RFID Networks*, 1st ed. New York, NY, USA: Springer-Verlag New York, Inc., 2010.

[100] Y. Su, A. Wickramasinghe, and D. C. Ranasinghe, “Investigating Sensor Data Retrieval Schemes for Multi-Sensor Passive RFID Tags,” in *IEEE International Conference on RFID*, 2015, pp. 201–208.

[101] R. L. Shinmoto Torres, D. C. Ranasinghe, Q. Shi, and A. v. d. Hengel, “Learning from Imbalanced Multiclass Sequential Data Streams Using Dynamically Weighted Conditional Random Fields,” *arXiv:1603.03627 [cs]*, 2016, arXiv: 1603.03627.

[102] C. A. Balanis, *Antenna Theory: Analysis and Design*. John Wiley & Sons, 2012.

[103] D. C. Ranasinghe, K. S. Leong, M. Ng, D. W. Engels, and P. H. Cole, “A distributed architecture for a ubiquitous RFID sensing network,” in *International Conference on Intelligent Sensors, Sensor Networks and Information Processing Conference*, 2005, pp. 7–12.

[104] Z. Cheng, X. Zhang, Y. Dai, and Y. Lu, “Design techniques of low-power embedded EEPROM for passive RFID tag,” *Analog Integrated Circuits and Signal Processing*, vol. 74, no. 3, pp. 585–589, 2013.

[105] Oracle, “Data sheet: Oracle warehouse management,” 2014. [Online]. Available: http://www.oracle.com/us/products/applications/ebusiness/054354.pdf

[106] IBM, “IBM - WebSphere sensor events - united states,” 2014. [Online]. Available: http://www-03.ibm.com/software/products/en/webssenseven/

[107] SAP, “SAP - SAP auto-ID infrastructure.” [Online]. Available: http://www.sap.com/platform/netweaver/autoidinfrastructure.epx

[108] X. Su, C.-C. Chu, B. S. Prabhu, and R. Gadh, “On the identification device management and data capture via WinRFID1 edge-server,” *IEEE Systems Journal*, vol. 1, no. 2, pp. 95–104, 2007.

[109] Fosstrak, “Fosstrak - open source RFID platform - google project hosting,” 2013. [Online]. Available: https://code.google.com/p/fosstrak/

[110] W. Wang, J. Sung, and D. Kim, “Complex event processing in EPC sensor network middleware for both RFID and WSN,” in *IEEE International Symposium on Object Oriented Real-Time Distributed Computing*, 2008, pp. 165–169.

[111] K. Gama, L. Touseau, and D. Donsez, “Combining heterogeneous service technologies for building an internet of things middleware,” *Computer Communications*, vol. 35, no. 4, pp. 405–417, 2012.

[112] D. Bade and W. Lamersdorf, “An agent-based event processing middleware for sensor networks and RFID systems,” *The Computer Journal*, vol. 54, no. 3, pp. 321–331, 2011.

[113] I. Abad, C. Cerrada, J. A. Cerrada, R. Heradio, and E. Valero, “Managing RFID sensors networks with a general purpose RFID middleware,” *Sensors*, vol. 12, no. 6, pp. 7719–7737, 2012.

[114] Y. Igarashi, K. Miyazaki, Y. Sato, and J. Mitsugi, “A network architecture for fast retrieval of user memory data from sensor RF tags,” in *IEEE International Conference on RFID*, 2013, pp. 184–190.

[115] C. Floerkemeier, C. Roduner, and M. Lampe, “Rfid application development with the Accada middleware platform,” *Systems Journal, IEEE*, vol. 1, no. 2, pp. 82–94, 2007.

[116] T. S. López, D. C. Ranasinghe, B. Patkai, and D. McFarlane, “Taxonomy, technology and applications of smart objects,” *Information Systems Frontiers*, vol. 13, no. 2, pp. 281–300, 2011.

[117] O. Banos, M. Damas, H. Pomares, A. Prieto, and I. Rojas, “Daily living activity recognition based on statistical feature quality group selection,” *Expert Systems with Applications*, vol. 39, no. 9, pp. 8013–8021, 2012.

[118] J. Fessler and B. Sutton, “Nonuniform fast fourier transforms using min-max interpolation,” *IEEE Transactions on Signal Processing*, vol. 51, no. 2, pp. 560–574, 2003.

[119] L. Greengard and J. Lee, “Accelerating the nonuniform fast fourier transform,” *SIAM Review*, vol. 46, no. 3, pp. 443–454, 2004.

[120] Z. He and L. JIN, “Activity recognition from acceleration data based on discrete consine transform and SVM,” in *IEEE International Conference on Systems, Man and Cybernetics*, Oct. 2009, pp. 5041–5044.

[121] J. d. Villiers, “Error analysis for polynomial interpolation,” in *Mathematics of Approximation*, ser. Mathematics Textbooks for Science and Engineering. Atlantis Press, 2012, no. 1, pp. 25–35.

[122] R. Keys, “Cubic convolution interpolation for digital image processing,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, no. 6, pp. 1153–1160, 1981.

[123] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “LIBLINEAR: A Library for Large Linear Classification,” *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[124] C.-C. Chang and C.-J. Lin, “LIBSVM: A Library for Support Vector Machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, 2011.

[125] H. He and E. Garcia, “Learning from Imbalanced Data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[126] P. Casale, O. Pujol, and P. Radeva, “Human activity recognition from accelerometer data using a wearable device,” in *Pattern Recognition and Image Analysis*. Springer Berlin Heidelberg, 2011, pp. 289–296.

[127] R. I. Shorr, A. M. Chandler, L. C. Mion, T. M. Waters, M. Liu, M. J. Daniels, L. A. Kessler, and S. T. Miller, "Effects of an intervention to increase bed alarm use to prevent falls in hospitalized patients," *Annals of Internal Medicine*, vol. 157, no. 10, pp. 692–699, 2012.

[128] H. Knight, J. Lee, and H. Ma, "Chair alarm for patient fall prevention based on gesture recognition and interactivity," in *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2008, pp. 3698–3701.

[129] J. Kimionis, A. Bletsas, and J. N. Sahalos, "Increased Range Bistatic Scatter Radio," *IEEE Transactions on Communications*, vol. 62, no. 3, pp. 1091–1104, 2014.

[130] Y. Dong, A. Wickramasinghe, H. Xue, S. Al-Sarawi, and D. C. Ranasinghe, "A Novel Hybrid Powered RFID Sensor Tag," in *IEEE International Conference on RFID*, 2015, pp. 55–62.

[131] R. Sankarkumar, D. C. Ranasinghe, and T. Sathyan, "A highly accurate and scalable approach for addressing location uncertainty in asset tracking applications," in *IEEE International Conference on RFID*, 2014, pp. 39–46.

[132] S. Hariharan and S. T. S. Bukkapatnam, "Misplaced item search in a warehouse using an RFID-based Partially Observable Markov Decision Process (POMDP) model," in *IEEE International Conference on Automation Science and Engineering*, 2009, pp. 443–448.

[133] A. Bordes, N. Usunier, and L. Bottou, "Sequence Labelling SVMs Trained in One Pass," in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, no. 5211, pp. 146–161.

[134] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research*, pp. 1453–1484, 2005.

[135] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[136] M. Kubat, R. Holte, and S. Matwin, "Learning when negative examples abound," in *European Conference on Machine Learning*, 1997, pp. 146–153.

[137] R. Yao, Q. Shi, C. Shen, Y. Zhang, and A. van den Hengel, "Part-Based Visual Tracking with Online Latent Structural Learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2363–2370.

[138] R. Tideiksaar, C. F. Feiner, and J. Maby, "Falls prevention: the efficacy of a bed alarm system in an acute-care setting." *The Mount Sinai Journal of Medicine, New York*, vol. 60, no. 6, pp. 522–527, 1993.

[139] A. Nawaz, J. Helbostad, N. Skjæret, B. Vereijken, A. Bourke, Y. Dahl, and S. Mellone, "Designing smart home technology for fall prevention in older

people," in *HCI International - Posters' Extended Abstracts*, ser. Communications in Computer and Information Science. Springer International Publishing, 2014, vol. 435, pp. 485–490.

[140] S. N. Robinovitch, F. Feldman, Y. Yang, R. Schonnop, P. M. Lueng, T. Sarraf, J. Sims-Gould, and M. Loughin, "Video capture of the circumstances of falls in elderly people residing in long-term care: an observational study," *The Lancet*, vol. 381, no. 9860, pp. 47–54, 2012.

[141] A. Bourke, P. van de Ven, M. Gamble, R. O'Connor, K. Murphy, E. Bogan, E. McQuade, P. Finucane, G. ÓLaighin, and J. Nelson, "Evaluation of waist-mounted tri-axial accelerometer based fall-detection algorithms during scripted and continuous unscripted activities," *Journal of Biomechanics*, vol. 43, pp. 3051–3057, 2010.

[142] Q. Li, J. Stankovic, M. Hanson, A. Barth, J. Lach, and G. Zhou, "Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information," in *Sixth International Workshop on Wearable and Implantable Body Sensor Networks.*, 2009, pp. 138–143.

[143] Y. He, Y. Li, and S.-D. Bao, "Fall detection by built-in tri-accelerometer of smartphone," in *IEEE-EMBS International Conference on Biomedical and Health Informatics*, 2012, pp. 184–187.

[144] H. Qian, Y. Mao, W. Xiang, and Z. Wang, "Home environment fall detection system based on a cascaded multi-SVM classifier," in *International Conference on Control, Automation, Robotics and Vision*, 2008, pp. 1567–1572.

[145] M. Lan, A. Nahapetian, A. Vahdatpour, L. Au, W. Kaiser, and M. Sarrafzadeh, "Smartfall: an automatic fall detection system based on subsequence matching for the smartcane," in *International Conference on Body Area Networks*, 2009, pp. 8:1–8:8.

[146] L. Klack, C. Möllering, M. Ziefle, and T. Schmitz-Rode, "Future care floor: A sensitive floor for movement monitoring and fall detection in home environments," in *Wireless Mobile Communication and Healthcare*, ser. LNICST. Springer, 2011, vol. 55, pp. 211–218.

[147] A. Braun, H. Heggen, and R. Wichert, "CapFloor – A Flexible Capacitive Indoor Localization System," in *Evaluating AAL Systems Through Competitive Benchmarking. Indoor Localization and Tracking*, ser. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2011, pp. 26–35.

[148] M. A. Aud, C. C. Abbott, H. W. Tyrer, R. V. Neelgund, U. G. Shriniwar, A. Mohammed, and K. K. Devarakonda, "Smart Carpet: Developing a sensor system to detect falls and summon assistance," *Journal of Gerontological Nursing*, vol. 36, no. 7, pp. 8–12, 2010.

[149] F. Werner, J. Diermaier, S. Schmid, and P. Panek, "Fall detection with distributed floor-mounted accelerometers: An overview of the development and evaluation of a fall detection system within the project eHome," in *International Conference on Pervasive Computing Technologies for Healthcare*, 2011, pp. 354–361.

[150] M. Alwan, P. J. Rajendran, S. Kell, D. Mack, S. Dalal, M. Wolfe, and R. Felder, "A Smart and Passive Floor-Vibration Based Fall Detector for Elderly," in *Information and Communication Technologies*, vol. 1, 2006, pp. 1003–1007.

[151] Future-Shape GmbH, "Future Shape | Technology | SensFloor® – large-area sensor system." [Online]. Available: http://future-shape.com/en/technologies/23/sensfloor-large-area-sensor-system

[152] R. L. Shinmoto Torres, A. Wickramasinghe, V. N. Pham, and D. C. Ranasinghe, "What if your floor could tell someone you fell? a device free fall detection method," in *Artificial Intelligence in Medicine*. Springer, 2015, pp. 86–95.

[153] S. M. Donelson and C. C. Gordon, "1995 matched anthropometric database of us marine corps personnel: Summary statistics." GEO Centers INC, Tech. Rep., 1995. [Online]. Available: http://www.humanics-es.com/ADA316646.pdf

[154] M. Zhou and D. C. Ranasinghe, "A novel approach for addressing wandering off elderly using low cost passive rfid tags," in *Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Springer, 2013, pp. 330–343.

[155] L. Yang, Y. Qi, J. Fang, X. Ding, T. Liu, and M. Li, "Frogeye: Perception of the slightest tag motion," in *IEEE Conference on Computer Communications*, 2014, pp. 2670–2678.

[156] EPCglobal Inc, "Low Level Reader Protocol (LLRP)," Oct. 2010. [Online]. Available: http://www.gs1.org/gsmp/kc/epcglobal/llrp/llrp_1_1-standard-20101013.pdf

[157] H. Li, C. Ye, and A. P. Sample, "IDSense: A Human Object Interaction Detection System Based on Passive UHF RFID," in *Annual ACM Conference on Human Factors in Computing Systems*, 2015, pp. 2555–2564.

[158] A. Jayatilaka and D. C. Ranasinghe, "Real-time fluid intake gesture recognition based on batteryless UHF RFID technology," *Pervasive and Mobile Computing*, 2016.