PERSONALIZED FALL DETECTION MONITORING SYSTEM BASED ON USER MOVEMENTS

by

Pranesh Vallabh

Submitted in partial fulfilment of the requirements for the degree
Master of Engineering (Computer Engineering)

in the

Department of Electrical, Electronic and Computer Engineering
Faculty of Engineering, Built Environment and Information Technology

UNIVERSITY OF PRETORIA

April 2018

SUMMARY

PERSONALIZED FALL DETECTION MONITORING SYSTEM BASED ON USER MOVEMENTS

by

Pranesh Vallabh

Supervisor(s):	Prof. Reza Malekian
Department:	Electrical, Electronic and Computer Engineering
University:	University of Pretoria
Degree:	Master of Engineering (Computer Engineering)

Keywords:	Fall detection, Machine learning, Accelerometer, Wearable device, Feature extraction, Feature selection, Personalized systems

High accuracy, fall detection systems is a fundamental requirement among the increasing elderly population, mainly due to expensive healthcare and a shortage of nurses for home-care. Fall detection systems have evolved over the past few years, from a button pendant to three newer types of fall detection systems - wearable sensors, ambient sensors, and camera-based sensors. Wearable sensors are regarded as the most popular, as it provides both indoor and outdoor monitoring and is the least expensive among the newer fall detection systems. Detection of a fall, using wearable sensors, started off at first by using a threshold method, where the features extracted from the wearable sensor data are compared to a pre-defined value. The problem with this approach is that the pre-defined value only works on a small set of people with certain user characteristics. It was also difficult to set a value that can distinguish between everyday activities and fall activities. To solve this problem, supervised machine learning algorithms were incorporated - these obtained higher accuracies when compared to the threshold method. Supervised machine learning algorithms achieved high accuracy during laboratory experiments. In a practical scenarios, the performance of these fall detections were low, due to the supervised machine learning algorithms making use of simulated fall data which is performed on a soft mattress which does not represent a real fall event (which is usually spontaneous). Since it is

difficult to obtain real fall data, a lot of studies make use of simulation data. Using artificial fall as training data can result in over-fitting, which causes poor decisions. Both threshold and supervised classifiers cannot provide a user-specific solution for each individual user. Since supervised machine learning algorithms require everyday activities and fall activities to classify, as well as limited fall data (which creates an imbalance i.t.o classification), it is hard for these algorithms to classify accurately. Another problem is that these systems are limited to a certain number of activities that a user can perform, and it does not work for everybody. User-specific personalization can be provided using unsupervised machine learning algorithms, resulting in the following advantages: a) more activities can be included in the classifier, and b) the fall detection system can address the inter-individual differences. In this research, the effects of personalization models using user movements are analysed (in terms of accuracy). A low-cost smartphone accelerometer sensor was used in the system. The study was divided into two parts: a simulation phase and an experimental phase. The simulation phase made use of a public dataset known as the *tFall* dataset. The type of input data to be used, which machine learning algorithm to use and the different types of personalization models, were investigated. For the type of input data, the following were considered: raw accelerometer values, statistical features extracted from the accelerometer, principal component analysis on the statistical features extracted, or the statistical features selected from the variance-threshold feature selection method. Both supervised and unsupervised machine learning algorithms were implemented to determine the best algorithm. The following unsupervised machine learning algorithms were implemented: nearest neighbour, one-class support vector machine, angle based outlier detection, and isolation forest. Angle based outlier detection and isolation forest were not implemented in any fall detection systems before. For the supervised machine learning algorithm, the two most popular machine learning algorithms were selected: support vector machine learning algorithm and k-NN. The following models were tested: a) Model 1 − the classifier itself; b) Model 2 the non-fall activity is retrained whenever the classifier correctly detects a non-fall activity; c) Model 3 the false positive is retrained when the classifier detects a non-fall activity as a fall activity, and d) Model 4 combining Model 2 and Model 3. The unsupervised machine learning algorithm is applied to all the models, whereas supervised machine learning algorithm is only applied to Model 1. During the simulation phase, the following evaluation parameters were used: sensitivity, specificity, geometric mean and F_1-measure. During the experimental phase, the best input data set (raw accelerometer values), model (Model 4) and classifier (angle based outlier detection) were implemented on an Android smartphone, to demonstrate how accurately the fall detection can classify. From experimental results, it was shown that personalization models using user movements can improve the overall performance of the system, achieving a sensitivity of 90.48% and specificity of 92.31%.

LIST OF ABBREVIATIONS

ABOD	Angle Based Outlier Degree
ACC	Accuracy
ADL	Activities of Daily Living
AUC	Area Under the Curve
BMI	Body Mass Index
DFT	Discrete Fourier Transform
ECG	Electrocardiography
EMG	Electromyogram
FFT	Fast Fourier Transform
FIPS	Fall Injury Prevention System
FN	False Negative
FP	False Positive
GMM	Gaussian Mixture Model
IR	Infrared
ISF	Isolation Forest
k-NN	k-Nearest Neighbour
M1	Model 1
M2	Model 2
M3	Model 3
M4	Model 4
MEMS	Micro-Electro-Mechanical Systems
MFCC	Mel Frequency Cepstral Coefficents
NN	Nearest Neighbor
PCA	Principal Component Analysis
PERS	Personal Emergency Response System
PIR	Passive Infrared
RGB	Red-Green-Blue
ROC	Receiver Operating Characteristic
SC	Supervised Classifier
SE	Sensitivity

SMV	Signal Magnitude Vector
SP	Specificity
SVM	Support Vector Machine
TN	True Negative
TP	True Positive

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION **1**
1.1 PROBLEM STATEMENT 1
1.1.1 Context of the problem 1
1.1.2 Research gap 2
1.2 RESEARCH OBJECTIVE AND QUESTIONS 2
1.3 APPROACH 3
1.4 RESEARCH GOALS 4
1.5 RESEARCH CONTRIBUTION 4
1.6 RESEARCH OUTPUTS 4
1.7 DISSERTATION OVERVIEW 5

CHAPTER 2 LITERATURE STUDY **6**
2.1 CHAPTER OVERVIEW 6
2.2 THE NEED FOR A FALL DETECTION MONITORING SYSTEM 6
2.2.1 How and why do fall occur? 7
2.2.2 How does a fall affect a person 's life? 7
2.2.3 What is expected from a fall detection system? 8
2.3 MODEL OF A FALL DETECTION SYSTEM 9
2.3.1 Data collection 9
2.3.2 Feature Extraction 10
2.3.3 Feature selection 10
2.3.4 Classifiers 11
2.3.5 Testing and evaluation of the system 13
2.4 FALL DETECTION SENSORS 15
2.4.1 Wearable sensors 15

2.4.2 Ambient sensors . . . 23
2.4.3 Camera-based methods . . . 27
2.5 PROBLEM OF CURRENT FALL DETECTION SYSTEMS . . . 33
2.6 PERSONALIZATION . . . 35
2.6.1 Why the need for a personalization model? . . . 35
2.6.2 How to create a personalized model? . . . 36
2.6.3 Who did this so far? . . . 37
2.7 CHAPTER SUMMARY . . . 40

CHAPTER 3 METHODS . . . 42
3.1 CHAPTER OVERVIEW . . . 42
3.2 SENSORS . . . 42
3.3 SIMULATION . . . 42
3.3.1 Data collection . . . 43
3.3.2 Features extraction . . . 44
3.3.3 Features selection . . . 48
3.3.4 Dataset . . . 51
3.3.5 Classifiers . . . 51
3.3.6 Models . . . 59
3.3.7 Evaluation . . . 60
3.4 EXPERIMENTAL . . . 62
3.5 CHAPTER SUMMARY . . . 65

CHAPTER 4 RESULTS . . . 68
4.1 CHAPTER OVERVIEW . . . 68
4.2 SIMULATION . . . 68
4.2.1 Datasets . . . 68
4.2.2 Models . . . 78
4.3 EXPERIMENTAL . . . 84
4.4 CHAPTER SUMMARY . . . 85

CHAPTER 5 DISCUSSION . . . 87
5.1 CHAPTER OVERVIEW . . . 87
5.2 DATASET . . . 87

5.2.1 Datasets used by the supervised machine learning algorithms 88
5.2.2 Datasets used by the unsupervised machine learning algorithms 88
5.2.3 Datasets conclusion 89
5.3 MODELS 90
5.3.1 M1 vs SC 92
5.3.2 M2 vs M1 92
5.3.3 M3 vs M1 93
5.3.4 M4 vs M1 93
5.3.5 M2 vs SC 93
5.3.6 M3 vs M2 94
5.3.7 M4 vs M2 94
5.3.8 M3 vs SC 94
5.3.9 M4 vs M3 95
5.3.10 M4 vs SC 95
5.3.11 Models conclusion 95
5.4 CLASSIFIERS 95
5.4.1 Unsupervised machine learning algorithms 95
5.4.2 Supervised machine learning algorithms 97
5.4.3 Classifier conclusion 97
5.5 ACCURACY 98
5.6 ANDROID APPLICATION 99
5.7 CHAPTER SUMMARY 100

CHAPTER 6 CONCLUSION 101
6.1 CONCLUSION 101
6.2 BENEFITS OF THE STUDY 102
6.3 RECOMMENDATION FOR FUTURE WORK 103

REFERENCES 104

LIST OF FIGURES

2.1 Fall detection classifier model. 9
2.2 Fall detection sensor categories. 15
2.3 Operations of the camera system when performing fall detection. 28

3.1 The raw x (a), y (b) and z (c) axis of the accelerometer data for ADL. 46
3.2 The raw x (a), y (b) and z (c) axis of the accelerometer data for the different fall activities. 47
3.3 Cumulative explained variance vs number of components using dataset 2 features. . . 51
3.4 Example of the ABOD model calculation of the angle between two points. 54
3.5 One-class SVM. 57
3.6 Model 1 algorithm. 60
3.7 Model 2 algorithm. 61
3.8 Model 3 algorithm. 62
3.9 Model 4 algorithm. 63
3.10 The method used to train and evaluate SC and M1. 64
3.11 The method used to train and evaluate M2, M3, and M4. 64
3.12 Algorithm used on the Android application for detecting falls. 66

4.1 Geometric-mean for all the unsupervised machine learning algorithms for the four models using dataset 1. 69
4.2 The difference in geometric-mean between M4 and M1 for the unsupervised machine algorithms using dataset 1. 70
4.3 Geometric-mean for all the unsupervised machine learning algorithms for the four models using dataset 2. 70
4.4 The difference in geometric-mean between M4 and M1 for the unsupervised machine algorithms using dataset 2. 71

4.5 Geometric-mean for all the unsupervised machine learning algorithms for the four models using dataset 3. 72
4.6 The difference in geometric-mean between M4 and M1 for the unsupervised machine algorithms using dataset 3. 72
4.7 Geometric-mean for all the unsupervised machine learning algorithms for the four models using dataset 4. 73
4.8 The difference in geometric-mean between M4 and M1 for the unsupervised machine algorithms using dataset 4. 73
4.9 Geometric-mean for all the supervised machine learning algorithms using datasets 1 to 4. 75
4.10 Difference in geometric-mean between model M1 (dataset 4) and SC (dataset 4). . . 79
4.11 Difference in geometric-mean between model M2 (dataset 1) and model M1 (dataset 4). 80
4.12 Difference in geometric-mean between model M3 (dataset 1) and model M1 (dataset 4). 80
4.13 Difference in geometric-mean between model M4 (dataset 1) and model M1 (dataset 4). 81
4.14 Difference in geometric-mean between model M2 (dataset 1) and SC (dataset 4). . . 81
4.15 Difference in geometric-mean between model M3 (dataset 1) and model M2 (dataset 1). 82
4.16 Difference in geometric-mean between model M4 (dataset 1) and model M2 (dataset 1). 82
4.17 Difference in geometric-mean between model M3 (dataset 1) and model SC (dataset 4). 83
4.18 Difference in geometric-mean between model M4 (dataset 1) and model M3 (dataset 1). 83
4.19 Difference in geometric-mean between M4 (dataset 1) and SC (dataset 4). 84
4.20 The screenshots of the Android fall detection application. The start-up screenshot (a), and the screenshot (b) is when a fall has occurred. 86

LIST OF TABLES

2.1 Characterizing different fall detection systems. 15
2.2 Summary of wearable sensors studies 21
2.3 Personalized fall detection studies summary 40

3.1 Information of the tFall dataset 44
3.2 The number of records for each person in the tFall dataset 45
3.3 Statistical formulas used to extract features from the time and frequency domain . . . 49
3.4 Features selected using variance threshold. 52
3.5 Features selected using variance threshold continued from Table 3.4. 53
3.6 Datasets used in this study 54
3.7 One-class SVM parameters 58
3.8 SVM parameters 59
3.9 ADL and fall activities movements list. 67

4.1 The best classifier for each dataset using M1 74
4.2 The best classifier for each dataset using M2 74
4.3 The best classifier for each dataset using M3 75
4.4 The best classifier for each dataset using M4 75
4.5 Values of SE, SP, geometric-mean, F_1-measure for ADL and fall activities for each person using SVM with dataset 4. The last row is the average value over subjects. . 76
4.6 Values of SE, SP, geometric-mean, F_1-measure for ADL and fall activities for each person using M1 with ABOD classifier and dataset 4. The last row is the average value over subjects. 77
4.7 Values of SE, SP, geometric-mean, F_1-measure for ADL and fall activities for each person using model M2 with NN classifier with dataset 1. The last row is the average value over subjects. 77

4.8 Values of SE, SP, geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities for each person using model M3 with ABOD classifier with dataset 1. The last row is the average value over subjects. 78
4.9 Values of SE, SP, geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities for each person using M4 with ABOD classifier with dataset 1. The last row is the average value over subjects. 79
4.10 Summary of the comparison with respect to SE, SP, geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities. For each pair of comparison, a cell contains the number of subjects for whom the member of the pair is better for SE, SP, geometric-mean or their F_1-measure. 85
4.11 Values of SE, SP, geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities for each subject. The last row is the average value over subjects. 86

CHAPTER 1 INTRODUCTION

1.1 PROBLEM STATEMENT

1.1.1 Context of the problem

The current research in fall detection is limited to a few activities which restricts a user's movements to certain activities. Fall detection can recognize simple activities such as walking, running, standing and sitting but does not address extreme activities such as housework and outdoor activities. False alarms are not addressed by the fall detection system which can lead to frustration. This frustration can occur when the normal activity being performed by the user is treated as a fall event because that normal activity is not learned by the system. Most of these systems are based on the experimental setup and not reality and therefore achieve a low accuracy. Current fall detection systems do not take into account the different characteristics of a person such as height and weight, which influence the performance of the system. Most studies only explored threshold and supervised machine learning methods and not novelty or outlier detection methods. The problem with threshold methods is that the threshold value does not work on different user characteristics. Supervised learning algorithms make use of label data for training the system and the output of the system is controlled [1], [2]. Certain classifiers can perform better on certain activities compared to other activities [2]. Classifiers such as voting machines or comparator machines can be combined [3]. A hybrid framework which makes use of both threshold-based and machine learning algorithms is implemented in this study [4]. Popular supervised machine learning algorithms include Naive Bayes, k-Nearest neighbour (k-NN), support vector machine (SVM), hidden Markov model, and artificial neural network. The problem with supervised methods is that they train with experimental fall activities which do not represent a true fall event. Unsupervised learning algorithms make use of unlabeled data for training the system [1]. This type of learning algorithm can only be trained on fall data or non-fall data [5]. The classifier can be trained with new activities on the fly. Popular unsupervised classifiers include one-class support vector machine, nearest-neighbour (NN), and Gaussian Mixture Model (GMM). This research investigates the question of adapting the outlier detection classifier method for user movements in order to provide a high accuracy fall detection system.

1.1.2 Research gap

A lot of research has been done in fall detection. The current fall detection system is limited to a set of activities of daily living (ADL) that the user can perform, and does not allow for new activities to be added to the system. The biggest problem is the false positives are not addressed in fall detection systems. Most studies make use of supervised classifier which achieves high accuracy in a laboratory and not in the real world. This is because the fall data used does not represent the actual fall event in real life. The current fall detection system does not work for everybody since everyone has unique characteristics and there is no system which learns to adapt to or personalize user movements. The proposed system addresses the problems mentioned above by implementing a personalized model that can include new activities and reduce false positives.

1.2 RESEARCH OBJECTIVE AND QUESTIONS

The proposed system addresses the problems of current fall detections system that does not work for everybody since everyone has unique characteristics and there is no system which learns to adapt to or personalize user movements.

The objective of the research is thus

(a) to design a personalized or adaptive system which
(b) only needs ADL data for training to design a more accurate classification model, which
(c) allows inclusion of new activities and handling of false positives.

This will overcome the limitations of not using real fall data for training and limited set of ADLs that the user can perform.

The research questions are as follows:

- How can personalized data be relevant to fall detection monitoring system to reduce false positives for fall events?
- What technology can be used to track user movements?
- What machine learning and feature extraction algorithms provide the best accuracy?
- Will accuracy of the system improve with the incorporation of user's information?
- What is the efficiency of training a system based on user movements?
- How does a supervised model compare to an unsupervised model?

1.3 APPROACH

In order to meet the objectives of personalized fall detection system, the approach highlighted below needs to be followed:

1. Investigate current implementations of personalized fall detection systems and analyze which aspects should be considered regarding the monitoring system.
2. Investigate and implement the technology used to track user movements and detect falls.
3. Investigate and implement different feature extraction methods on the sensor data, and perform feature selection on the extracted features.
4. Investigate and implement different personalized models.
5. Investigate and implement different supervised and unsupervised machine learning algorithms.
6. Implement the best features, best-personalized models and the best machine learning algorithms on an Android smartphone device.
7. Tests will be performed on different subjects to evaluate the accuracy of the fall detection Android application by performing ADL and fall activities.
8. The obtained results will be analyzed to evaluate the effects of personalized fall detection on the user and determine whether fall detection decreases false positives and improves system reliability. The results obtained from the solution will be recorded and compared to the proposed solution.

The following approach numbers 3 - 5 will be simulated using a public dataset, to determine the best approach and how these parameters influence the overall performance of the fall detection system.

1.4 RESEARCH GOALS

The research goal is to create a personalized system that will allow the addition of new activities to the system by investigating and implementing different personalized models and machine learning algorithm. The false alarms will be addressed by allowing the user to add the false alarm activity to the system. The personalized system will not force a user to perform certain activities, but rather learn the user's movements. It will be a system that can be customized to the user characteristics, which can lead to an improved performance.

1.5 RESEARCH CONTRIBUTION

The contributions will include features extraction for the machine learning algorithm and evaluation of different types of unsupervised machine learning algorithm that can allow the creation of a personalized model. A comparison between supervised and unsupervised machine learning algorithms will be done. The study will develop a real-time personalized fall detection model with the capabilities of learning user movements from a sensor-based technology and addressing the problem of false alarms. A fall detection Android system will be implemented.

1.6 RESEARCH OUTPUTS

The following papers have been published over the course of this research:

1. P. Vallabh, R. Malekian, N. Ye and D. Bogatinoska, "Fall detection using machine learning algorithms", *2016 24th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2016.

2. P. Vallabh, R. Malekian, N. Ye, D.C. Bogatinoska, A. Karadimce, J. Ritonja, “Classification of fall detection in elderly persons based on smartphone data”, *Journal of Biotechnology*, vol. 231, pp. s29-s30, Aug. 2016.
3. P. Vallabh, R. Malekian, “Fall detection monitoring systems: A comprehensive review”, *Journal of Ambient Intelligence and Humanized Computing*. (Accepted)

1.7 DISSERTATION OVERVIEW

In Chapter 2 a comprehensive literature survey of the different fall detection systems, limitations of current fall detection systems, and the need for a personalized system is provided. In Chapter 3 personalized adaptive models that were developed, which includes implementation of the different classifiers and the different features extracted, is discussed. In Chapter 4, the obtained results of the developed systems are provided. Various classifiers are tested and compared. Adaptive models are investigated and tested. In Chapter 5, an in-depth analysis of the obtained results is provided. In Chapter 6 some concluding remarks are provided, followed by the benefits of the study, and lastly, some recommendations for future work are also presented.

CHAPTER 2 LITERATURE STUDY

2.1 CHAPTER OVERVIEW

In this chapter, a comprehensive literature study on the different fall detection studies is presented. In Section 2.2, the need for a fall detection monitoring system is discussed and analysed in terms of how do falls occur and the effect it has on a person's life; and what is expected from a fall detection system. In Section 2.3, the existing or most common fall detection classifier model is discussed. In Section 2.4, a detailed analysis on the different fall detection sensors is covered which includes wearable sensors, ambient sensors and camera-based methods. In Section 2.5, analyse of the current fall detection systems problems. In Section 2.6, personalization fall detection systems are investigated and analysed in terms of the need for fall detection systems and discuss studies which implemented personalized fall detection systems.

2.2 THE NEED FOR A FALL DETECTION MONITORING SYSTEM

One-third of the elderly population aged 65 years or more fall at least once each year, whereas half of the elderly population older than 80 years fall each year [6], [7], [8]. The increase in elderly population, notably in developed countries, and the number of elderly people living alone can result in increased healthcare costs which can cause a huge burden on the society and individuals [7], [9], [10], [11]. Due to the shortage of nursing homes, more elderly people are required to stay at home [12]. Elderly people who live alone cannot alert anyone for help when a fall occurs if they sustain serious injuries or if they become unconscious [13]. The World Health Organization reported that about 28% of people aged 65 fall and about 32% of people aged 70 fall each year [13], [14]. An automatic fall device that can successfully discriminate between ADL and fall activities is required. ADLs are a “wide set of actions characterizing the habits of people, especially in their living places e.g. walking, sitting, standing, and etc.". [15]. An ADL can also provide vital information about the health status of the user [15].

2.2.1 How and why do fall occur?

A fall is not considered a normal activity, but it occurs rarely [5]. Falls can be grouped into different categories which include forward fall, backward fall, vertical fall, sideways fall, and fall on buttocks [16]. Fall is defined as, "an event which results in a person coming to rest unintentionally on the ground or other lower level, not as a result of a major intrinsic event (such as a stroke) or overwhelming hazard" [4]. Falls can be identified by the large acceleration velocity the body produces which is greater compared to the normal posture or, patterns from an ADL can be monitored and if a change of pattern is detected, there is a chance of risk of a fall [6], [17], [18]. The risk of falls is divided into two categories, i.e. extrinsic and intrinsic risks [6], [19]. The extrinsic risk is related to the environmental factors such as drug usage, slippery floors, poor lighting, loose carpets, unstable furniture, clutter and obstructed paths; whereas the intrinsic risk is related to the characteristics of the person such as age, general clinical condition and mental impairment [6], [19], [20]. Extrinsic factors can be prevented by taking precautions, whereas intrinsic factors cannot be prevented [19]. Falls can be caused by loss of balance, environmental factors, deterioration of body functions, visual impairment, and deterioration of muscle strength [5], [16], [17], [21]. Factors that contribute to the increase in the rate of falls are; an increase in person's age, mortality, morbidity, disability and frailty [5], [22]. A fall can occur in one second, usually taking between 0.45s and 0.85s, where an individual's posture and shape changes [21]. These changes are of great importance when detecting a fall [21].

2.2.2 How does a fall affect a person 's life?

The risk of a fall occurring is significantly high when an elderly person is living independently, and in 3% of falls that occur, the elderly people were found to be helpless or dead at home [23], [24]. Falls can result in serious injuries such as bone fractures, internal tissue damage, small contusions, immobilization, strokes and head trauma or even death [3], [13], [15], [25], [26], [27]. The quality of life deteriorates because of the psychological depression and stress that occurs due to a fall [5], [6], [13]. Falls can also result in fear of falling (FOF), loss of independence, no social contract, lack of movement, and decrease in productivity, which increases the risk of another possible fall [4], [5], [6], [14], [28]. This can result in loss of self-confidence which can lead to social isolation and lower quality of life [14]. FOF is linked to an increase of neuroticism and anxiety, which results in elderly people avoiding participation in any physical activity [14]. The biggest danger of falling is a "long-lie" condition, where the fall victim is unable to stand up from a fall, and remains on the ground for a couple of hours [27]. Long lie can result in dehydration, internal bleeding, physiological and psychological consequences; and "half of the people who experience long lie die within 6 months" [6], [7], [9], [18], [29], [30]. There is a 50% chance that if a fall victim was left on the ground for 1 hour he/she will die within six months after the fall [7], [27], [28].

2.2.3 What is expected from a fall detection system?

Real-time activity recognition can be used to provide monitoring, assistance and analysis of subjects' activities [5], [13]. Fall detection devices that are available on the market are not satisfactory, in terms of high false alarms, high maintenance cost, and they are not ergonomic [19]. Falling does not only cause serious damage to human bones, but it can also lead to death and therefore, a fast, low-cost and accurate fall detection system is required to provide immediate attention from medical personnel to prevent any further damage to the human body [9], [13], [16], [27], [31]. A precise, robust and reliable fall detection system is required for elderly people living independently thus reducing the risks of living alone [11], [23], [32]. Fall detection systems need to respond instantly in order to reduce impact and recovery time, inform others quickly to reduce the time a fall victim remains on the floor and to avoid any injuries that can occur [4], [11], [32]. The problem with the current systems is that they are limited in terms of accuracy, usability, and cost [7]. The ideal fall detection system should not produce false fall alarms [11], [33] and should not force elderly people to change their lifestyle [8]. A basic fall detection system is a device with a button, known as an user-activated device, which is usually worn on the wrist or as a necklace. It requires a user to be conscious when a fall occurs and to press the button to alert emergency personnel [6], [11], [12]. The problem with the push button is that it cannot be pressed if the fall victim has lost consciousness, is confused, or is panicking. The button can also be accidentally pushed or may not be worn by the falling victim [6], [10], [30]. There is no standard method for fall detection in terms of the type of sensors to use, features to extract, and what machine learning algorithm performs better [5]. A fall detection system should have the following characteristics: no intrusion on the user's privacy, no restriction on the user's independence, and should not degrade the user's quality of life [19]. Quick detection and notification that a fall has occurred can decrease the risk of hospitalization by 26% and death by 80% [23].

2.3 MODEL OF A FALL DETECTION SYSTEM

Figure 2.1 shows the most common fall detection model which is used in many studies when designing a fall detection system. The model comprises of the following parts which will be discussed below: data collection, feature extraction, feature selection, classifier and evaluation.

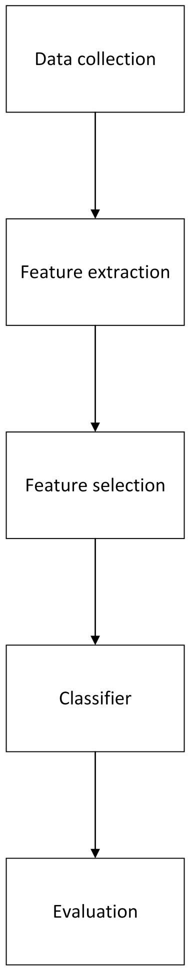


Figure 2.1. Fall detection classifier model.

2.3.1 Data collection

Fall detection starts by collecting data from sensors. These can be either wearable sensors or ambience sensors and/or camera-based sensor systems. These sensors will be discussed in more detail in Section 2.4.

2.3.2 Feature Extraction

Feature extraction is a method whereby significant attributes are found from raw data which consists of meaningless information which plays a vital part in determining the accuracy of the fall detection system [1], [2], [21]. Fall detection systems require distinctive features to represent the different activities and need to be able to classify falls from ADLs [34]. There are different features, each having relevant characteristics to specific ADLs or fall activity being performed [2]. Features can be grouped into two categories; time or frequency based features [2]. In wearable devices, the most popular features are acceleration magnitude of the accelerometer and angular magnitude of the gyroscope [1]. In camera-based systems, the aspect ratio is the most common feature; whereas in Doppler, a vibration device and acoustic device the Mel frequency spectral coefficients (MFCC) features are the most popular. A lot of features are calculated using statistical models such as median, max, min, variance, etc. [2]. Features should be carefully selected in order to produce a small descriptive dataset [1]. The dataset's descriptive power is impacted by the number of features that the dataset is comprised of [1]. Extracting features are performed on data using a sliding window method [2].

2.3.3 Feature selection

The more features a database has, the more descriptive it becomes and it becomes difficult to find meaningful relationships among the class as feature space grows exponentially. The performance of the machine learning algorithm is also dependent on the feature space [1], [2], [21], [23]. By finding features which describe the data better and discarding the redundant features, computational speed and prediction accuracy can be improved [1], [23]. The problem of selecting features from a N dimensional feature space is known as feature selection method [30]. The feature selection algorithms are used to detect and discard features that provide a minimum contribution to the performance of the classifier [2]. Feature selection provides the following advantages; it reduces the cost of pattern detection and the dataset and provides better performance [23], [30]. There are two categories of feature selection methods, i.e. filter methods and wrapper methods. Filter methods or ranking methods make use of search algorithms to score the different features and rank the features from the best to the worst [1], [2]. Filter methods make use of statistical tests such as T-test, F-test, Chi-squared, etc. The wrapper method takes a combination of different features and compares the combination of features based on the classifier results, where the classifier is part of the selection process [1], [2]. The combination of features is chosen based on the one which provides an accurate model for classification [1], [2].

The disadvantage of the wrapper method is that it requires a huge amount of processing power and it is very time-consuming [2]. Instead of selecting features, all the features that are extracted are combined together to create new features using principal component analysis (PCA). The PCA is an unsupervised linear transformation method, for identifying patterns in high dimension data and expressing the data in a manner that highlights their similarities and differences [13], [15]. A PCA algorithm provides an orthogonal transformation of a large feature space into a new set of values made of linearly uncorrelated variables called principal components, which result in significant small feature space and decrease in dimensionality [13], [15].

2.3.4 Classifiers

The fall detection classifiers can be divided into two groups, i.e. threshold-based or rule-based and machine learning algorithms [3], [4], [14].

2.3.4.1 Threshold or rule-based

The most popular classification method used in fall detection studies is the threshold analytical method [5], [17]. The basic principle for the threshold analytical method is that a possible fall could be detected based on the value read from the sensors and compared to the reference value [5], [17]. Fall detections that make use of accelerometer sensors, use a threshold parameter to detect falls and compare it to a pre-defined value [13]. The pre-defined value is determined by a fall signal [3], [13]. Fall training data is required to compute the threshold value using domain knowledge or data analysis techniques [5]. The advantage of the threshold method is that it is easy to implement, has a low power budget and uses less computational power [14], [15], [17]. The problems with threshold systems include limited recognition ability, low precision, difficultly in determining the pre-defined value which results in high false rates from running or jumping, resulting in low accuracy [13], [17], [20]. The performance of fall detection methods is affected by the selection of the fall indicators and detection thresholds [35]. Thresholds result in low accuracy which makes researchers focus more on machine learning classifiers which achieve higher accuracy.

2.3.4.2 Machine learning

Machine learning algorithms achieved greater performance compared to threshold classifiers when using an accelerometer sensor [3]. Machine learning algorithms have complex implementation compared to the threshold implementation; the decision is based on posture calculation which results in a higher fall detection rate [17]. Advantages of machine learning algorithms include: different falls can be customized, high accuracy compared to the threshold methods, it can manage anomalies such as noise and incompleteness well and detection of patterns in signals is possible [4], [17]. Disadvantages of machine learning algorithms include: requiring a large amount of representative training data which is complex and requires heavy processing [4], [17], [33]. Machine learning algorithms can be grouped into two categories; i.e. supervised and unsupervised learning algorithm.

2.3.4.2.1 Supervised

Discrete Fourier Transform (DFT) was performed on all the axes on the gyroscope and accelerometers which resulted in a six output array of DFT [36]. A principal component analysis was used to select the best 10 coefficients to form each six output arrays [36]. An SVM classifier was used to detect falls [36]. In [29] acoustic sensors that share similar operating characteristics to a stethoscope is used to obtain the audio signal through the floor. Features that are extracted and fed into SVM include the MFCC which represent low-level acoustic features and the Gaussian means supervectors [29]. In [37] the audio segments from a single far-field microphone is model using GMM supervector [37]. The GMM supervector is feed into an SVM which was used to differentiate falls from other noises [37]. In [38] a system make use of a Doppler sensor where the MFCC features are extracted and fed into a SVM classifier [38]. In [24], characterizing the fall using depth images is done using Curvature Scale Space features with Fisher Vector encoding [24]. The Fisher Vector encoding is fed into the SVM classifier [24].

In [39] an eight circular microphone array is employed to detect a possible fall, where MFCC features are extracted and are fed into a k-NN where k is equal to 1 [39]. The system detects falls based on the tri-axial accelerated velocities and angular accelerated velocities, to generate a real-time human posture and falls [17]. A Bayesian classifier will be used to predict the user next action [17]. The following features are extracted: shock response spectrum and MFCC from a floor vibration sensor and microphone which is fed into a Bayes classifier which was used to differentiate between falls and non-fall activities [30]. In [40], a hidden Markov model was implemented using signals obtained from passive infrared sensors. The system captures the acceleration from the sensor, where the wavelet transform is performed on the sensor [3]. The system makes use of the following classifiers: artificial neural network, k-NN, radial basis function, probabilistic principal components and linear discriminant analysis (LDA) [3]. From all the classifiers tested, the k-NN achieved the highest accuracy.

2.3.4.2.2 Unsupervised

In [41] a system makes use of a tri-axial accelerometer to capture the accelerometer data. Fall data is used to train the classifier where the outliers of the classifiers are ADLs. The following features are extracted: the sum of the x-axis and z-axis, and the total sum of tri-axis was feed into a one-class SVM classifier [41]. The MFCC features are extracted from the processed audio signal and feed into a one-class SVM classifier, where normal activities are trained [9]. In [42] the following features were extracted from a camera and feed into a one-class SVM classifier: ellipse features, shape-structure features, and position features [42]. In [43] unsupervised classifiers make use of non-fall sounds recorded from a single microphone. The noise was removed using a Wiener filter and MFCC features were extracted [43]. The following classifiers were tested: neighbour classifier, one-class SVM, and Gaussian classifier was used [43]. The one-class SVM outperforms other classifiers [43].

2.3.5 Testing and evaluation of the system

The system is typically tested by performing the leave-one-out method or cross-validation method [4]. The dataset can also be split into 70% for training the classifier and 30% for testing the classifier [2]. Statistical tests are done to determine the overall performance of the classifier [1]. The classification model can produce the following four possible outcomes [3]: 1. True Positive (TP) when a system properly detects a fall when a fall has occurred. 2. False Positive (FP) when a system detects a fall when no fall has occurred. 3. True Negative (TN) when a system detects no fall when no fall has occurred. 4. False Negative (FN) when a system detects no fall when a fall has occurred. False negatives are falls which remain undetected and false positives are ADL activities which are classified as falls [14]. Recall (or sensitivity), precision, specificity, accuracy and F_1-measure are the most popular methods for measuring the performance of classifiers.

Recall or sensitivity (SE) calculates how well a fall detection system can correctly detect falls over the whole set of fall instances [1], [3], [28], as follows:

$$SE = \frac{TP}{TP+FN} \tag{2.1}$$

Precision calculates how well a fall detection system can correctly detect falls to the whole set of instances detected as falls. The precision measures the ability of the classifier to return the fall results that were correctly detected [1], [3], as follows:

$$precision = \frac{TP}{TP+FP} \tag{2.2}$$

Specificity (SP) calculates how well a fall detection system can correctly detect ADL over the whole set of ADL instances [1], [3], [27], [28], as follows:

$$SP = \frac{TN}{FP+TN} \tag{2.3}$$

Accuracy (ACC) is measured as the portion of fall results that were correctly classified amongst all outcomes [1], [3], as follows:

$$ACC = \frac{TP+TN}{TN+TP+FP+FN} \tag{2.4}$$

F_1-measure combines the precision and sensitivity indicators [1], [29], as follows:

$$F_1 - measure = \frac{2 \times precision \times recall}{precision + recall} \tag{2.5}$$

The receiver operating characteristic (ROC) theory can be used to define threshold values based on constraints on the system sensitivity and specificity [15]. By adjusting the threshold value, the ROC curve is created [18]. Equation (2.6) is used to find the threshold point from the ROC curve where the threshold point is the maximum geometric mean of the sensitivity and specificity [18].

$$geometric\text{-}mean = \sqrt{specificity \times sensitivity} \tag{2.6}$$

The area under the curve (AUC) is the ROC curve and expresses the performance of the classification model [38], [44]. The closer the AUC is to 1, the better the performance of the classification model.

2.4 FALL DETECTION SENSORS

Fall detection systems are also known as context-awareness systems and should be able to recognize, interpret, and monitor different activities the user performs and detect fall events [19]. There are different types of fall detection methods which include camera-based, acoustic-based and wearable sensors, shown in Figure 2.2 [28]. Each method of fall detection consists of numerous sensors but none of these sensors provides 100% accuracy however each sensor has its own advantage(s) and disadvantage(s) [45]. Table 2.1, shows the general characteristics of these sensor types.

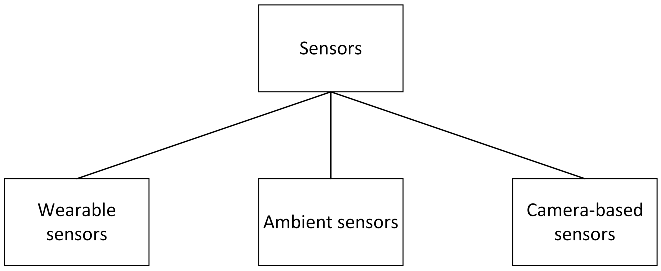


Figure 2.2. Fall detection sensor categories.

Table 2.1. Characterizing different fall detection systems.

Method	Price	Continuous monitoring	Battery problems	Obtrusive	Privacy	Monitored multiple people	Influence by the environment
Wearable	Cheap	Yes	Yes	Yes	Yes	No	No
Ambient	Medium pricing	No	No	No	Yes	No	Yes
Camera	Expensive	No	No	No	No	Yes	Yes

2.4.1 Wearable sensors

Due to the increase in wearable telemedicine technology, solving these problems becomes easier [22]. The growth of Micro-Electro-Mechanical System (MEMS) resulted in miniaturized, more compact and cheap sensors [19], [31]. This enabled the development of small, lightweight, portable devices which can be worn without interfering with physical activity [31]. Wearable sensors are worn by the subject of interest [21].

Wearable devices make use of embedded sensors to calculate the motion of the monitored body in any unsupervised environment, periods of inactivity, and the posture of the person [10], [14], [21]. Wearable sensors include the following inertial measurement units: accelerometer, gyroscope, and horizon finder [8], [13], [17]. A popular wearable device is a tri-axial accelerometer (detecting changes of body dynamics due to the fall) with a threshold decision tree classifier to detect a fall [8], [46], [47]. The first automatic fall detection system is a wearable device that is placed on the user to detect falls which make use of acceleration or rotation information [48]. Wearable sensors can detect a fall by analysing the impact of the body with the ground and taking the body orientation post before a fall occurs [49]. The sampling frequency of capturing data can range from 50 Hz to 250 Hz [27]. Collecting activity data from wearable sensors is not restricted to the laboratory environment, which allows collection of real-world activities [27]. Wearable devices can be implemented using a micro-controller or a smartphone.

2.4.1.1 Using smartphone for activity monitoring

Smartphones are now equipped with MEMS sensors which can be used for monitoring possible falls which make them suitable to use as they are part of daily life [5], [14], [15], [31]. Smartphones nowadays consist of various high precision sensors such as accelerometer, cameras, gyroscope, and proximity sensor [14], [16], [49]. The biggest advantage of the smartphone is that it has most of these sensors are integrated into it and it does not require extra devices [13], [15]. The biggest problem with smartphone devices used in fall detection is that the devices consume a lot of battery power; and have limitations in memory and real-time processing capabilities [13], [14].

2.4.1.2 Different types of wearable categories

Wearable fall monitoring systems are grouped into three categories: fall alert, fall risk assessment and impact prevention [4]. Fall alert or PERS (personal emergency response system) alerts medical personnel or caregivers to assist a user in a fall event [4], [28]. Fall risk assessment is the study of falls, particularly the causes and detection of patients who should be monitored based on their movements [4]. Impact prevention or FIPS (fall injury prevention system) detects a fall event before it happens and triggers a protection or prevention device to protect a user [4], [28]. An example of FIPS is detecting falls in the pre-impact phase where protection device can be activated, e.g. an inflatable airbag or similar projection device which is used to avoid any injuries from a fall [35], [47]. PERS prevents a long-lie by notifying caregivers when a fall is detected since some falls are too hard for the user to get up or the user may be unconscious [28]. The most popular type of system is PERS and more research is being conducted on it. PERS can be split into posture and motion device [49]. Only PERS system will be analysed because FIPS relies on pre-fall data to detect a possible fall. Pre-fall is shown to achieve a low accuracy [4]. Examples of FIPS systems are provided in [36], [50].

2.4.1.3 Different wearable sensors

Wearable sensors include, but are not limited to, tilt switches, accelerometers, gyroscopes, pressure sensors, magnetometers, and microphones [4]. Each sensor has different characteristics and the sensors can operate independently or combined.

2.4.1.3.1 Accelerometer

Accelerometer sensors are the most popular and widely used for detecting fall accidents and to sense body motions. It can achieve high accuracy in noisy environments as well as read acceleration measurements down to 0 Hz [21], [28], [31], [45], [51]. Accelerometers are feasible, effective, fast, easy to set up and operate, simple, lightweight, low-power, and cost-effective solutions for fall detection systems [3], [11], [52]. In [4], [18] and [52], it was reported that several studies achieved high accuracy using only accelerometers. In [53] a study was conducted to detect the type of wearable sensor that can accurately detect falls based on sensors that use acceleration, acceleration integrated with another method, and no acceleration sensors. The study concluded that using sensors which can sense acceleration is good at detecting falls; whereas methods that did not use acceleration are less accurate and can lead to many false alarms [53]. Falls can be detected by applying different signal evaluation techniques on accelerometer data [13]. The most popular feature extracted from the accelerometers is the Signal Magnitude Vector (SMV) which is given below:

$$SMV = \sqrt{\ddot{x}^2 + \ddot{y}^2 + \ddot{z}^2}, \tag{2.7}$$

where $\ddot{x}$, $\ddot{y}$, and $\ddot{z}$ are the acceleration values along the x, y and z axis of the accelerometer [53], [54]. A fall acceleration signal comprises of peaks and valleys, and fall activities usually associated with large SMV peaks [45], [49]. Fall decision which makes use of only SMV and considers only the abrupt peaks in the acceleration will result in high false positives, due to the sudden movements which occur when performing complex movements, for example, sitting down fast [14], [49]. Most fall detection systems use impact of the accelerometer to detect if a fall has occurred, which is usually not the case when a fall has occurred [7]. Most acceleration-based studies use a threshold-based algorithm for detecting a fall which results in high false alarms. In order to reduce false alarms, machine learning algorithms can be implemented [31]. Some studies transmit the raw data from the wearable device to a local computer for processing [3], [55], [56].

The placement of sensors also plays a vital role as it can directly impact the accuracy of fall detection techniques [53]. In [57] different positions on the human body were tested to identify the best place for the accelerometer. The following positions were tested: head, waist, and wrist to detect falls [57]. The results showed that placement of the accelerometer sensor on the person's head and waist achieves an SE of 97-98% and SP of 100% when using a simple threshold algorithm [57]. Investigation in [4], determined what phase should the fall be detected in and placement of the tri-axial accelerometer on the body which will achieve the best accuracy. The hybrid framework, which makes use of rule-based knowledge and a two-layer Gaussian classifier was implemented in [4]. The following accuracies were obtained at different phases of a fall: 86.54% for pre-impact, 87.315% for impact, and 91.15% for post-impact [4]. The paper found that the side of the waist is the best position for the sensor during post-impact, followed by the head, wrist, and front of waist, thigh, chest, ankle, thigh, and upper arm [4]. The reasons for not achieving 100% accuracy in the post-impact phase include signal loss and post-impact and high impact ADLs were classified incorrectly [4]. The following sensors' placements result in false positives by failing to differentiate falls between sitting and standing: head, upper arm, wrist, ankle, and chest [4]. Placing the sensor close to the person's centre of gravity makes the sensor less sensitive to spurious movements.

The disadvantage of an accelerometer is that the output of the accelerometer does not only consist of acceleration but also gravity, which can create errors when calculating the angles which result in high false positives [22]. Accelerometer methods require high sampling rate which can result in fast battery draining [8]. There are a lot of assumptions from smartphone detections that hardware sensors measure acceleration with sufficient precision but that is not the case [58]. The sensors from different manufacturers record values in significantly different ranges for identical test sensors which make it impossible to set a reliable threshold value [58]. The accuracy of the system increases when an accelerometer is incorporated with other sensors such as a gyroscope, magnetometers, and barometers [13].

2.4.1.3.2 Gyroscope

The most common feature extracted from the gyroscope sensor is the magnitude of the resultant angular velocity(w), which is given below:

$$w = \sqrt{w_x^2 + w_y^2 + w_z^2}, \tag{2.8}$$

where w_x , w_y and w_z are the angular velocity along the x, y, z-axis of the gyroscope [22]. In [59], a study was conducted to understand the use and the contribution of a gyroscope sensor when classifying physical activities. Accelerometer and gyroscope data was collected and fed into different classifiers [59]. The study concluded that adding the gyroscope sensor to the system can improve accuracy because gyroscope data makes use of orientation which most activities consist of; since the accelerometer only measures the linear motion along specified directions [59]. There are a lot of studies which combine the accelerometer and gyroscope [17], [22], [26], [60].

The disadvantage of low-cost gyroscopes is that they suffer from time-varying zero shifts and introduce significant errors in the calculated angular acceleration and angular position through differential and integral operations [17], [22], [26]. If the noise is not removed and the data is accumulating, the error can be large [17]. The Kalman filter algorithm with dynamic information of the target is required to remove the noise in order to estimate the angle [17]. In terms of availability, a gyroscope sensor is only found in higher grade smartphone devices [61].

2.4.1.3.3 Health sensors

Electromyogram (EMG) sensors measure the muscle control signals [62]. In [63], an EMG sensor is used to measure four lower limb muscle activities where Co-Contraction Indices are used to differentiate between falls and activities. In [62] acceleration and muscle activity signals are used to create a physiological monitoring system. The posture of the user is determined from the EMG signals. The tilt angle from the accelerometer is calculated to validate whether the fall that occurred was intentional or not [62]. When a person falls, the state of the heart-rate can increase anxiety [54]. In [54] an accelerometer and cardio-tachometer are used to analyse and detect falls. In [64] an accelerometer and a 3 channel Electrocardiography (ECG) circuit are used to detect the seriousness of the fall. The disadvantage of using health sensors is that they are difficult to put on and they can interfere when performing ADLs.

2.4.1.3.4 Wearable Camera

Compared to the different wearable sensors, a wearable camera provides a much richer set of data including contextual information about the environment, which includes an analysis of a variety of activities, including falls [11], [32]. The wearable camera system monitor can also work outdoors [32]. A wearable camera does not affect the privacy of the user since the system only records the surroundings of the user environment and processes everything locally on the device [32]. In [11] the study makes use of a camera system that is worn on the user's waist, which can provide continuous monitoring and is not limited to certain areas compared to static cameras. The advantage of this system is that the privacy concerns are removed compared to the static cameras [11]. In [32] a system integrates an accelerometer and a wearable camera to improve the accuracy detection [32]. The standalone accelerometer reported a sensitivity of 65.66%, standalone camera reported 74.66% and finally, a combination of these two achieved a sensitivity of 91% [32]. The wearable camera records the surrounding environment, which will make other people around the user uncomfortable, as it will seem to be recording them and is highly invasive for subjects [20].

2.4.1.3.5 Ambient sensors as wearable sensors

Ambient sensors such as a pressure sensor and a microphone can be attached to the user's footwear [7], [65]. The advantage of attaching ambient sensors on wearable items is that it can provide outdoor monitoring, and it is not limited to an area since it is attached to the user. In [7] a fall sensor that comprises of accelerometer and pressure sensor is attached to the user's footwear. The orientation of the user's foot is used to differentiate orientations from normal activities [7]. The accelerometer and microphone sensors are attached to the foot since user movement is dependent on the movement of the feet [65]. The disadvantage of the system is that it is influenced by the environment. In Table 2.2 , it provides a summary of studies that used wearable sensors.

Table 2.2. Summary of wearable sensors studies

Study	Sensors	Algorithm	Results (%)	Smart-phone
[17]	Accelerometer, gyroscope	Bayesian	ACC: 94.6	No
[13]	Accelerometer	Statistical analysis and probability	ACC: 95	Yes
[49]	Accelerometer, gyroscope, compass, proximity	Decision tree and SVM	ACC: 90	Yes
[51]	Accelerometer, gyroscope	One-class SVM	ACC: 78.3	No
[3]	Accelerometer	Comparator System	ACC: 99	No
[63]	Electromyography	Decision tree	SE: 83.2, SP: 72.4	No
[26]	Accelerometer, gyroscope	Decision tree	SE: 81, SP: 98	Yes
[32]	Accelerometer, Camera	Decision tree	SE: 91	Yes
[28]	Accelerometer, gyroscope, barometric altimeter	Decision tree	SE: 80, SP: 100	No
[11]	Camera	Decision tree	Indoor ACC: 93.78, Outdoor ACC: 89.8	Yes
[61]	Accelerometer, compass	SVM and state machine	SE: 92, SP: 99.75	Yes
[54]	Accelerometer, cardiio tachometer	Decision tree	ACC: 97.5, SE: 96.8, SP: 98.1	No
[66]	Gyroscope	Decision tree	SP: 100	No

2.4.1.4 Disadvantage of wearable sensors

2.4.1.4.1 Placement and intrusion

The major disadvantages of wearable devices include intrusion, undesirable placement of device, neglect, a user not interested in wearing them and inconvenience to the user's movement [19], [21], [24], [44], [49], [61], [67], [68]. Neglect or forgetting to wear the device, can make a wearable device an ineffective solution [8], [10], [67]. The undesirable placements of sensors on the user's body can cause obtrusiveness, inconvenience and discomfort when performing ADLs [8], [9], [10], [21]. Wearable devices which are placed on the belt, around the hip, cannot be worn when changing clothes and when sleeping which results in the inability to monitor when a person is getting up from the bed [8], [57]. The addition of extra sensors causes discomfort and inconvenience to the user. [61]. This can be resolved by allowing the user to choose the placement of the device, and the device should perform on-body sensor localization to detect the location of the device on the user [60]. This will eliminate undesirable placements. To make it convenient for the user, the trouser pocket can be used for placing device [16]. The bathroom has a high occurrence rate of falling, but it is difficult for a person to wear a wearable device since the system is affected by water and makes it uncomfortable when bathing [30], [69].

2.4.1.4.2 Power

Wearable sensors are all battery powered, which means they cannot be used when the devices are recharging or when batteries need to be replaced [19], [29], [68]. The battery problem can be compensated by using a low sampling frequency scheme together with a hierarchical scheme methodology [13]. This will also reduce the computational complexity of the system thus saving processing time [13]. To make the system usable in reality, a smaller number of sensors is preferable on the user [4]. The advantage of keeping the number of sensors to a minimum is that it can cope with resource constraint issues such as battery power, storage, computational power, and network bandwidth [4].

2.4.1.4.3 Hardware and software

Wearable devices are limited to the hardware and software [49]. Each smartphone device has a fixed number of built-in sensors and to add more sensors, the smartphone has to be upgraded. A basic sensor that is available in all smartphones is the accelerometer sensor. Compared to microcontrollers where the software is fixed, the software of the smartphone can be updated anytime in the relevant software marketplace. Smartphones can address the problem of low-power microcontrollers where classification algorithms are constrained to limited memory and processing power. Most microcontroller systems only implement threshold classification, whereas smartphones can implement machine learning algorithms on it.

2.4.1.4.4 Generates a lot of false positives

Wearable sensors generate a lot of false alarms when performing daily activities, which can lead to frustration of users [8]. The reason for poor accuracy and high false positives when using accelerometers is lack of adaptability and lack of context understanding [46]. False positives can be limited by implementing communication between the user and the device. If a fall has occurred, the user is communicated to first, to determine if a fall has occurred. If the user does not respond within a specified time period, the emergency service is communicated to [70].

2.4.2 Ambient sensors

The ambient device makes use of event sensing by collecting and examining the environment which can monitor the elderly person's movement. This is done through the use of external sensors which are attached around the surrounding environment such as a home or close to the subject [14], [20], [21], [54]. The other application that ambient sensors provide is indoor localization and security [21]. The advantage of ambient devices is that the user does not need to wear the device or remember putting it on, it is passive and unobtrusive [17], [49]. Ambient devices are non-intrusive and invisible to the elderly, which would not affect user privacy [23]. Ambient devices are cheaper, but less intrusive compared to camera-based systems [49].

2.4.2.1 Vibration detection

Ambient devices make use of vibration data where detection of falls are based on the characteristics of vibration patterns [11], [49]. Vibrations can be used to detect falls based on the observation that normal activities cause measurable vibrations on the floor. This means when a user falls, the impact caused by the body on the ground will generate vibrations that will be transmitted throughout the floor [71], [72]. Using events and changes in vibration data makes it useful for monitoring, tracking and localization [49]. In [71] a system comprised of a special piezoelectric sensor which is coupled to the floor surface by means of mass and spring arrangement was used to capture the vibration pattern. In [72] floor-mounted MEMS accelerometer sensors are used to capture the vibration signal around the floor. Floor vibrations are inexpensive, and they can preserve the privacy of the user, but the performance is influenced by the floor type and detection range is limited [39].

2.4.2.2 Acoustic detection

The basic idea of an acoustic sensor is to make use of a microphone sensor to capture the movements of users where MFCC features are extracted to detect falls. MFCC features are extracted by first removing the high-frequency component and segmentation of the audio signals into different frames [9]. A fast Fourier transform (FFT) is applied to each frame to get the frequency spectral features [9]. After the FFT, mel-scale mapping is performed and finally, discrete cosine transform is applied to obtain 12 MFCC [9]. The acoustic system makes use of a Rescue Randy doll for mimicking human falls for testing the system [29].

In [37], the system classifies other noise from a home environment from falls by applying a universal background model which was trained from the shared acoustic feature space for falls and other noise. To remove the effects of environmental noise, the system can combine floor vibration sensors and microphones [30]. There are a lot of sounds in a realistic home environment and by combining sound and vibration information, it is possible to accurately detect a fall [30]. Audio solutions require the installation of several sensors around the building, which are usually placed on the ceiling, or near the falls [29].

In [39], an eight circular microphone array is employed. When a sound is detected, the system locates the source of the sound, enhances the sound signal and then classifies the sound signal [39]. The sound signal is located using the steered response power with phase transform technique, which can work in any condition [39]. The sound signal is enhanced using a beam forming technique [39]. Applying the beam forming technique can enhance the desired signal and reduce interference [39].

Classifier design for acoustic fall design is difficult to design since it is impossible to obtain realistic fall sound signatures for training and testing of the system [43]. Generating fall data is difficult to simulate [43]. When capturing simulating falls, the test subject tries to prevent a painful fall [43]. Most of the acoustic studies make use of Randy Rescue dolls which make detecting low impact falls difficult. To address this, differently weighted rescue dolls are needed to train the system [69]. The studies which make use of Randy Rescue dolls cannot replicate realistic fall sounds due to the hard skin and the lack of bones of the mannequins [43].

2.4.2.3 Movement detection

This method is comprised of the following sensors: pressure sensor, Doppler sensors, passive infrared sensors, and electric near-field sensors. The advantage of the movement detection sensor is that it can be used to perform indoor localization [23].

2.4.2.3.1 Pressure sensor

Pressure sensors are the most common method for ambient sensor since they are low cost and non-obtrusive and a fall is detected based on sensor pressure changes [21]. If the person is closer to the sensor, the pressure is high [73]. In [73] a carpet was used where four piezoresistive sensors were placed at each corner of the carpet. The piezoresistive sensor makes use of atmospheric pressure as a reference from an air pump to measure the exerted pressure on the carpet [73]. The sensor consists of two chambers where one is the reference chamber where the atmospheric pressure is applied and the other one is the measurement chamber of the analogue pressure that is exerted [73]. A differential voltage between the two chambers is outputted by the sensor, where a threshold decision tree is applied to the value to detect a fall [73]. A fall is detected when the person falls and 3 or 4 sensors are detected and the pressure value is greater than the threshold [73]. The disadvantage of pressure sensors is the low detection precision which is below 90% [21]. The disadvantage of only using a pressure sensor to detect a fall is that it can sense the pressure of everything in and around the object, which leads to false positives hence low accuracy is achieved [21], [49]. The distance from impact to where the pressure sensor is located can impact the accuracy of the system [21]. Another problem with using only pressure sensors is that they cannot differentiate between lying and falling postures [23]. To solve this problem in [23], they make use of intelligent tiles which consist of pressure sensors and three-axis accelerometers. The system has each tile placed with a 3-axis accelerometer in the centre of the tile and has four force sensors at each corner of the tile [23]. The posture is detected using the pressure sensor by reading the pressure amount, the pressure duration on a tile, and the tiles proximity using a simple algorithm [23]. The accelerometer is used to detect hard human falls, but cannot detect soft falls [23]. The accelerometer is used to enforce the differentiation between the falling and the lying down posture [23]. The disadvantage of the system is the cost associated with each tile, and it requires a power supply for each tile [23]. Pressure sensors can have high false alarms because the person's weight is not factored in when detecting a fall and the system is usually implemented on a small scale, e.g. a mat, which makes it costly to implement in a home environment. The factors which influence a pressure sensor are the placement and sensitivity to pressure.

2.4.2.3.2 Passive infrared sensor

A Passive infrared (PIR) sensor detects falls using infrared signatures [40]. The received signal strength from the PIR sensor changes with the motion of an object [74]. The strength of the received signal from the PIR sensor changes with the motion of a hot object within the range of the sensors [74]. The PIR sensor cannot be used to differentiate falls since a walking person can produce a signal similar to a PIR sensor [74]. To solve this, in [74], a combination of both PIR and floor vibration sensors is used to detect a fall. PIR sensors reduce the false alarms in the system by detecting if the vibration signal is caused by a human and by detecting the presence of the user [74]. The biggest problem of using PIR sensors is the line of sight and coverage area.

2.4.2.3.3 Doppler sensor

Doppler sensors are motion sensors that can sense, track, and recognize moving objects and monitor human activity [38]. Doppler sensors are small and cheap, only detect moving targets by suppressing stationary background cluster and are noise tolerant systems [75]. A Doppler sensor has different irradiation direction which is less sensitive to movements orthogonal to the irradiation direction but when moving in the irradiation direction it becomes sensitive [75]. A Doppler continuous electrometric wave signal is sent at the carrier frequency and the reflected wave is received which contains the frequency shift of the moving object [75]. The velocity of the moving object can be determined by the frequency shift within the detection range [75]. To detect if a human fall has occurred, the spectrogram analysis was performed on the Doppler frequency shift measured by the system using short time Fourier transform, and the power spectral density as calculated [75]. In [38] a system makes use of a Doppler sensor and ceiling radar since Doppler sensor is sensitive to motion when there are a lot of people at home. This can result in added noise to the activity data, which can result in false alarms. Doppler is sensitive to motion and can penetrate apartment walls [75].

2.4.2.4 Disadvantage of ambient sensors

2.4.2.4.1 Coverage

Ambient sensors only work indoors or where the devices are confined and in dead spaces. They suffer from blind spots, have a limited recording area, can only monitor one person and can be an expensive setup [11], [13], [17], [29]. Since ambient devices are limited to certain areas, they cannot be used in outdoor monitoring which ties the user inside [23]. Most ambient systems assume that only one person is present in the room being monitored.

2.4.2.4.2 Noise

Ambient sensors are affected by environmental interference, background noise and ambient noise [12], [19]. Ambient devices can produce many false alarms due to other falls caused by everyday objects [14]. Acoustic and vibration sensors can only work on certain floor types. Movement sensors are affected by obstructions or occlusions which can deteriorate the signal.

2.4.3 Camera-based methods

The advancement in computer vision and image processing techniques can also be applied in fall detection problems, where a camera sensor is used to monitor the user behaviour and detect fall activities without interfering with the user's routines [14], [20]. Visual monitoring can provide a lot of advantages provided privacy is handled properly, such as detecting unusual activities like falls and has low intrusiveness [8], [25], [31]. Using computer vision to detect a fall can be difficult since the human body is made up of numerous parts which can move freely, which makes the process of identifying and locating people more difficult [10]. To overcome the problem, the current studies use human parts which can be detected such as the head, waist or feet [10]. The benefit of camera-based methods is that there is no intrusion on the users since the sensors do not need to be worn. Users are not required to remember to put it on because the camera system is contactless. The system can be used to monitor one or more people simultaneously, and can be used to detect falls in public areas [17], [20], [31], [44], [46], [49]. Multiple people can be tracked in the frame through segmentation and marking module [20], [21]. Camera-based methods can be used to serve two purposes: fall detection and security monitoring. The advantage of camera-based methods compared to other methods is that it is more robust, can accurately detect falls and different ADLs, it has low intrusiveness and it can verify a fall remotely if it has occurred [8], [5], [20], [49], [76]. Camera-based systems are best suited for situations where multiple people need to be monitored e.g. hospital rooms or aged care homes [24]. With camera systems, multiple events are detected simultaneously with less intrusion. Figure 2.3 shows how a camera system detects a fall.

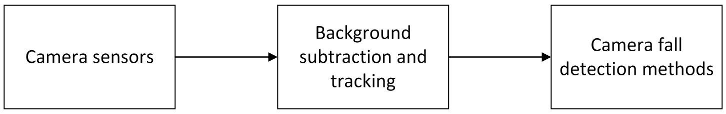


Figure 2.3. Operations of the camera system when performing fall detection.

2.4.3.1 Camera sensors

Falls can be detected using a single RGB camera, or a 3D-based method with multiple cameras or depth cameras [8], [21], [76]. The most popular vision-based method is the RGB camera which is the cheapest and is easy to setup [21], [24]. Multiple cameras are required to cover a large area and this can be resolved using omnidirectional cameras or a wide-angle camera [10], [31]. The wide-angle camera has a wide field of view lenses which can be used to monitor large areas [10]. The problem with this type of camera is that the images produced are highly-distorted [10]. The camera lens has high radial distortion which needs to be corrected before the calibration process starts [10]. Omni-camera can capture a 360 degrees view in a single shot which compensates for the blind spots [77]. The lack of in-depth information from RGB cameras can lead to a lot of false alarms [8], [31], [46]. The 2D camera methods can cause misjudgements when there are more than 2 people in the frame [67]. A single camera cannot extract features that characterize a 3-D object movement which creates a robust fall detection system, but this can be created from multiple RGB cameras [21], [68]. Multi-camera systems construct a 3-D object from back projecting multiple silhouettes where features such as velocity are extracted for detecting falls [68]. For multi-camera systems, installation, calibration and synchronizing, the cameras in the same reference frames is difficult, time-consuming and increases the cost of the system [21], [68]. The 3D techniques which are implemented from RGB cameras are not automatic and require manual initialization. Appearance deformation can occur as a result of the 2D grey or colour images that are the projection of 3D targets [21]. The use of stereo vision techniques of multi-cameras results in high computational costs [67]. The use of colour cameras in a controlled environment achieves high accuracy, but would not work in a non-controlled environment where the lighting and tracking of a user are fully controlled [8], [31].

Depth information alleviates problems where users or objects do not have consistent colour and texture, but they need to occupy an integrated region in the 3D space [46]. Depth camera allows a person to extract from the image at low computational cost [46]. Depth cameras can be used to calculate the distance from the top of the person to the floor [21]. Depth cameras can perverse the privacy of the user, and the light conditions do not have any effect on it [8], [21], [24]. Depth images can be extracted in dark rooms from an infrared light [8]. Depth cameras can also be used to solve occlusion problems and tracked key joints of the human body [21], [67]. The different depth cameras include stereo vision, Time-of-Flight and structured light camera [48]. Stereo vision camera constructs a depth image from two views of a scene [48]. The problem with this camera is that the system needs to be calibrated, it is computationally expensive and fails when the picture does not contain enough texture [48]. The system cannot work in low light conditions, which can be solved by integrating an infrared light to it, but the loss of colour information can cause segmentation and matching difficulties [48]. The earliest depth camera was the time-of-flight 3D camera, but the cost of setup is expensive, and it is restricted to a low image resolution [21], [67], [48]. A time-of-flight image can be used to obtain partial volume information which returns precise depth image compared to stereo vision cameras for tackling occlusion problems [48]. The most popular depth sensor is the structured light camera which includes the Kinect sensor [48]. The Kinect sensor can track the body movements in 3D unlike 2D [8]. The Kinect sensor can be used for human behaviour recognition, and detect a fall in 24 day-night cycle [46]. The Kinect sensor is not affected by the external light conditions due to the depth interference done by making use of an active light source [8]. The limitation of the Kinect sensor is that the sunlight interferes with the pattern-projecting laser, which is not suitable for outdoors [8].

2.4.3.2 Background subtraction and user tracking

Background subtraction is performed to extract the moving object from the image known as foreground segmentation. The simplest background subtraction technique requires an original image with no moving objects. The current frame is subtracted from the original image to obtain the moving object. The disadvantage of this technique is that it does not take into account the lighting changes, shadow changes, and the changes in the background due to short-term movements [78]. This can be solved using a GMM background model or using approximate median filter [44], [78], [79]. Morphological operations can be applied to reduce noise in the background. The extracted object is tracked continuously until the object is out of camera view angle.

2.4.3.3 Camera-based detection methods for fall detection

The vision-based can be split into shape change, inactivity, posture, and 3D head motion [14], [21], [49], [54]. Shape analysis is used when the shape of the person changes from an upright position to a falling position over a certain time period [34]. A simple method for detecting a fall using 2D methods is to locate the person in the video, and draw a bounding box around the person [48], [68], [80]. The most common 2D feature that is extracted is the aspect ratio [44], [48]. A small aspect ratio means the user posture is upright, whereas a high aspect ratio means the user posture is lying down [44]. Ellipse provides greater information than the bounding box e.g. fall angle [21], [48]. A small angle shows that person has fallen down [44]. The problem with using a bounding box alone is that it does not provide enough information regarding the human motion, and the performance of this technique relies heavily on the camera view angles [25]. Analysing aspect ratio can be inaccurate due to the position of the person, camera and occluding objects. Shape analysis can be implemented in real-time, but 3D shape requires more computation and more cameras which can become unreliable. Most shape-analyse algorithms make use of a few features which are not enough to detect sub-fall actions. The method of analysing a fall by placing a bounding box around a person can be efficient only by placing the camera sideways and the accuracy of the system depends on the occluding objects. Body silhouette changes and body orientation fail to detect falls where there are occlusions. Accuracy is affected by the proximity of shape attributes [34].

Inactivity detection is when a fall is detected within an inactivity period of time that the person has been in a lying state. The use of inactivity analysis is fast as it has light computing load where it can run on a small computing device. False alarms are generated since they rely on the context information such as period and threshold and require the person to lie on the ground for long which can cause serious injury. Some users get up from the fall rapidly, hence the static activity after a fall has occurred is not captured [4].

Falls are detected by classifying the posture of the user. Posture recognition achieves high accuracy as it correlates the observed video sequences to the stored ladled video sequence [10]. From the body, positions posture can be determined. Posture recognition is computationally very expensive [10]. Posture recognition requires a huge database to successfully recognise the different postures. Posture is hugely affected by occlusions and requires advanced video segmentation algorithms to avoid human intervention [42].

A fall is detected if there is an occurrence of considerable movement associated with the head. Head tracking can help to avoid occlusion problems since the head is always visible in the scene [81]. Simple head tracking is to represent a head by an ellipse and update it by a local search using gradient and colour information. The disadvantage of head tracking is that the head speed is greater when sitting down fast which generates false alarms, and the head speed is less during slow fall period which results in fall not being detected [82].

2.4.3.4 Disadvantages of camera-based sensors

Camera-based methods' accuracy is dependent on how efficient and accurate shape modelling methods used are [49]. The problems with camera-based systems are occlusions, light conditions, coverage, privacy, cost and high processing.

2.4.3.4.1 Occlusions

Elderly people tend to collect furniture or antiques which have sentimental value [44]. When they move to a smaller residence, they tend to take all these items with them resulting in the room being filled with these items; which means the user is partially occluded when moving around the room [44]. Image processing difficulties arise when changes occur in the monitoring area e.g. furniture being shifted around the room which can affect the accuracy of the system [9], [44]. To accomplish the bounding box, the RGB camera should be placed sideways but this can fail due to occlusions [80]. To solve this, the camera is required to be placed higher in the room not to suffer occlusions and to have a greater field view [80]. In this case, depending on the relative position of the person, the field of view of the camera, a bounding box will not be sufficient to discriminate a fall from a person sitting down [80]. To avoid occlusions, some researchers placed the camera on the ceiling, where the 2D velocity of the person is used to classify the person. The problem of velocity in a 2D method is that it increases when the person is near the camera, which makes the threshold for differentiating falls from sitting down fast difficult to define; and 2D methods also suffer from occlusion problems. This can be easily solved using 3D vision systems [34], [48], [80]. Occlusions can also be solved by using multiple cameras to create a 3D system, but it is difficult to extract the 3D silhouette. Monitoring the whole body can fail when elderly people who struggle to walk are assisted by a walking aid such as a rollator or walking frame which causes the lower part of the body to be occluded by the system and when objects are carried [44], [82]. Head tracking can also be used to solve occlusion problems, where the user is covered by objects [81]. The use of colour or intensity distribution to track the person would not work, since the appearance of the person changes over time e.g. changing of clothes [44]. Other moving objects which are not taken into consideration, can affect the system e.g. opening of cupboards and opening of a door can hugely affect the system, as the other side of door changes, a door could be considered a user [9], [44].

2.4.3.4.2 Light

The camera system should be able to monitor the user in any light conditions [9], [44]. The different light sources and the different light intensities that occur during the day can result in overexposure in some parts of the image and decrease the quality of the images [8], [12], [14], [44], [76]. Overexposure can be slightly compensated through careful placement of the camera in the room [44]. The disadvantage of foreground extraction using a traditional camera is that it relies on background modelling in colour image space when in reality, it is affected by lighting conditions and shadows [8], [31], [46], [68]. The use of colour-based shadow detection algorithms can improve the output of the background subtraction algorithm; but these algorithms rely on the assumption that if an area is covered by a shadow, only the brightness of the image is affected and there is no change in colour information [44]. This assumption does not work in reality [44]. There is a high risk of falls occurring in low lighting conditions compared to normal illumination conditions [46]. To solve the issue of lighting conditions for single cameras, an active source of infrared (IR) light can be installed along with the camera however, there will be no colour available due to the IR illumination for background modelling [68]. Colour information is not available in near-infrared night images and colour images that are available during the daytime are not reliable [44]. Depth cameras can resolve the lighting conditions and can work in both day and night conditions [48].

2.4.3.4.3 Cost and high Processing

The infrastructure and installation of sensor equipment are expensive. The image quality is much lower in reality compared to the lab experiment setup. This can be resolved by installing a high -quality camera, which results in high-cost [44]. Most camera-based systems require the video captured to transfer to a central location for processing which requires extensive communication [11]. Camera-based systems require a hard-wired power source, high bandwidth network connection and a PC for the processing [69]. They also require considerable computational power running real-time algorithms [46]. Computational power can be minimised by integrating the camera-based system with an accelerometer [46]. Camera-based systems only start processing when a possible fall is detected from an accelerometer sensor [46]. An accelerometer sensor is used to identify if a possible fall has occurred and the camera system is used to authenticate a fall [31]. The frames are not processed instead, they are stored in a circular buffer and only processed when a fall has occurred [31].

2.4.3.4.4 Coverage

Camera-based systems can only work indoors or where the devices are confined to, which can create blind spots, occlusions cannot be detected, limited field view and dead spaces are created [5], [8], [12], [13], [14], [17], [49], [76]. Multiple cameras should be installed to solve these problems and provide continuous monitoring, which increases the cost of the system [44]. Wide angle cameras can be used to provide coverage of the room, but the spatial resolution of the camera system decreases due to the lens of the wide angle cameras [44]. Omni-cameras can also be used but they are heavily affected by light conditions.

2.4.3.4.5 Privacy

The ethical issues associated with camera-based methods include confidentiality and privacy of the monitored person, which makes it difficult to monitor a person in the bedroom and bathroom [8], [14], [49], [76]. The problem of colour camera-based systems is that they contain facial characteristics of users which result in privacy concerns. These concerns can be addressed by capturing low-quality images, using depth images or image processing technique such as silhouettes [46], [68], [71]. Even though privacy techniques are applied, people still have a feeling of “being-watched” based on their perception of a camera system [31], [71]. Instead of capturing the user, the environment scene can be captured like in [11] and [32].

2.5 PROBLEM OF CURRENT FALL DETECTION SYSTEMS

High classification accuracy is reported in almost all the fall detection studies, but the studies were tested on a small set of subjects, fall types and activities [4], [27]. The reason for simulated falls is that it is extremely hard to collect real-world elderly person fall data; since 30% of the elderly population over the age of 65 years fall at least once per year [27]. Current fall detection studies are only tested in controlled experiments where they achieve high accuracy, but when placed in the real world the accuracy of these systems decreases [11]. Studies test the specificity of ADL through laboratory experiments by the same subjects who generate fall data [27]. These data could be biased since these activities performed are not spontaneous [27]. The choice of the mattress to reduce the impact of the falls to protect volunteers can reduce the accuracy of the system when applied to the real world [27].

It is difficult to make a fair comparison between the different fall detection studies since each study makes use of its own dataset from different conditions [18]. The problem comes in when comparing systems since each study validated its research on different data collection protocols, subject groups, and environment settings, hence they cannot be directly compared to previous studies [4]. The factor which influences performance is the number of training samples used for training the system [18]. The main problem of acceleration based studies is that it is difficult to compare the different studies because each research study makes use of its own dataset composed of simulated falls and ADL [18]. It is difficult to judge whether the results obtained from these studies are influenced by the dataset compiled, and it is impossible to make a comparison since the dataset used in each study is different [18]. Since these devices are required to be worn long-periods or the whole day, a complete dataset is required compared to fall detection studies where the dataset is limited [11].

In [27] evaluation was conducted on real falls based on accelerometer fall detection algorithms where 29 real-world falls were tested. The results of the evaluation show reduced sensitivity and specificity values compared to when conducted in an experimental environment to evaluate the effectiveness of the algorithms to detect falls in real-life events [27]. The study achieved average specificity of the algorithms of 83.0% and the average sensitivity of the algorithms is 57.0% which is much lower compared to the simulated environment [27]. There is a huge number of false alarms generated from the algorithms in a one-day monitoring period which ranged from 3 to 85 [27]. The results obtained from the study encourage researchers to take real activities into consideration [27]. Unbalanced data are a challenge, where a balanced dataset is required for a supervised machine learning algorithm [44].

The main problem with vision-based systems is the absence of flexibility, as these systems are case specific as they are designed and optimized for a certain situation or scenario [14]. Camera-based study algorithms are evaluated from data collected from the controlled environment with optimal conditions such as perfect illumination, simple scenarios or scenes and falls are simulated by actors [44]. The challenges found in real life data compared to the simulated data are that the image quality is low, falls are rare and vary a lot in terms of speed and the nature of fall [44].

2.6 PERSONALIZATION

2.6.1 Why the need for a personalization model?

It is difficult to detect the different types of falls for different types of people since the input signal has high variation among various body types [32]. The phone placement differs from person to person [32]. Personal information can make the system smarter by adapting the different parameters for different people [77]. If different body postures are not learnt, a high false rate could occur [17]. The occurrence of falls is rare, which results in insufficient or no data available [5]. Different types of falls can occur, which makes it very difficult to model [5]. The problem with the current studies is that data for detecting falls is for voluntary, not actual falls which occur in real life [35]. Collecting fall data is futile as it requires a person to perform a real fall which can result in serious injuries [5]. About 94% of fall detection studies used simulated falls from laboratory experiments for training the classifiers [83]. This shows the difficulty in obtaining real fall data [83]. Instead of real falls, artificial falls are collected in a controlled laboratory environment, which does not represent an actual fall [5]. The advantage of an artificial fall is that it provides information about how falls occur but do not make it easier for detecting falls [5]. Classifiers which use artificial falls as training data can result in over-fitting, which can cause poor decisions on the actual fall [5]. The fall data is limited in quantity and suffers from ethic clearance [5]. To get accurate fall data, a long-term experiment needs to be conducted in nursing homes using wearable sensors, ambient sensors, or camera-based methods [5]. This can be solved through personalization.

Methods that make use of thresholds are most popular and easy to implement and computationally inexpensive however, they do not work on different people and do not provide a good trade-off between false positives and false negatives [5], [35]. People have different types of body figures thus using the same threshold in fall detection algorithm will not work for everyone or would not be optimal [77]. With thresholds, it is difficult to adapt the threshold to new types of falls and make it work on different people [5], [14]. Duration of falls for elderly people may be longer compared to young adults [34]. The values from the threshold method are determined without being based on any theories and/or experiments, and the fall detection model fails in that it cannot address inter-individual difference [35].

Each individual has different characteristics and motion patterns compared to people used in training data [35]. Questions arise on the validation of these systems when most fall detection studies make use of voluntary falls during simulation compared to accidental falls, which are mainly involuntary [35]. Studies have been done which make use of knowledge about falls such as a change in acceleration and it's short duration [5]. The problem with these studies is that they cannot work in the real world since no training data for falls was used and a low accuracy will be achieved since the classifier cannot predict a fall that it never observed before [5]. ADLs such as lying down and sitting down can generate high impacts which can be misclassified as a fall for overweight users [4]. An actual free fall is not created due to the cautiousness of subjects, which results in an inaccurate fall detection [11]. Even when safety precautions are there, subjects are still too afraid to fall [11].

2.6.2 How to create a personalized model?

Classifiers can be trained using the user data or non-user data or a combination of both user and non-user data. The use of a supervised machine learning algorithm cannot solve the problem as the fall data that are used are from simulated falls [5]. Since falls are rare, supervised machine learning algorithms cannot be used since they can only classify known classes if their classifier was trained with that data [5]. Supervised machine learning algorithms require the data to be labelled which wastes time and effort [5]. Supervised classifiers cannot provide a person-specific solution for individuals [42]. Due to the lack of fall data, supervised classification algorithms may not work as desired. The following classification can be used instead: over/under-sampling, semi-supervised learning, cost-sensitive learning, outlier/ anomaly detection and one class classification [5].

A large dataset needs to be created for training the supervised classifier and it should contain data for different activities. If a person does not fit the dataset e.g. if the person is obese, a good performance cannot be obtained for the specific individual [42]. Supervised learning algorithms require a balanced dataset which has equal misclassification costs for different classes [5]. When unbalanced data are used to train the algorithms, the algorithms fail to distinguish the characteristics of the data, which results in low accuracies and their prediction tends to favour the majority class [5]. The imbalance class can be handled by performing cost sensitive-classification, where the cost of the classification problem is treated differently [5]. This can be accomplished by adding a cost matrix to a cost-insensitive classifier or by integrating a cost function in the classification algorithm to generate a cost-sensitive classifier [5]. A cost matrix of a fall detection problem is defined by getting the optimal decision threshold of the classifier [52]. The costs are unknown and are difficult to compute [5]. In [44] the study makes use of a weighted SVM to compensate the imbalance of data of the falls and normal activities from the camera. The weights are determined using cross-validation and grid search maximizing the AUC of ROC [44]. The lack of fall data could also be compensated by using sampling techniques to generate fall data [5]. Fall can be oversampled or the normal activity class can be under-sampled to train a supervised classifier [5]. The disadvantage of oversampling is that it can lead to over-fitting if a lot of artificial data points are generated and do not represent a fall [5]. The disadvantage of under-sampling is that it can lead to under-fitting if the normal activities class is reduced to match the number of total activities of falls [5].

Classifiers that only require normal activities for training, which eliminates data imbalance between fall and normal activities are known as unsupervised machine learning algorithm [5]. If the normal behaviour is not properly learned, it can result in a lot of false positives, as a slight variation from normal activities can be detected as a fall [5]. The classifier needs to adapt and learn new activities to reduce the false alarm rate when detecting falls [43]. The advantage of the one-class approach is that the classifier can easily adapt to new data without data imbalance concerns[43].

2.6.3 Who did this so far?

There have been a few studies that have tried to personalize the system to suit the users. In [84], a study was conducted to compare personalised systems to non-personalised systems using a smartphone accelerometer. Three unsupervised methods were implemented: NN, local outlier factor, and one-class SVM and one supervised method SVM [84]. The study was divided into two stages: the first stage to determine the best-unsupervised method and the second stage to determine how the personalized system performs on both the best-unsupervised method and supervised method [84]. The raw data of the three axes of the accelerometers are fed into the classifiers [84]. From the first stage, it was found that NN outperforms the rest of the unsupervised methods [84]. For the second stage, the personalized model of the NN is trained with the normal activities of the user whereas the non-personalized model is trained with the normal activities of other people's data [84]. The personalized model of the SVM is trained with the normal activities of the user and fall activities of other people whereas the non-personalized model is trained with both normal and fall activities of other people [84]. It was found that both the personalized model, NN and SVM, outperform the non-personalized model [84]. The personalized SVM model achieved a slightly higher geometric mean of 0.9764 compared to the personalized NN model of 0.9688 [84]. The NN model is better compared to SVM model because it can adapt to the new data and it can recognize more fall types.

The basic personalization is to customize the threshold based on personal characteristics such as height, weight, etc. [70], [77]. This type of approach is applied in [70] and [77]. In [77] an Omni-camera is used to record the activities, where a bounding box is placed on the user [77]. The system requires a background image with no user present in the background [77]. To detect a fall, the foreground is extracted by performing background subtraction [77]. A fall is detected if the bounding box aspect ratio is greater than pre-defined threshold value [77]. The pre-defined threshold value is customized based on the following personal information: height, weight, and electronic health history [77]. The personal information is used to adjust the detection sensitivity which reduces false alarms and provides more attention to the elderly person with specific needs [77]. The use of electronic health history increases detection sensitivity automatically if the person experiences cardiovascular disease or if a fall incident has happened before [77]. In [70] the tri-axial accelerometer from the smartphone was used, along with a threshold classification algorithm to detect a fall and the thresholds were based on the user personal details. The accelerometer data is compared to different types of pre-defined thresholds to determine if a person has fallen [70]. The following personal details were included: height, weight and level of activity [70]. The system can adapt to different user movements and does not work only when placed on the user chest or trunk position, but other positions as well [70]. When a fall is detected, an alarm would go on, if the alarm is not turned off due to a false alarm, an emergency alert would be sent [70]. In [85] a smartphone system which is based on the user information's such as the ratio of height and weight, sex and age are used to adjust the threshold value and sampling of the acceleration data. From the tri-axis acceleration sensor, the direction of the three-axis was extracted [85]. The system calculates SMV [85]. Based on the user BMI, the user age, and sex; the maximum and minimum threshold from the acceleration and the sampling frequency is determined by the range of the personal information [85]. A fall is classified based on the thresholds and the system achieves an SE of 92.75% and SP of 86.75% [85].

Wearable glasses which consist of a magnetometer, an accelerometer and a gyroscope are used to detect falls [86]. A self-adaptive fall detection algorithm is used to detect falls by collecting information about the user ADL and adjusting thresholds of the system [86]. The thresholds are adjusted using the GMM. Thresholds are adjusted for head rotations and falls in [86]. A magnetometer sensor is used to detect normal events and filter them out by using a GMM which is based on the variations of rotation angles [86]. The SMV thresholds are determined by a GMM and an optimized thresholding technique

[86]. The fall is based on the SMV obtained from the accelerometer and the direction fall is based on the accelerometer and gyroscope sensor [86]. The system achieves an accuracy of 92.1% [86].

Another approach is to adapt the classifier to accept additional ADL data and re-train the classifier in order to learn the user movements. In [33] the system is personalized by optimizing the activity patterns so the system can adapt to different user characteristics. Personalization of the system is done in three phases: an initialization phase, personalization phase, and adaption phase [33]. The initialization phase is when the user uses the system for the first time, the activity patterns are retrieved from a global database and are adapted to the user characteristics, which are then stored in a local database [33]. The personalization phase will occur daily and randomly for a certain period and a certain number of new activities from the user will be added to the local database [33]. In the adaption phase, the monitoring system will adapt to changes in the level of user activity [33]. A fall is detected using a threshold method with a two-stage process, where the first stage determines if a possible fall has occurred and the second part to detect if the fall has actually occurred [33]. In [45] a smartphone tri-accelerometer sensor was used with an NN classifier; where the capture magnitude acceleration data is compared to the store ADL data from the smartphone. A fall is detected when the difference between the store pattern and the incoming pattern is high [45]. The new ADL is added every time the system classifies the incoming data as ADL where the old ADL record is replaced with new ADL [45]. To reduce processing power and computational time, the system only classifies when the magnitude of the acceleration value is greater than 1.5g, and if long lie occurs [45]. The advantage of NN classifier is that it easy to add new data and it does not require simulated falls for training the system [45]. The simulated fall data was used only for testing the classifier [45]. The disadvantage of the system is that it cannot detect soft falls and it uses long-lie. If a person attempts to get up from a fall but fails each time during the long-lie period, the system will not detect a fall event [45].

The following papers [9], [51], [43], [42], [41] encourage the concept of personalization in terms of retraining the classifier with new records for new activities and learning the user movements, but these studies never implemented the concept. The most popular classifier among the studies is the one-class SVM. Table 2.3 below summarises the systems which make use of personalized models.

Table 2.3. Personalized fall detection studies summary

Study	Sensor	Method of personalization	Features	Algorithm	Results before person-alization	Results after person-alization
[77]	Omni- camera	Changing the pre-defined threshold value based on user height, weight and electronic health history	Aspect ratio	Threshold tree	ACC: 78%	ACC: 90%
[84]	Smartphone accelerometer	Training the system with only personalized data	X, Y, Z axis from the accelerometer	NN	SE: 94.15% SP: 93.84%	SE: 96.65% SP: 97.15%
[84]	Smartphone accelerometer	Training the system with only personalized data	X, Y, Z axis from the accelerometer	SVM	SE: 96.48% SP: 95.73%	SE: 97.97% SP: 97.34%
[45]	Smartphone accelerometer	Adding new records to the system	SMV	NN	AUC: 0.969	AUC: 0.978
[86]	Magnetometer, accelerometer, gyroscope	Self-adaptive update the threshold for the user	SMV	Threshold tree	ACC: 90.7% SP: 97.7% SE: 79.8%	ACC: 92.1% SP: 98.7% SE: 81.7%

2.7 CHAPTER SUMMARY

In this chapter, the different fall detection systems that exist were discussed and analysed and each one has their own advantages and disadvantages. The accuracy of the system depends on the sensors used and the type of classifications. Wearable and camera-based sensors are the most popular ones compared to ambience sensors. Ambience sensors are highly influenced by the environment. The wearable sensor can include a device of MEMS sensors or the use of a smartphone and the system can include a false alarm button. The main disadvantage of a vision-based sensor is limited coverage and its performance is affected by objects in the environment. The wearable sensor's main disadvantage is that it is intrusive and the placing of the device on the human body is uncomfortable. Wearable sensors are the preferred method as they are practical, allow continuous monitoring and are not influenced by the environment. The wearable sensor also provides outdoor monitoring and can be used to collect real data in a cost-effective way. A smartphone can be used as a wearable device since a lot of people have it and it is not intrusive. A wearable device can be placed in the user's pocket, which would not interfere when the user is performing ADLs. Experimental systems are limited to the laboratory setting, which would not work in reality and is limited to certain ADLs. Personalization is key in fall detection systems since it does not only increase the accuracy of the system but can also be adapted to learn new activities. Adapting to new activities can be done by implementing an unsupervised machine learning algorithm since data balance would not be an issue. In Chapter 3, the design and approach of a personalized wearable fall detection system are discussed and implemented.

CHAPTER 3 METHODS

3.1 CHAPTER OVERVIEW

In this chapter, the methods of designing and implementing a personalized fall detection system are discussed. In Section 3.2, a discussion on which type of wearable sensor to use and implement is provided. In Section 3.3, is the simulation section, where a public dataset was used to investigate the features to extract, the different machine learning algorithms and different personalised models that can be used. The simulations were done using Python scripting language. In Section 3.4, the experimental section discusses the implementation of the Android application system and the tests performed to evaluate the fall detection system.

3.2 SENSORS

From the literature study, it was concluded that accelerometer sensors were the most used wearable sensors for detecting falls because they can measure free fall of the body, the impact of the body on the floor and determine if the user is in lying state which all of the above occur when a fall is performed [87]. The smartphone accelerometer sensor was chosen because smartphones have a good degree of accuracy when calculating the free-fall time [88]. The biggest advantage of using a smartphone accelerometer is that it is available in all low-cost smartphones and allows for both indoor and outdoor monitoring [88].

3.3 SIMULATION

The purpose of the simulation is to investigate the following questions:

- What type of features should be extracted to achieve the best accuracy?
- Which machine learning algorithm will result in high accuracy?
- What type of personalized model should be implemented?

From the results, the best feature space, machine learning algorithm and personalized model will be implemented on the Android smartphone device. The following steps in Figure 2.1, will be followed in Section 3.3.2.

3.3.1 Data collection

There are quite a few public fall datasets that are available online. The following parameters vary among the different fall detection datasets [88]:

- Number of subjects participated in the experiments.
- The characteristics of the subjects such as weight, age and height.
- The environments where the experiments are being performed are different such as gym hall, office, lab, home environment, and outdoor.
- The samples of the activities recorded are varied from 1s to several seconds.
- Different types and number of ADL and fall activities used.
- The placement of sensors and the number of sensor units attached to the human body, vary.
- Different types of sensors used such as accelerometer and gyroscope.
- Recording of the signal can be done using smartphone or external sensors.
- The sampling rate and range of the sensors vary.

The “tFall ” dataset was used in the simulation phase, which can be downloaded from [89]. The data was captured from a three-axis accelerometer sensor which was sampled at 50 Hz [90]. Each data record recorded in the dataset has a time window of 6s [90]. The ADL data are captured from real scenarios and the fall activities are simulated [90]. The following fall activities are simulated: forward falls, backward falls, left and right lateral falls, syncope, sitting on an empty chair, preventing falls or reducing the impact of fall [90]. The ADL was carried out in real conditions, where the volunteers placed the phone in the trouser pocket for seven days to record their movements [90]. The fall experiment was selected from Nourey and Kangas [90]. Table 3.1 provides information of the users in the dataset [90]. Each of the fall activities was conducted three times, which makes a total of 24 simulations per participant [90]. Falls were performed on a soft mattress in a laboratory environment where the subjects placed the smartphone in both their left and right pockets [90]. The advantage of using the “tFall ” dataset compared to the other datasets is that the ADL is not captured in a laboratory environment, and it represents subjects’ real movements in their everyday life. The disadvantage of the dataset is that the type of ADL performed is not known.

From the dataset, only nine subjects out of the ten were used in the simulation. The number of ADL and fall records for each subject is shown in the Table 3.2, with the total number of records for both fall and ADL shown in the last row of the table. The reason for simulated falls is that it is extremely hard to collect real-world elderly person fall data since 30% of the elderly population over the age of 65 years old fall at least once per year [27]. The simulated data could be biased since subjects are forced to perform activities, which are typically spontaneous [27]. The choice of the mattress to reduce the impact of the falls to protect the volunteers can reduce the overall accuracy of the system when applied to the real world [27]. The challenges found from real life data compare to the simulated data is that falls are rare and vary a lot in terms of speed and the nature of fall [44].

Table 3.1. Information of the tFall dataset

Criteria	Dataset information
Males	7
Females	3
Aged	20-42 years old (31.3 $\pm$ 8.6 years)
Body mass	54-98 kg (69.2 $\pm$ 13. Kg)
Height	1.61 to 1.84 m

The raw accelerometer data of some of the ADLs is shown in Figure 3.1. The raw accelerometer data for the different fall activities in the dataset is shown in Figure 3.2.

3.3.2 Features extraction

The feature extraction method is used to extract relevant characteristics from the dataset to allow the classifier to differentiate fall events from ADL events. Feature extraction plays an important role which contributes to the accuracy of the system. For each record in the dataset, the time index of the

Table 3.2. The number of records for each person in the tFall dataset

Subject number	Number of ADL records	Number of Fall records
1	927	51
2	912	53
3	1037	38
4	708	54
5	817	52
6	881	50
7	711	54
8	575	53
9	439	51
Total	**7007**	**456**

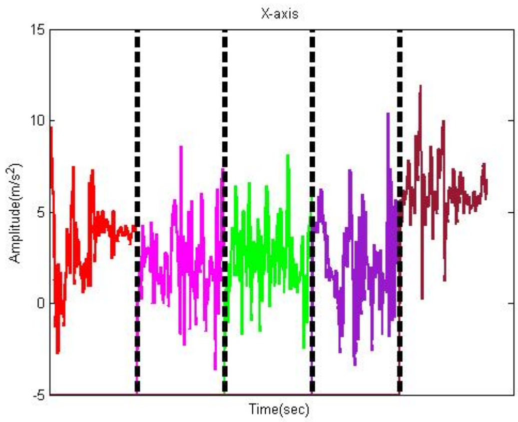
maximum peak of the SMV is found. From the time index of the SMV peak, where 2 seconds (50 Hz $\times$ 1 second = 50 samples) of data from the left and right the peak is extracted. A sliding window of the signal is captured for 101 values since the duration of the fall is about 2 seconds [91]. This reduces computational complexity when training and testing the models and machine learning algorithms [91]. Two datasets are created: the first one contains raw acceleration values, and the second one uses the statistical features. For the first dataset with the raw acceleration values, the x-axis, y-axis, and the z-axis from the accelerometer is extracted and the following vector is created,

$$Vector = [x_1, \ldots, x_{101}, y_1, \ldots, y_{101}, z_1, \ldots, z_{101}]. \tag{3.1}$$

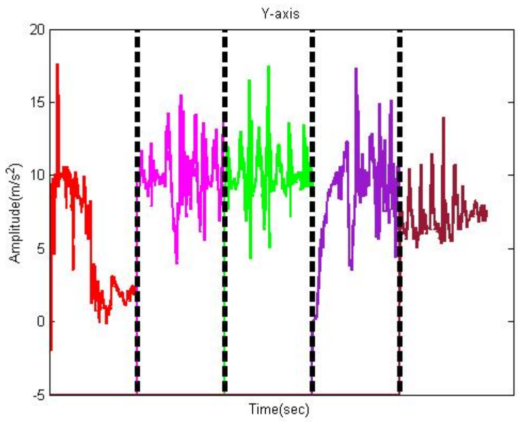
The statistical feature is extracted from the time-domain and frequency domain accelerometer data. The time-domain is the raw accelerometer data itself. Equation (3.2) gives the discrete Fourier transform (DFT) which is applied on the time-domain data, to get the data to the frequency domain, shown below [92],

$$DFT(k) = \sum_{i=0}^{N-1} d_i e^{\frac{-j2\pi ki}{N}}, k = 0, 1, \ldots, N-1, \tag{3.2}$$

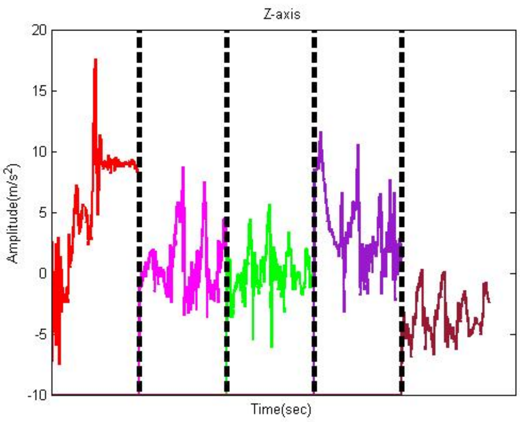
where d_i is the input data, N the size of the input data. From the raw accelerometer data, the SMV and the slope SMV are calculated. Equation (3.3) gives the formula to calculate the slope SMV,



(a) X-axis

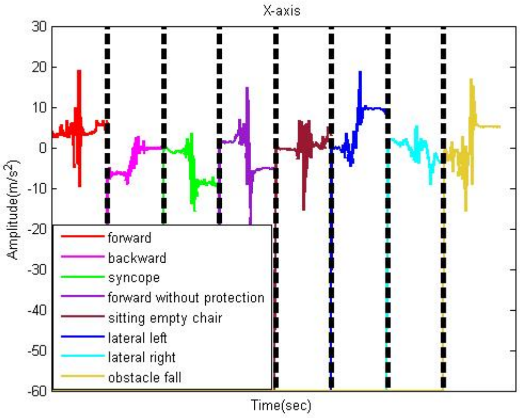


(b) Y-axis

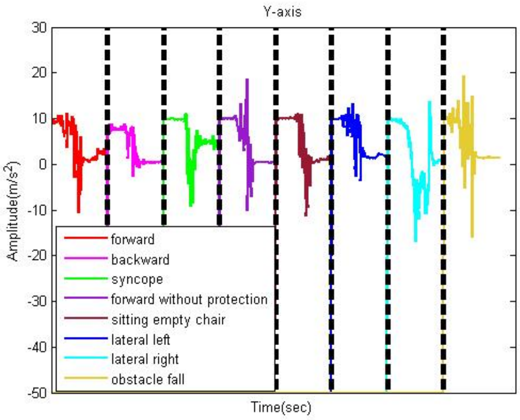


(c) Z-axis

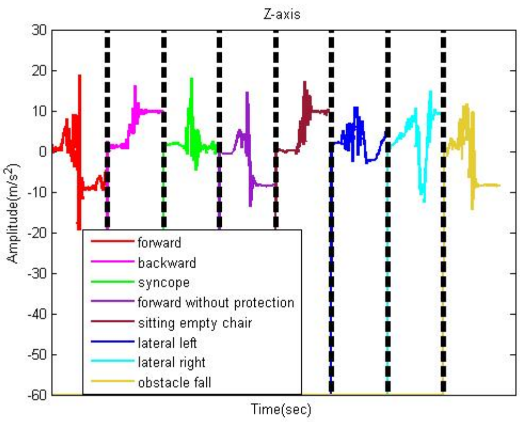
Figure 3.1. The raw x (a), y (b) and z (c) axis of the accelerometer data for ADL.



(a) X-axis



(b) Y-axis



(c) Z-axis

Figure 3.2. The raw x (a), y (b) and z (c) axis of the accelerometer data for the different fall activities.

$$slopeSMV = |SMV|_t - |SMV|_{t-1}\,, \tag{3.3}$$

where t is the sample value. The time domain data includes the raw data from the x, y, z accelerometer axes, SMV, and slope SMV. The time-domain data is also converted into the frequency domain. The second dataset will contain statistical features (shown in Table 3.3) which are extracted from both the time-domain and frequency-domain data. A total of 80 features were extracted for each record. Feature extraction of data was done in Python, using own code.

3.3.3 Features selection

Feature selection will be performed on the second dataset, where statistical features were extracted. Two different feature selection methods were applied: PCA and variance threshold thus, another two datasets were created. The PCA feature selection method creates new features from the dataset whereas variance threshold selects the best features from the dataset. The aim of PCA is to scan the data to search for patterns in order to reduce the dimensions of the dataset, with minimum loss of information [93]. The advantage of PCA is that the reduction in computational costs, the error of parameter estimation and improves the performance of the classification since the feature vector is reduced [92], [93]. The advantage of PCA is that the dataset becomes more descriptive [92]. In PCA, the whole dataset is projected into a different subspace, where class labels are not taken into consideration [93]. To perform PCA, the following steps were followed:

1. Create a dataset of d-dimensional samples, without taking into account the class labels [93].
2. Calculate the d-dimensional mean vector [93],

   $$\overline{x} = \frac{1}{n}\sum_{i=1}^{n}(x_i)\,, \tag{3.4}$$

   where x_i is the sample for feature in the dataset [93].
3. Calculate the covariance matrix Σ of the entire dataset, which is a $d \times d$ matrix where each value represents the covariance between two features [93], [94]. Equation (3.5) computes the covariance between two features [94].

   $$\sigma_{jk} = \frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)(x_{ik} - \overline{x}_k) \tag{3.5}$$

Table 3.3. Statistical formulas used to extract features from the time and frequency domain

Feature extracted	Description	Formula	Symbol description
Max	To get the max value from the input data.	$Max(X)$	X: input data
Min	To get the minimum value from the input data.	$Min(X)$	X: input data
Mean	The average of all the values in the input data.	$\frac{\Sigma_{i=1}^{n} \lvert x_i \rvert}{n}$	x_i: input data sample n: the size of the input data
Median	Finding the middle value of the input data.	$\frac{x_{n+1}}{2}$	x: input data sample n: the size of the input data
Standard deviation	Is a measure of how spread out the input data values are from the mean.	$\sqrt{\frac{\Sigma_{i=1}^{n}(x_i-\bar{x})^2}{n-1}}$	x_i: input data sample n: the size of the input data $\bar{x}$: mean
Mean absolute deviation	It is the average absolute difference between each value in a set of data, and the mean of the data set.	$\frac{\Sigma_{i=1}^{n} \lvert x_i-\bar{x} \rvert}{n}$	x_i: input data sample n: the size of the input data $\bar{x}$: mean
Skewness	Determine the symmetry of a distribution from the mean.	$\frac{1}{n\sigma^3}\Sigma_{i=1}^{n}(x_i-\bar{x})^3$	x_i: input data sample n: the size of the input data $\bar{x}$: mean σ: standard deviation
Kurtosis	Measure of the “tailedness” to determine if the input data is heavy-tailed/ light-tailed relative to a normal distribution.	$\frac{1}{n\sigma^4}\Sigma_{i=1}^{n}(x_i-\bar{x})^4$	x_i: input data sample n: the size of the input data $\bar{x}$: mean σ: standard deviation

Equation (3.6) gives the summarized matrix calculation of the covariance matrix,

$$\Sigma = \frac{1}{n-1}\left((X-\bar{x})^T(X-\bar{x})\right), \tag{3.6}$$

where $\bar{x}$ is the mean vector, and X is the entire dataset.

4. Eigendecomposition on the covariance matrix is performed; where the eigenvectors and eigenvalues are computed [93], [94].
5. The eigenvectors will form the axes of the reduced feature space [94]. The eigenvalues are used

to determine which eigenvectors to drop in order to create a smaller-dimensional feature space with minimum loss of information [94]. The eigenvector of the lowest eigenvalue that contains the least information can be dropped [94]. The eigenvalues are ranked from the highest to lowest to determine the top k eigenvectors [93], [94].

6. To determine the number of k eigenvectors to select, the explained variance is computed from the eigenvalues [94]. All the components below 85% of the cumulative explained variance are selected.
7. The projection matrix is constructed from the selected k eigenvectors, which is used to transform the data onto the new feature subspace [94]. The eigenvectors selected are used to form a $d \times k$ dimensional eigenvector matrix W (where each column represents an eigenvector) [93], [94].

Equation (3.7) transforms the new samples onto the new feature subspace using the projection matrix given by [93]:

$$y = W^T \times x, \tag{3.7}$$

where x is the $d \times 1$ dimensional vector, which represents one sample, W is the projection matrix, and y is the transformed dimensional sample onto the new feature subspace [93]. PCA was implemented in Python, using own code.

In Figure 3.3, PCA is applied to dataset 2 where the cumulative explained variance is plotted, with 15 eigenvectors selected; since it is below 85% of the cumulative explained variance. The projection matrix is constructed using the selected 15 eigenvectors.

The feature selection method used is the variance threshold (filtering feature selection method), which does not take into account the class labels, hence can be used for unsupervised learning. The variance threshold removes all low-variance features which do not meet a certain threshold. The feature selection method also removes zero-variance features. The python built-in variance threshold function was used. The advantage of PCA and variance threshold is that it allows for only ADL data records to be used. Variance threshold is applied to dataset 2, where the threshold was set to 20, thus if the feature variance is below 20, the feature is discarded. Out of the 80 features, the variance threshold feature selection method selected 46 features shown in Table 3.4 and Table 3.5. From Table 3.4 and 3.5, 69.57% of the features is from the frequency domain.

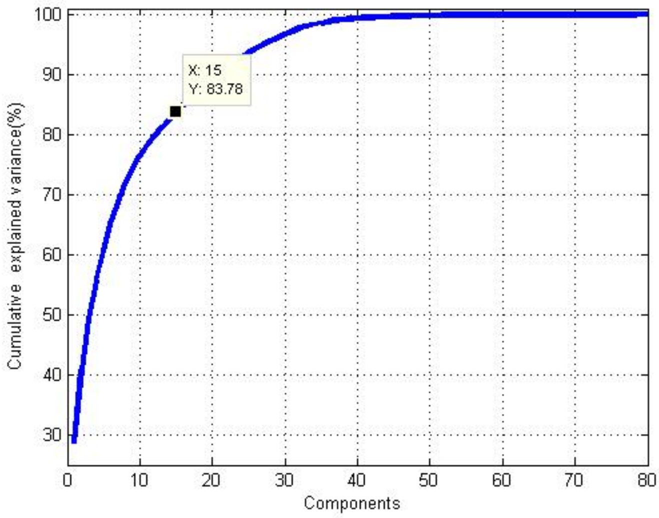


Figure 3.3. Cumulative explained variance vs number of components using dataset 2 features.

3.3.4 Dataset

From the feature extraction and feature selection, four datasets were created which are summarized in the Table 3.6. The purpose is to determine which type of data will provide the best performance and impact of the datasets on the performance of the fall detection algorithm. Each dataset from Table 3.6, will contain 9 different records database (for each person in the *tFall* dataset).

3.3.5 Classifiers

Different supervised and unsupervised machine learning algorithms were tested to determine the classifier that achieved the best performance. The following unsupervised machine learning algorithms were selected: Angle based outlier degree (ABOD), nearest neighbour (NN), one-class SVM and isolation forest (ISF). The support vector machine (SVM) and k- nearest neighbour (k-NN) were used for the supervised machine learning algorithms.

3.3.5.1 Angle based outlier degree

The increase in the dimensionality of feature space causes the relative contrast between distances to become smaller [95], [96]. The idea of the neighbourhood becomes useless if the dimensionality of feature space increases [95]. This is resolved by the use of angles rather than distances in high dimensional space since angles are more stable [95], [96]. The basic idea of ABOD is that outliers are

Table 3.4. Features selected using variance threshold.

Feature	Data	Domain	Feature	Data	Domain
Max	SMV	Time	Median	X-axis accelerometer	Frequency
Min	SMV	Time	Standard deviation	X-axis accelerometer	Frequency
Max	X-axis accelerometer	Time	Mean absolute deviation	X-axis accelerometer	Frequency
Min	X-axis accelerometer	Time	Kurtosis	X-axis accelerometer	Frequency
Mean	X-axis accelerometer	Time	Max	Y-axis accelerometer	Frequency
Median	X-axis accelerometer	Time	Min	Y-axis accelerometer	Frequency
Max	Y-axis accelerometer	Time	Mean	Y-axis accelerometer	Frequency
Min	Y-axis accelerometer	Time	Median	Y-axis accelerometer	Frequency
Mean	Y-axis accelerometer	Time	Standard deviation	Y-axis accelerometer	Frequency
Median	Y-axis accelerometer	Time	Mean absolute deviation	Y-axis accelerometer	Frequency
Max	Z-axis accelerometer	Time	Kurtosis	Y-axis accelerometer	Frequency

located at the end of the data distribution and the normal points are in the centre of the data distribution [95]. The advantage of this method is that it does not need any parameter selection [95]. The distance is used only to normalize the results. [95]. To detect if the incoming data is an outlier, the variance of the angles between the different vectors in the dataset is calculated [95]. An outlier is detected if the angle is small [95]. Figure 3.4 shows the input data point (A), the angle between $\overrightarrow{AB}$ and $\overrightarrow{AC}$ for any two B, C data point from dataset (D).

Table 3.5. Features selected using variance threshold continued from Table 3.4.

Feature	Data	Domain	Feature	Data	Domain
Min	Z-axis accelerometer	Time	Max	Z-axis accelerometer	Frequency
Min	Slope SMV	Time	Min	Z-axis accelerometer	Frequency
Kurtosis	Slope SMV	Time	Mean	Z-axis accelerometer	Frequency
Max	SMV	Frequency	Median	Z-axis accelerometer	Frequency
Min	SMV	Frequency	Standard deviation	Z-axis accelerometer	Frequency
Mean	SMV	Frequency	Mean absolute deviation	Z-axis accelerometer	Frequency
Median	SMV	Frequency	Kurtosis	Z-axis accelerometer	Frequency
Standard deviation	SMV	Frequency	Max	Slope SMV	Frequency
Mean absolute deviation	SMV	Frequency	Mean	Slope SMV	Frequency
Max	X-axis accelerometer	Frequency	Median	Slope SMV	Frequency
Min	X-axis accelerometer	Frequency	Standard deviation	Slope SMV	Frequency
Mean	X-axis accelerometer	Frequency	Mean absolute deviation	Slope SMV	Frequency

The ABOD factor is calculated as the variance angles between the differences vectors of the A to all pair of points in the dataset (D) weighted by the distance of the points, which is given by [96],

$$ABOD\left(\vec{A}\right) = VARIANCE_{\vec{B},\vec{C}\in D}\left(\frac{\left\langle \overrightarrow{AB},\overrightarrow{AC}\right\rangle}{\left\|\overrightarrow{AB}\right\|^{2}.\left\|\overrightarrow{AC}\right\|^{2}}\right). \tag{3.8}$$

Table 3.6. Datasets used in this study

Dataset number	Description
1	Raw x, y, and z axes values from the accelerometer.
2	The following statistical features from Table 3.3 are extracted from both time and frequency domain using the following data: raw x, y, and z axes values from the accelerometer, SMV and slope SMV.
3	New features created using PCA on the dataset 2.
4	Features are selected using variance threshold method on dataset 2. The features selected are shown in Table 3.4 and Table 3.5.

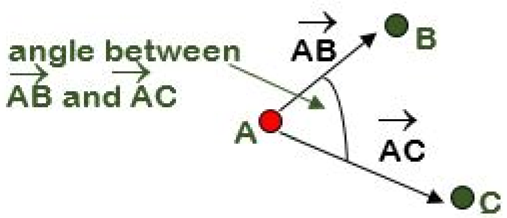


Figure 3.4. Example of the ABOD model calculation of the angle between two points.

The reason for dividing by distance is for the low dimensional datasets where angles are less reliable [95], [96]. The threshold value of the classifier is determined from the ROC curve. ABOD was implemented in Python, using own code.

3.3.5.2 Isolation forest

Isolation forest outperforms one-class support SVM in terms of area under the curve, processing time, and it is not affected by masking and swamping effects [97]. One-class SVM trained using normal instances then identified anomalies that do not match the normal instances [97]. The problem with one-class SVM is that it is a by-product of SVM, which was designed for another purpose instead of anomaly detection [97]. This can result in two problems:

- The algorithms are not optimized to detect outliers which result in many false alarms [97].
- They are limited to low dimensional data and small data size [97].

The advantage of ISF is that it does not use distance to detect outliers, which removes the high computational cost associated with distance calculations [97]. A binary tree structure called isolation tree is designed to effectively isolate instances [97]. The basic idea is that outlier is isolated closer to the root of an isolation tree; whereas the normal points are isolated at the bottom end of an isolation tree [97]. An isolation forest builds an ensemble of isolation trees for the dataset, where outliers are those instances which have a small average path length on the isolation tree [97]. Isolation forest takes in two parameters: the number of trees to build and subsampling size [97]. The isolation tree is generated by randomly selecting a feature from the sample of instances, which is split by randomly selecting a value between the maximum and minimum values of the selected feature [97]. To detect an outlier, a single path length $h(x)$ is calculated by counting the number of edges from the root node to an external node as the incoming instance goes through the isolation tree [97]. When an instance reaches a predefined height limit, the return value is the edges plus adjustment [97]. The adjustment takes into consideration estimating an average path length of a random sub-tree which could be constructed using data of a size beyond the tree height limit [97]. The average path length of unsuccessful searches in the binary search tree is given [97],

$$c(\psi) = \left\{ \begin{array}{l} 2H(\psi-1) - \dfrac{2(\psi-1)}{n} \; for \; \psi > 2, \\ 1 \; for \; \psi = 2, \\ 0 \; otherwise \end{array} \right\} \tag{3.9}$$

where ψ is the set of instances and $H(i)$ is the harmonic number. The harmonic number is estimated given by [97],

$$H(i) = ln(i) + 0.5772156649. \tag{3.10}$$

The outlier score (s of an input sample (x) is calculated, when $h(x)$ for each isolation tree of ensemble is determined by [97],

$$s(x,\psi) = 2^{-\dfrac{E(h(x))}{c(\psi)}}, \tag{3.11}$$

where $E(h(x))$ is the average of $h(x)$ from a collection of iTrees and $c(\psi)$, is the average path length of unsuccessful searches in the binary search tree. Based on the outlier score (s), the following criteria is derived [97]:

1. If s is close to 1, the instance is an outlier.
2. If s is smaller than 0.5, the instance is regarded as a normal instance.
3. If s is around ~ 0.5, then the instance does not contain any distinct outlier.

There are two parameters that need to be set: the number of trees and the number of samples. The number of trees is set to the recommended default value of 100. The number of samples is set to the size of the training data size. ISF was implemented in Python, using own code.

3.3.5.3 Nearest Neighbour

The basic idea is to determine whether the input data point is an outlier, based on the distance to its neighbours [95]. The assumption is that normal data points have a dense neighbourhood where outliers are further away from neighbours or have a less dense neighbourhood [84], [95]. The basic concept of NN is to allocate the incoming record to the class that has a record closest to the incoming record [39]. The Euclidean distance is computed for the incoming record with each of the stored records, where the minimum distance between the incoming record and stored record is used [18], [84]. Given a set of N instances from a dataset $\{D^j, j \ldots N\}$, the nearest neighbour distance of the new record, A, was given by [84],

$$d_{NN}(A) = min_j d(A, D^j), \tag{3.12}$$

where d is the Euclidean distance between record A and record D^j. If the minimum distance is higher than a threshold value, the incoming record is considered an anomaly [18], [84]. The advantage is that it is simple and easy to implement [84]. The disadvantage of d_{NN} is that the performance is low when the data has regions of varying densities [84]. The threshold value of the classifier is determined from the ROC curve. The NN was implemented in Python, using own code.

3.3.5.4 One-class SVM

The one-class SVM can be seen as two normal SVM classes in Figure 3.5, where all the training data belongs to $+1$ class, and the origin is the outlier class which belongs to the -1 class [98].

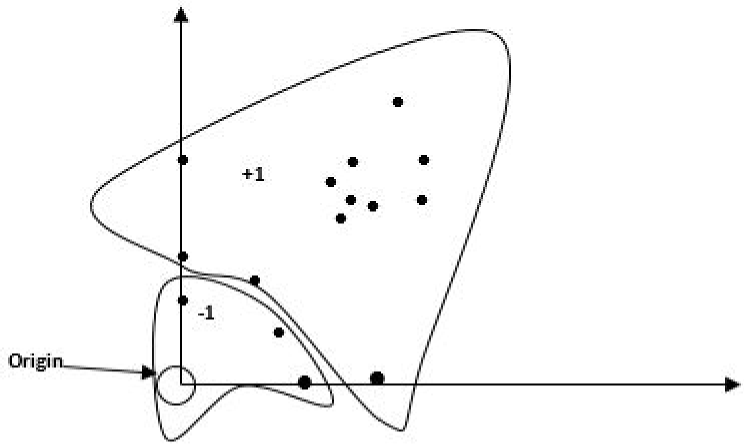


Figure 3.5. One-class SVM.

One-class SVM converts the data to a feature space which is surrounded by a hypersphere and it searches for the appropriate hyperplane that splits a portion of the input data from the rest of the data by the sign of the distance to the hyperplane ($f(C) > 0$ or $f(C) < 0$) [84]. The kernel function is responsible for mapping the data into a feature space [99]. The classifier makes use of hyper-plane as a decision boundary to classify the binary data, which is given by [98],

$$f(x) = < w, x > + b, \tag{3.13}$$

where x is the input data, w is the normal vector and b is the bias term [98], [84]. If $f(x) < 0$, the input data is an outlier, and if $f(x) > 0$, the input data is a normal instance [98]. The advantage of one-class SVM is that it describes the data in a flexible way since it does not need to ensure that the data follows a certain distribution [84]. The parameters for one-class SVM are shown in Table 3.7, and the parameters will be determined using grid search. One-class SVM was implemented in Python, using scikit-learn library.

3.3.5.5 Support Vector machine

The SVM uses a kernel trick where it transforms the inputs which are features extracted into a higher dimensional space using a non-linear mapping in which an optimum hyperplane is found separating

Table 3.7. One-class SVM parameters

Parameter	Description
Kernel	Determine what type of kernel to be used. The kernel type includes the following linear, polynomial, sigmoid and the most popular one being the Gaussian radial base function.
nu	This control the number of support vectors and training errors. An upper bound on the fraction of training errors and a lower bound on the fraction of support vectors. Should be in the interval (0,1].
gamma	The kernel coefficient if the following kernel types are selected: radial base function, polynomial and sigmoid.

two classes from a given training dataset [18], [24], [61]. A hyperplane is used to separate the two classes by creating a decision boundary (maximum margin hyperplane) [1]. Optimization of separating hyperplane is done by maximizing the distance between the hyperplane and the nearest data points [8], [24]. The maximum margin hyperplane is learnt based on the support vectors, which the classifier used to classify the new feature vector [1], [18]. The Radial Basis Functions (RBF) kernel is the most popular kernel used in SVM [18]. The classifier makes use of hyperplane as a decision boundary to classify the binary data, which is given by [1], [18],

$$f(x) = \sum_i \alpha_i K(x_i, x) + b, \tag{3.14}$$

where b is the bias term, $K(x_i, x)$ is the radial base function. Equation (3.15), gives the formula for the radial base function used.

$$K(x_i, x) = exp(-\gamma ||x - x^{'}||^2). \tag{3.15}$$

The parameters for one-class SVM are shown in Table 3.8, and following parameters determined from a grid search are C, kernel and gamma. SVM was implemented in Python, using the scikit-learn library.

Table 3.8. SVM parameters

Parameter	Description
C	Penalty parameter C of the error term
kernel	Radial base function
Gamma	The kernel coefficient if the following kernel type is selected: radial base function, polynomial and sigmoid.
Class weight	To compensate for uneven sample

3.3.5.6 k-Nearest Neighbour

The k-NN, also known as a lazy learner, classifies a new feature vector based on the closeness of the other training feature vectors [1], [19], [22]. Each time a new feature vector is inputted into the classifier, all the training feature vector sets are compared to the new feature vector in terms of a Euclidean distance. From the Euclidean distance, the shortest distance will be determined, the centroid the feature vector has joined and in what class it lies [1], [3], [22]. The value k determines the number of centroids that are available for each class. Special attention should be applied to determine the value of k. If a smaller value k is selected, the variances increase and the results are less stable and a large k value will result in an increase in biasing which will reduce the sensitivity [19]. The disadvantage of the classifier is that the time complexity increases as the training data increases. The number of neighbours (k) to use is determined using grid search and the threshold value of the classifier is determined from the ROC curve. The k-NN was implemented in Python, using own code.

3.3.6 Models

The different types of adaptive or personalized models will be investigated. The basic classification model, model 1 (M1), for fall detection, which trains the classifier with other people's data except the user's data is shown in Figure 3.6. In order to adapt the model, three adaptive models are designed and shown in Figures 3.7 - 3.9. In Figure 3.7, model 2 (M2), the classifier is retrained with the input data if the classifier predicted that the input data is an ADL. This type of adaptive model will learn to predict the user movements better. In Figure 3.8, model 3 (M3), the classifier is retrained with the input data if the classifier incorrectly predicted that the ADL input data is a fall activity, hence a false positive. This type of adaptive model will reduce the number of false alarms and learn a new ADL if the ADL activity resembles a fall activity. Most studies trained the classifier with limited ADL and if the user performed a certain daily activity which is not learnt by the classifier, the classifier will detect it as a fall activity. This will result in high false alarms and irritation among the users. The false alarms can be reduced by allowing the system to retrain false positives. In Figure 3.9, model 4 (M4), is the combination of model M2 and model M3. The reason for having different models is to show the overall improvement in terms of performance by using model M4. Model M1 is tested on all the machine learning algorithms. The supervised machine learning algorithms using model M1 are known as supervised classifiers (SC). The following models M2, M3 and M4 are applied only on unsupervised classifiers. Supervised machine learning algorithms cannot be used since they require a balanced dataset. An unbalanced dataset can cause the supervised machine learning algorithm to fail in distinguishing characteristics of the data. This can result in low accuracies and their prediction tends to favour the majority class [5]. The unbalanced dataset can be compensated by applying a cost sensitive-classification but the problem is that the cost is unknown and difficult to compute [5].

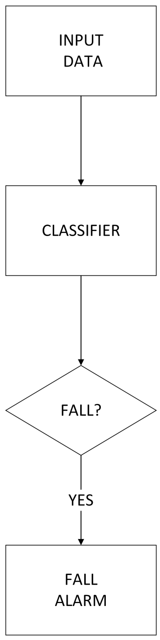


Figure 3.6. Model 1 algorithm.

3.3.7 Evaluation

To determine the thresholds for the ABOD and NN, a ROC curve is plotted, with different parameters or threshold value. To select the threshold or parameter, the maximum geometric mean of the sensitivity

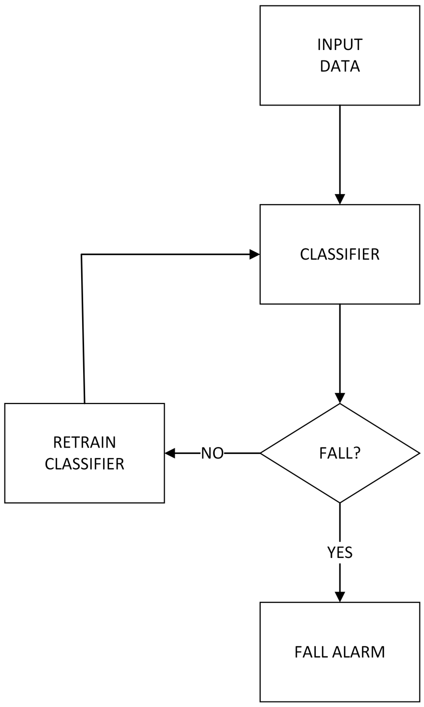


Figure 3.7. Model 2 algorithm.

and specificity is selected from the curve. The datasets in Table 3.6 is applied to each model and on the different classifiers that are assigned to each model. The different dataset was created to investigate whether extracting features affect the performance of the system. To train and test M1 and the SC is shown in Figure 3.10 where the classifiers are trained with all the data except the person who is going to be tested. The process is done 9 times since there are 9 people data in the dataset. The evaluation of model M2, M3, and M4 is shown in Figure 3.11, where a 10-fold cross-validation is used, which divides the entire dataset into 10 parts, and takes one part for testing each time and the rest of the parts for training. This ensures that the entire dataset is tested. The cross-validation is applied only on person p ADL records, where one-part is used for validation and the rest for training. From each person in the dataset except for p, 50 random ADL records are extracted which will create an initial training dataset of 400 ADL records. The reason for not using all the records from each person except for p, is to determine whether personalizing the system will improve the performance of the fall detection system. This is repeated 1000 times because random ADL records were extracted to create the initial training dataset. The following classifiers were tested on model M1, M2, M3, and M4: ABOD, NN, ISF, and one-class SVM. The following classifiers were tested on SC: SVM and k-NN.

To evaluate performance, the following metrics will be used; geometric mean given by (2.6), sensitivity

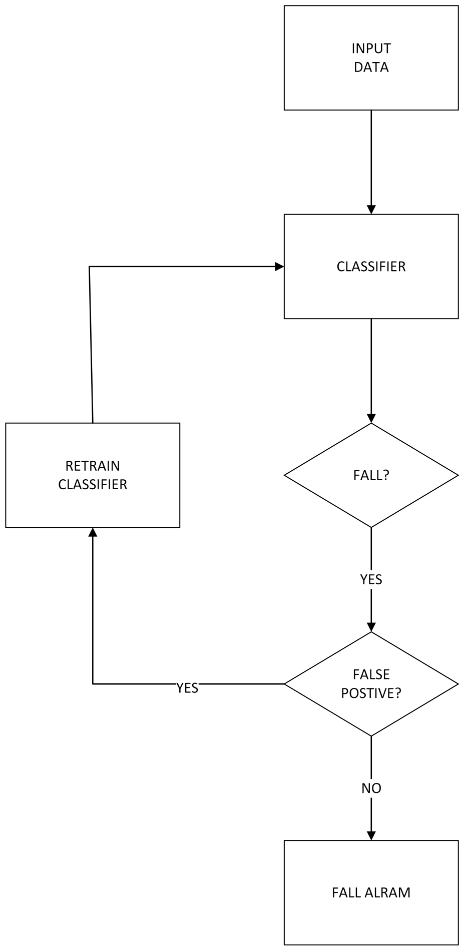


Figure 3.8. Model 3 algorithm.

given by (2.1), specificity given by (2.3), and F_1-measure for both ADL and fall activities given by (2.5).

3.4 EXPERIMENTAL

The aim of this experiment is to validate the system to determine if the most common ADL can be detected, since the *tFall* dataset used does not mention the type of ADL. The model, dataset, and algorithm with the best performance indicators from the simulation phase will be implemented on the Android device. The data is sampled at 50 Hz, which is sufficient to detect if a fall has occurred [88]. Since an Android smartphone device is used, which is a battery-powered device, this requires low computational complexity algorithms [87]. To reduce the processing power and computation time, the following steps need to be taken before the classifier detects if an actual fall has occurred. The

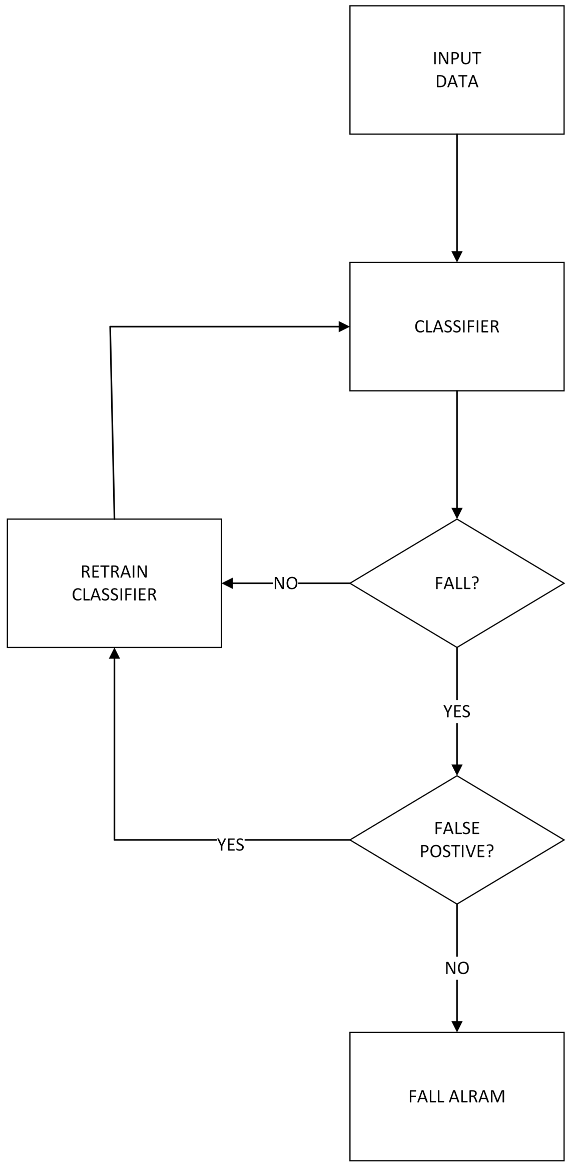


Figure 3.9. Model 4 algorithm.

first step is to detect if a possible fall has occurred if the SMV is greater than 14.7. The second step is to detect if the angle of the smartphone is greater than 55 degrees, which indicates a lying state. The angle is calculated using only the x and y-axis of the accelerometer. The angle calculation is shown in Algorithm 1. If the second step is met, the best model and classifier from the simulation phase is used to detect if the actual fall has occurred. The final algorithm implemented on the Android smartphone is shown in Figure 3.12.

Young and healthy adult subjects were used to test the fall detection system since it is very difficult to test the falls on an elderly person. The following move-sets in Table 3.9 will be tested on four adults.

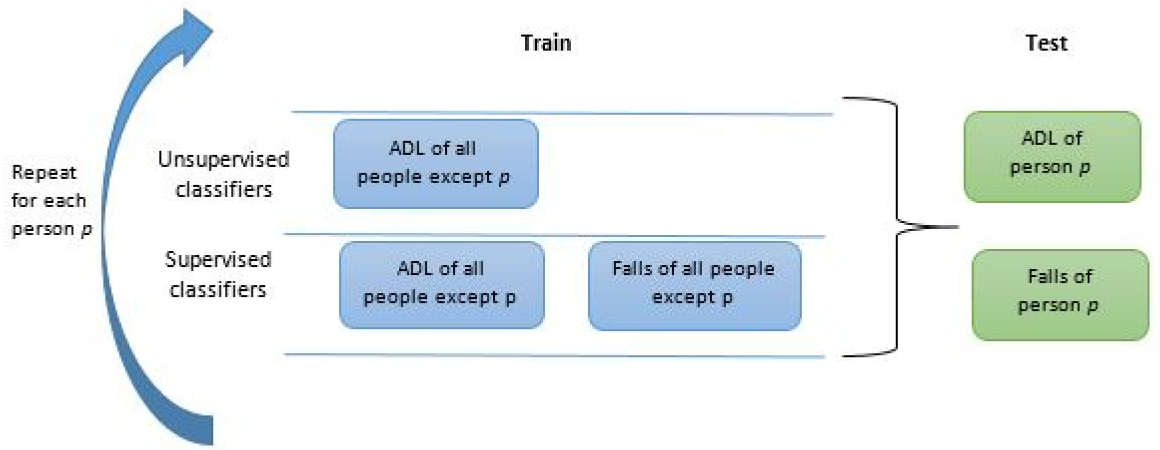


Figure 3.10. The method used to train and evaluate SC and M1.

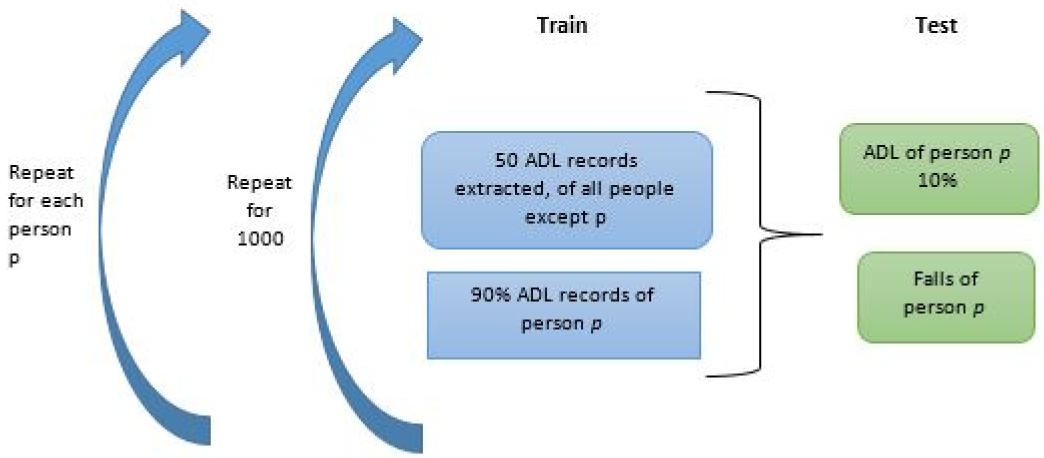


Figure 3.11. The method used to train and evaluate M2, M3, and M4.

Each move-set will be performed three times, thus every subject will perform 39 ADL move-sets and 21 fall move-sets which brings the total to 60 move-sets for each subject. The subjects will be three men and one woman, ages of 24±2, body mass between 60 - 90kg, and of a height between 1.5 - 1.81m. The user will perform these experiments on a 20cm soft mattress wearing a helmet, wrist and knee guards. To evaluate performance, the following metrics will be used, geometric mean given by (2.6), sensitivity given by (2.1), specificity given by (2.3), and F_1-measure for both ADL and fall activities given by (2.5).

Algorithm 1 Algorithm for angle calculation where x is the x-axis and y is the y-axis from the accelerometer.

if $y \geq 0$ **then**
  if $x \geq 0$ **then**
    $Angle = 90 - tan^{-1}(\frac{y}{x}) \times (\frac{180}{\pi})$
  else
    $Angle = 90 + tan^{-1}(\frac{y}{x}) \times (\frac{180}{\pi})$
  end if
else
  if $x \leq 0$ **then**
    $Angle = 90 - tan^{-1}(\frac{y}{x}) \times (\frac{180}{\pi})$
  else
    $Angle = 90 + tan^{-1}(\frac{y}{x}) \times (\frac{180}{\pi})$
  end if
end if

3.5 CHAPTER SUMMARY

In this chapter, the method of realizing a personalized fall detection system was explored. The different features extracted, machine learning algorithms and personalized models were explained in detail, along with the performance evaluators used to analyse them. The algorithm used on the Android smartphone was explained, which is created using the best features, machine learning algorithm and personalized models. The experimental setup for the Android application was discussed. In Chapter 4, the results of the different features extracted, machine learning algorithms and personalized are captured and compared to one other using the public dataset. The results of the Android application are tested on people and the results of the experiment are recorded.

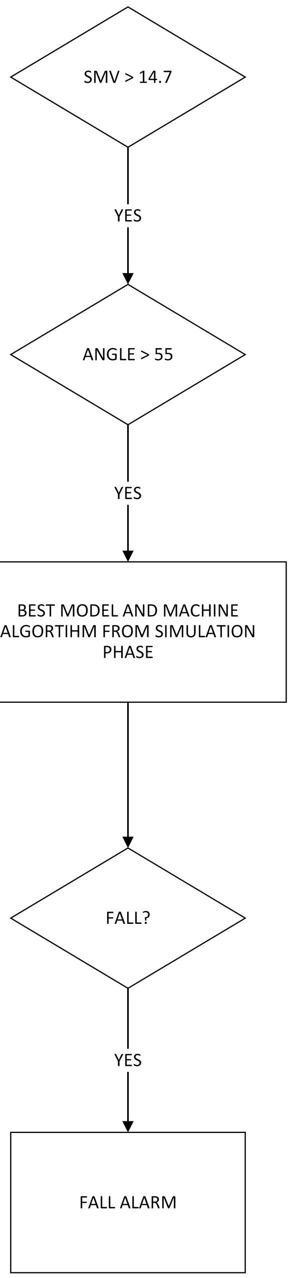


Figure 3.12. Algorithm used on the Android application for detecting falls.

Table 3.9. ADL and fall activities movements list.

ADL's experiments			Fall experiments		
#	**Label**	**Description**	**#**	**Label**	**Description**
1	Walking fw	Walking forward	41	Front lying	From a standing position, going forward to the ground
2	Walking bw	Walking backward	42	Back lying	From a standing position, going backward to the ground
3	Jogging	Running	43	Rolling out bed	From a lying position, rolling out, ending on the ground
4	Stairs up	Going up the stairs (3 stairs)	44	Back sitting	From a standing position, to the ground while trying to sit on a chair
5	Stairs down	Going down the stairs (3 stairs)	45	Front knees lying	From standing position, to knees, to lying on the ground
6	Sit chair	Sitting on a chair	46	Right side	From standing position, going down on the ground, ending in right lateral position
7	Sit sofa	Sitting on a sofa	47	Left side	From standing position, going down on the ground, ending in left lateral position
8	Sit bed	Sitting on a bed			
9	Lying bed	From standing to lying position on a bed			
10	Pick object	Bending down to pick up an object on the floor			
11	Reach object	Standing on the tip of the toe to reach for an object in the cupboard			
12	Cough	Sneezing			
13	Jumping	Continuous jumping			

CHAPTER 4 RESULTS

4.1 CHAPTER OVERVIEW

In this chapter, the results of the different features extracted, machine learning algorithms and personalized models described in Chapter 3 are presented and compared using the public dataset. The metrics for measuring the system discussed in Section 3.3.7 were used to evaluate the system. In Section 4.2, the results of the simulation are recorded for the different feature extraction datasets, for every machine learning algorithm and personalized model. The best features and machine learning algorithms for each model are recorded. The models are compared against one another. In Section 4.3, experimental sections the Android application is presented and evaluated by performing the experiments mentioned in Section 3.4.

4.2 SIMULATION

4.2.1 Datasets

Unsupervised algorithms are applied to the four different models using the different datasets from Table 3.6. Figure 4.1, presents the comparison between unsupervised machine learning algorithms based on the geometric-mean for the four models using dataset 1. The average geometric-mean of the different subjects was taken and plotted. The geometric-mean increases from model M1 to model M4, with model M2 and model M3 swapping positions for having the highest geometric-mean. The ABOD classifier achieved the best results among the unsupervised machine learning algorithm using model M1, with the one-class SVM performing the worst. For model M2, the NN classifier performed the best and one-class SVM performed the worst. For model M3, the ABOD classifier performed the best, although the difference with the NN classifier is not statistically significant, and the one-class SVM performed the worst. For model M4, the ABOD classifier performed the best, although the difference with the NN is not statistically significant, and the one-class SVM performed the worst.

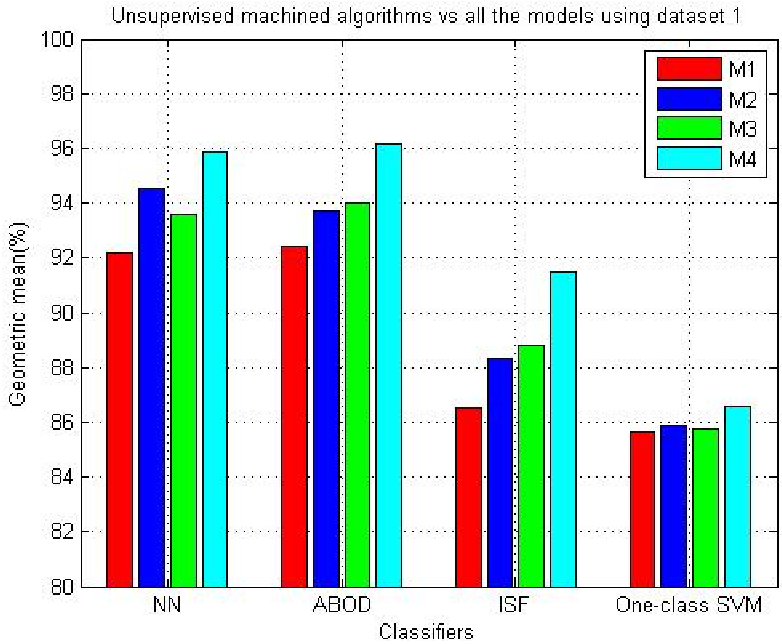


Figure 4.1. Geometric-mean for all the unsupervised machine learning algorithms for the four models using dataset 1.

To analyze the effect of incorporating ADL and false positive data into the system, the difference in the average geometric-mean of the different subjects between model M4 and model M1 is plotted in Figure 4.2. All the classifiers showed a drastic improvement, with the ISF classifier having the biggest increase in performance, and the one-class SVM having the least increase in performance.

Figure 4.3, presents the comparison between unsupervised machine learning algorithms based on the geometric-mean, for the four models using dataset 2. The average geometric-mean of the different subjects was taken and plotted. The geometric-mean increases from model M1 to model M4. The figure illustrates that the ABOD classifier achieved the best performance for all the models, and the ISF classifier achieved the worst results for all the models in dataset 2. To analyze the effect of incorporating ADL and false positive data into the system, the difference in the average geometric-mean of the different subjects between model M4 and model M1 is plotted in Figure 4.4. All the classifiers showed an improvement with the ISF classifier having the biggest increase in performance and the ABOD classifier having the least increase in performance.

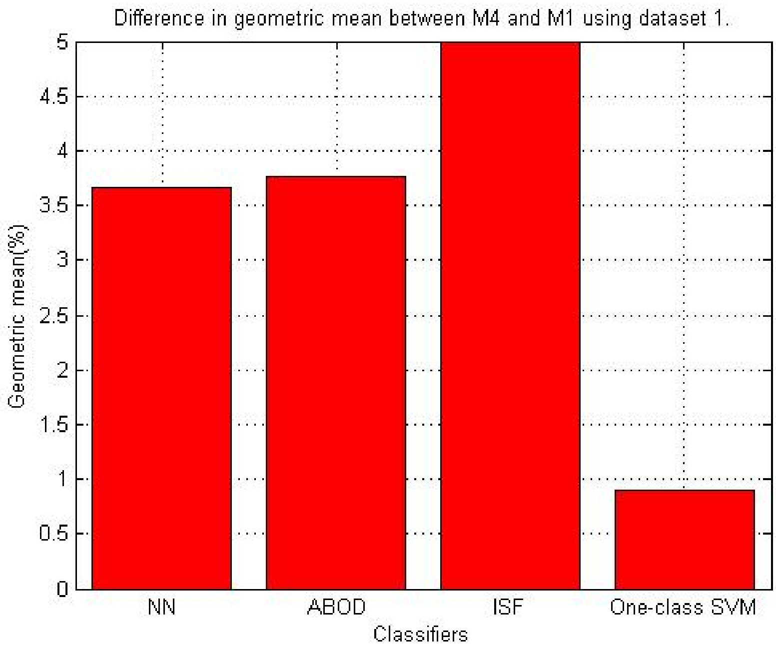


Figure 4.2. The difference in geometric-mean between M4 and M1 for the unsupervised machine algorithms using dataset 1.

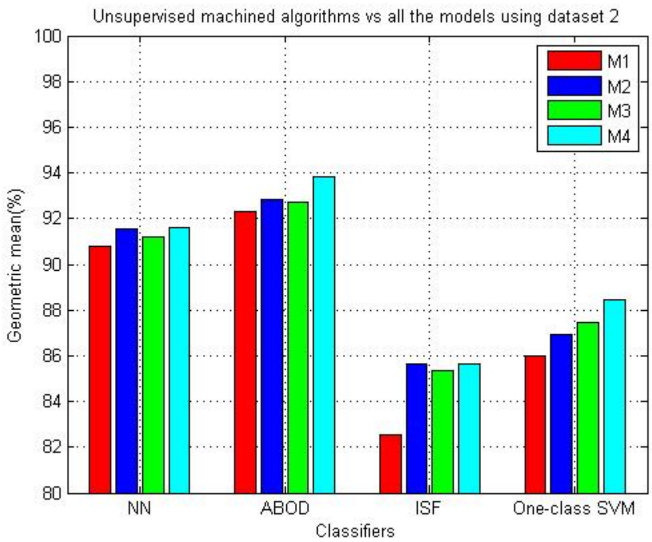


Figure 4.3. Geometric-mean for all the unsupervised machine learning algorithms for the four models using dataset 2.

Figure 4.5, presents the comparison between unsupervised machine learning algorithms based on the geometric-mean, for the four models using dataset 3. The average geometric-mean of the different subjects was taken and plotted. The geometric-mean increases from model M1 to model M4. The

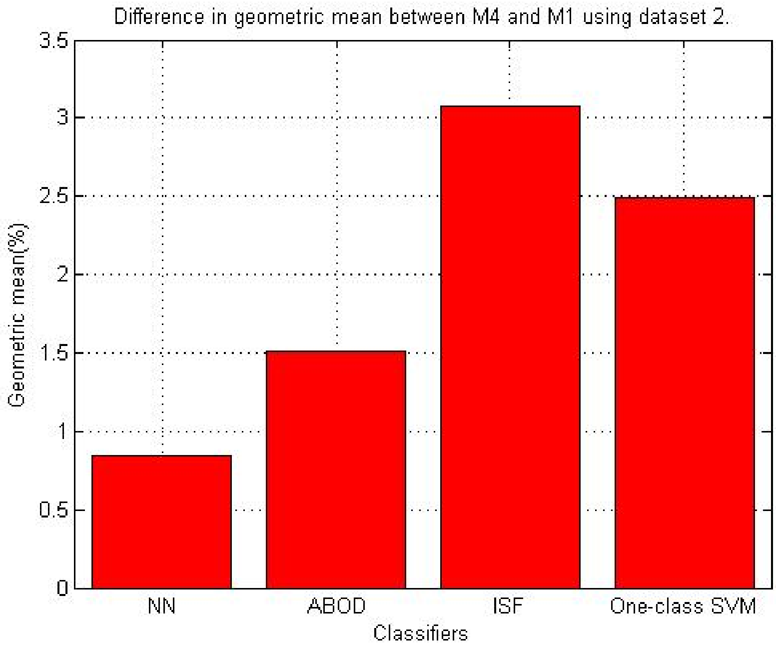


Figure 4.4. The difference in geometric-mean between M4 and M1 for the unsupervised machine algorithms using dataset 2.

ABOD classifier achieved the best performance for all the models and the ISF classifier achieved the worst results for all the models in dataset 3. To analyze the effect of incorporating ADL and false positive data into the system, the difference in the average geometric-mean of the different subjects between model M4 and model M1 is plotted in Figure 4.6. All the classifiers showed an improvement, with one-class SVM having the biggest increase in performance and the ABOD classifier having the least increase in performance.

Figure 4.7 presents the comparison between unsupervised machine learning algorithms based on the geometric-mean for the four models using dataset 4. The average geometric-mean of the different subjects was taken and plotted. The geometric-mean increases from model M1 to model M4. The ABOD classifier achieved the best performance for all the models, and the ISF classifier achieved the worst results for all the models in dataset 4. To analyze the effect of incorporating ADL and false positive data into the system, the difference in the average geometric-mean of the different subjects between model M4 and model M1 is plotted in Figure 4.8. All the classifiers showed an improvement, with one-class SVM having the biggest increase in performance, and the ABOD classifier having the least increase in performance.

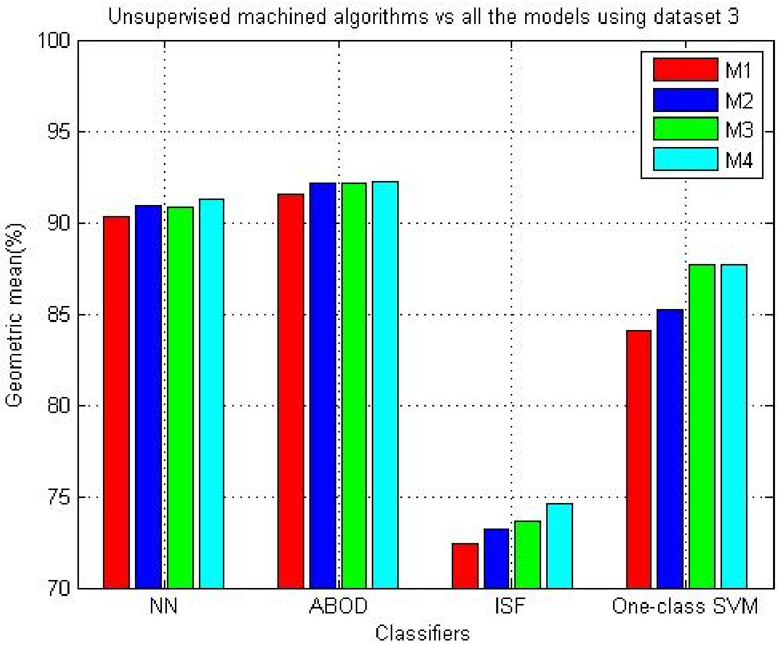


Figure 4.5. Geometric-mean for all the unsupervised machine learning algorithms for the four models using dataset 3.

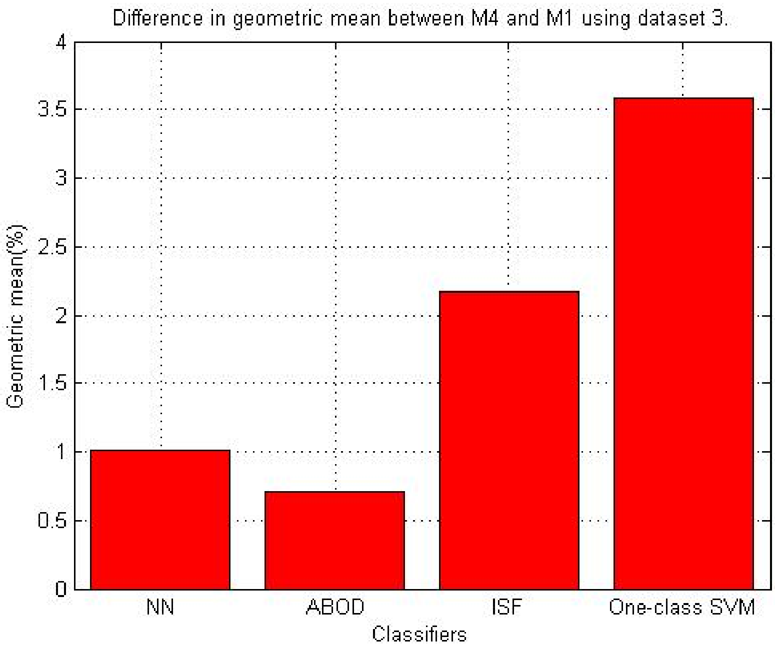


Figure 4.6. The difference in geometric-mean between M4 and M1 for the unsupervised machine algorithms using dataset 3.

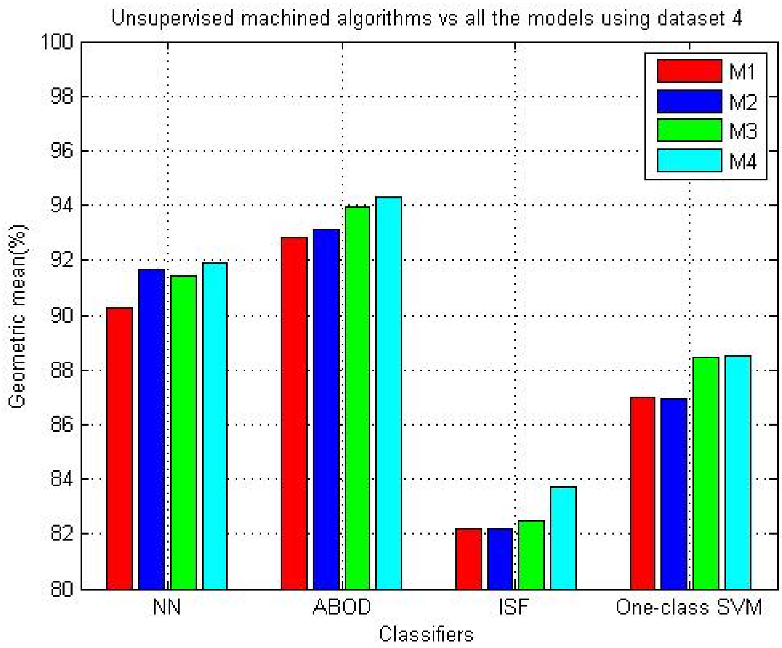


Figure 4.7. Geometric-mean for all the unsupervised machine learning algorithms for the four models using dataset 4.

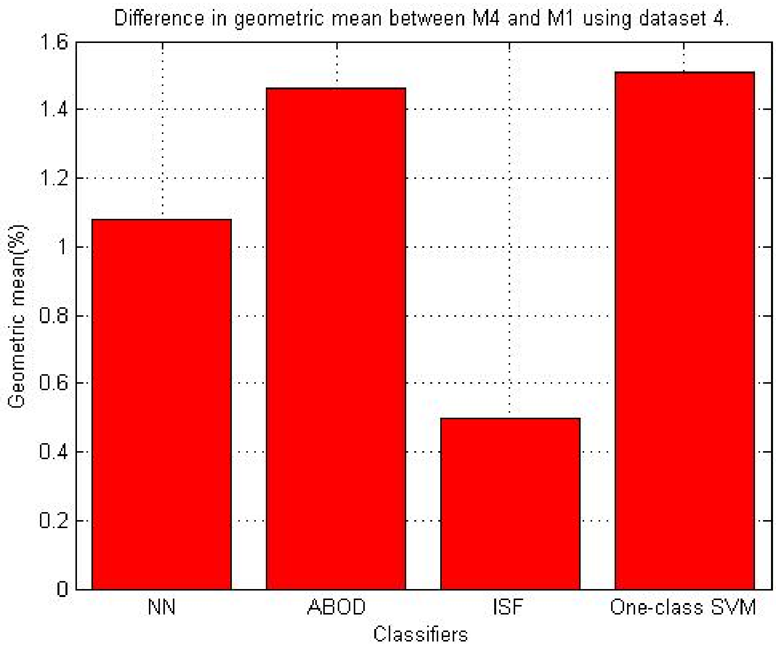


Figure 4.8. The difference in geometric-mean between M4 and M1 for the unsupervised machine algorithms using dataset 4.

Table 4.1 shows the best classifier for each dataset using model M1. The best classifier is ABOD using dataset 4, and the worst is ABOD using dataset 3. Table 4.2 shows the best classifier for each dataset using model M2. The best classifier is NN from dataset 1 and the worst is ABOD using dataset 3. Table 4.3 shows the best classifier for each dataset using model M3. The best classifier is ABOD from dataset 1 and the worst is ABOD using dataset 3. Table 4.4 shows the best classifier for each dataset using M4. The best classifier is ABOD using dataset 1 and the worst is also ABOD using dataset 3.

Figure 4.9, presents the comparison between the two SC algorithms SVM and k-NN based on the geometric-mean using all of the datasets from Table 3.6. The average geometric-mean of the different subjects was taken and plotted. For dataset 1, the k-NN classifier achieved better results, although the difference with the SVM classifier is not statistically significant. For dataset 2 $-$ dataset 4, the SVM classifier achieved better performance. The best performance across all the dataset is the SVM classifier using dataset 4.

Table 4.1. The best classifier for each dataset using M1

Dataset	**Classifier**	**Geometric-mean (%)**
1	ABOD	92.40
2	ABOD	92.32
3	ABOD	91.57
4	ABOD	92.85

Table 4.2. The best classifier for each dataset using M2

Dataset	**Classifier**	**Geometric-mean (%)**
1	NN	94.53
2	ABOD	92.85
3	ABOD	92.12
4	ABOD	93.14

Table 4.3. The best classifier for each dataset using M3

Dataset	Classifier	Geometric-mean (%)
1	ABOD	94.03
2	ABOD	92.74
3	ABOD	92.13
4	ABOD	93.97

Table 4.4. The best classifier for each dataset using M4

Dataset	Classifier	Geometric-mean (%)
1	ABOD	96.17
2	ABOD	92.83
3	ABOD	92.28
4	ABOD	94.73

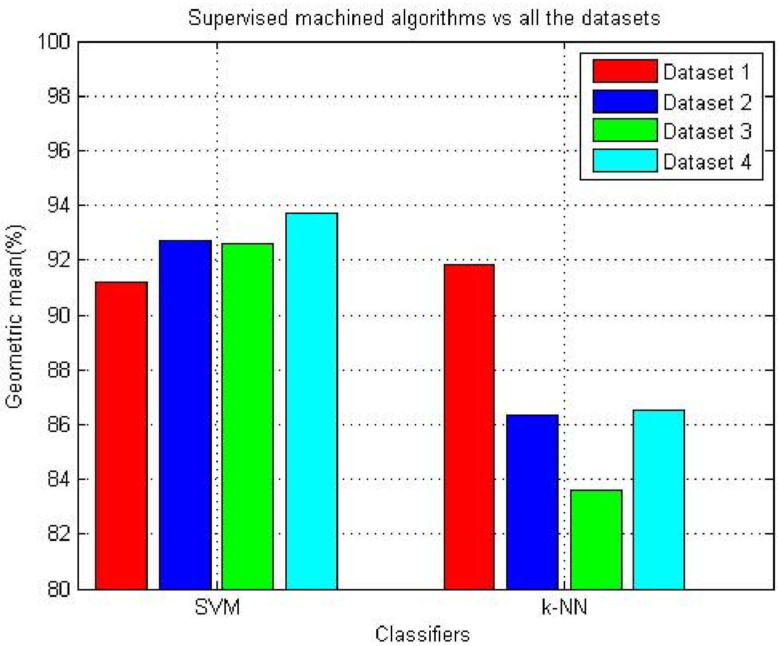


Figure 4.9. Geometric-mean for all the supervised machine learning algorithms using datasets 1 to 4.

The values of SE, SP, their geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities values are recorded for the SC model using SVM classifier and dataset 4 is shown in Table 4.5. About

84% of cells are higher than 90% and 36% of cells contain values higher than 95%. The mean values are above 92% (last row, bold face).

The values of SE, SP, their geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities are recorded for the model M1 detector using the ABOD classifier and dataset 4 is shown in Table 4.6. About 86% of cells are higher than 90%. The mean values are above 90% (last row, bold face). The values of SE, SP, their geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities are recorded for the model M2 detector using the NN classifier and dataset 1 is shown in Table 4.7. Each cell contains the average value of the corresponding cross-validation and number of runs. About 96% of cells are higher than 90%, and 42% of cells contain values higher than 95%. The mean values are near 95% (last row, bold face).

Table 4.5. Values of SE, SP, geometric-mean, F_1-measure for ADL and fall activities for each person using SVM with dataset 4. The last row is the average value over subjects.

Person	**SP (%)**	**SE (%)**	**Geometric mean (%)**	**F_1-measure (%) (ADL)**	**F_1-measure (%) (Fall)**
1	93.39	91.16	92.27	92.36	92.18
2	95.71	97.11	96.41	96.38	96.44
3	97.46	88.47	92.86	93.27	92.63
4	96.74	89.74	93.17	93.47	92.99
5	93.37	85.54	89.37	89.85	89.03
6	84.93	97.0	90.76	90.39	91.48
7	97.31	91.59	94.41	94.61	84.28
8	97.96	93.34	95.62	95.75	95.55
9	96.95	99	97.97	97.95	97.99
Mean	**94.87**	**92.55**	**93.70**	**93.78**	**93.62**

The values of SE, SP, their geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities are recorded for the model M3 detector using ABOD classifier and dataset 1 is shown in Table 4.8. Each cell contains the average value of the corresponding cross-validation and number of runs. About 96% of cells are higher than 90%, and 36% of cells contain values higher than 95%. The mean values are above 93% (last row, bold face).

Table 4.6. Values of SE, SP, geometric-mean, F_1-measure for ADL and fall activities for each person using M1 with ABOD classifier and dataset 4. The last row is the average value over subjects.

Person	SP (%)	SE (%)	Geometric mean (%)	F_1-measure (%) (ADL)	F_1-measure (%) (Fall)
1	95.53	94.16	94.84	94.88	94.81
2	93.67	98.10	95.86	95.79	95.97
3	97.27	94.10	95.67	95.75	95.62
4	88.58	96.30	92.36	92.14	92.72
5	93.31	92.38	92.84	92.88	92.81
6	74.42	94	83.64	82.68	85.62
7	84.28	96.30	90.09	89.67	90.84
8	98.35	94.34	96.32	96.42	96.27
9	95.85	92.20	94.01	94.13	93.91
Mean	**91.25**	**94.65**	**92.85**	**92.70**	**93.17**

Table 4.7. Values of SE, SP, geometric-mean, F_1-measure for ADL and fall activities for each person using model M2 with NN classifier with dataset 1. The last row is the average value over subjects.

Person	SP (%)	SE (%)	Geometric mean (%)	F_1-measure (%) (ADL)	F_1-measure (%) (Fall)
1	95.70	90.24	92.93	93.16	92.77
2	95.41	98.11	96.75	96.72	96.80
3	99.38	94.11	96.71	96.83	96.66
4	99.14	96.44	97.78	97.82	97.76
5	94.72	90.65	92.66	92.83	92.53
6	84.39	98.0	90.94	90.55	91.76
7	89.61	96.30	92.89	92.71	93.18
8	97.37	90.68	93.97	94.22	93.82
9	98.14	94.16	96.13	96.22	96.07
Mean	**94.87**	**94.30**	**94.53**	**94.56**	**94.59**

Table 4.8. Values of SE, SP, geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities for each person using model M3 with ABOD classifier with dataset 1. The last row is the average value over subjects.

Person	**SP (%)**	**SE (%)**	**Geometric mean (%)**	**F_1-measure (%) (ADL)**	**F_1-measure (%) (Fall)**
1	91.89	92.20	92.04	92.03	92.06
2	92.88	100	96.37	96.31	96.56
3	95.59	91.47	93.51	93.66	93.39
4	97.86	94.59	96.21	96.29	96.16
5	91.75	96.23	93.96	91.92	94.12
6	91.20	98.0	94.54	94.41	94.78
7	86.23	96.30	91.13	90.80	91.68
8	97.79	85.052	91.18	91.92	90.82
9	98.51	96.08	97.29	97.33	97.26
Mean	**93.74**	**94.43**	**94.03**	**93.85**	**94.09**

The values of SE, SP, their geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities are recorded for the model M4 detector using ABOD classifier and dataset 1 is shown in Table 4.9. Each cell contains the average value of the corresponding cross-validation and number of runs. About 98% of cells are higher than 90% and 54% of cells contain values higher than 95%. The mean values are above 95% (last row, bold face).

4.2.2 Models

To evaluate the models, a detailed comparison is made between pairs of models is depicted in bar diagrams, where for each subject, the difference in geometric-mean is calculated. In Figure 4.10, a comparison of model M1 with SC is shown. In this case, four people model M1 shows better performance and five people SC provides better performance. Based on the average of the people, SC is better by 0.85%.

Table 4.9. Values of SE, SP, geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities for each person using M4 with ABOD classifier with dataset 1. The last row is the average value over subjects.

Person	SP (%)	SE (%)	Geometric mean (%)	F_1-measure (%) (ADL)	F_1-measure (%) (Fall)
1	95.93	93.20	94.56	94.64	94.49
2	95.97	100	97.96	97.94	98.03
3	99.50	92.47	95.92	96.12	95.84
4	99.86	97.44	98.64	98.67	98.64
5	94.11	93.77	93.94	93.95	93.93
6	96.75	100	98.36	98.35	98.40
7	90.04	99.30	94.56	94.41	94.90
8	99.02	86.02	92.29	92.98	92.0
9	99.44	99.08	99.26	99.26	99.26
Mean	**96.74**	**95.70**	**96.17**	**96.26**	**96.17**

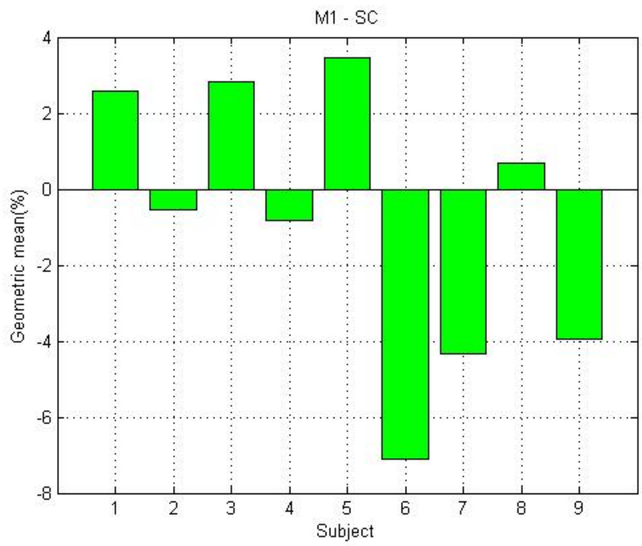


Figure 4.10. Difference in geometric-mean between model M1 (dataset 4) and SC (dataset 4).

In Figure 4.11, a comparison between model M2 and model M1 is shown. For six people, model M2 is better and the remaining three people achieve high detection rate with model M1. For person 5, the difference between the two models is small. Based on the average of the people, model M2 is better by 1.68%. In Figure 4.12, a comparison between model M3 and model M1 is shown. For six people model M3 is better, and the remaining three people achieve high detection rate with model M1. Based on the average over the people, model M3 outperforms model M1 by 1.18%.

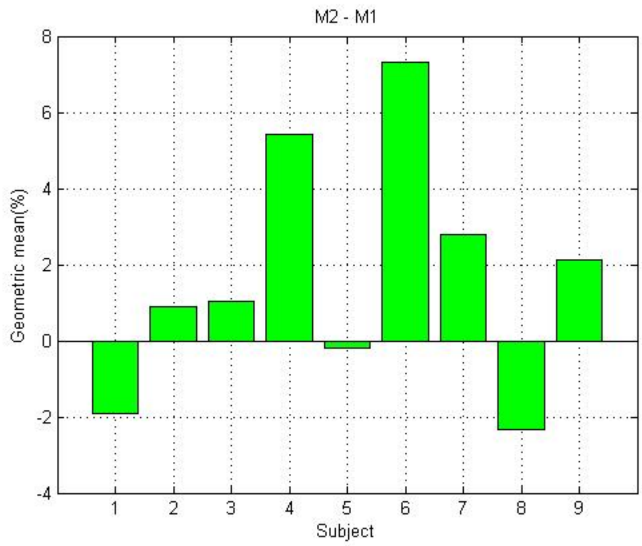


Figure 4.11. Difference in geometric-mean between model M2 (dataset 1) and model M1 (dataset 4).

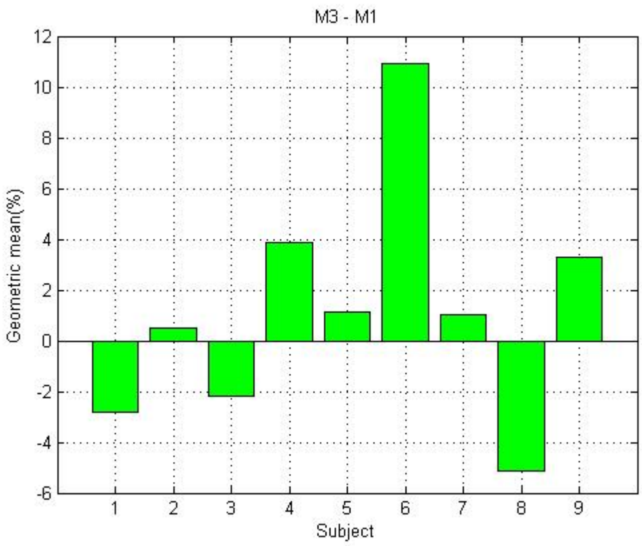


Figure 4.12. Difference in geometric-mean between model M3 (dataset 1) and model M1 (dataset 4).

In Figure 4.13, a comparison between model M4 and model M1 is shown. For seven people, model M4 is better, and the remaining two people prefer model M1. For person 1 and 3, the difference between the two models is small. Based on the average over the people, model M4 outperforms model M1 by 3.32%.

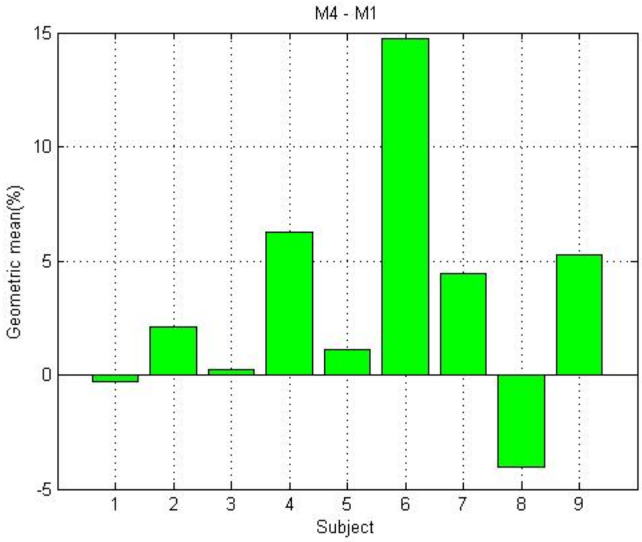


Figure 4.13. Difference in geometric-mean between model M4 (dataset 1) and model M1 (dataset 4).

In Figure 4.14, a comparison between model M2 with the SC is shown. For six people, model M2 is better and the remaining three people achieve high detection rate with SC. Based on the average over the people, M2 outperforms SC by 0.83%.

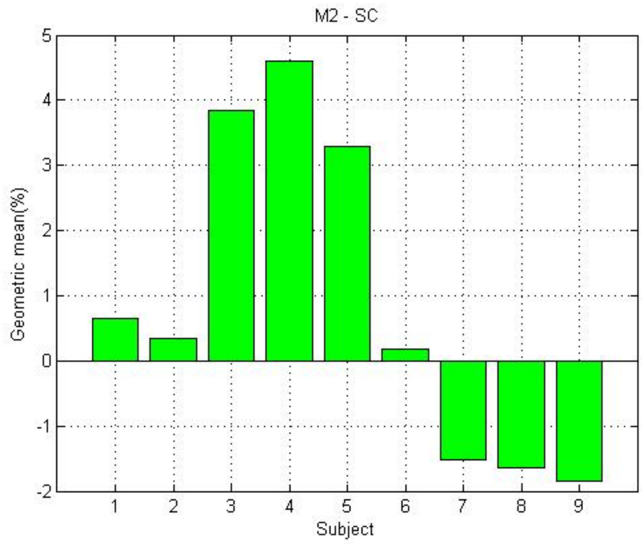


Figure 4.14. Difference in geometric-mean between model M2 (dataset 1) and SC (dataset 4).

In Figure 4.15, a comparison between model M3 and model M2 is shown. For three people, model M3 is better and the remaining six people prefer model M2. Based on the average over the people, model M2 outperforms M3 by 0.5%.

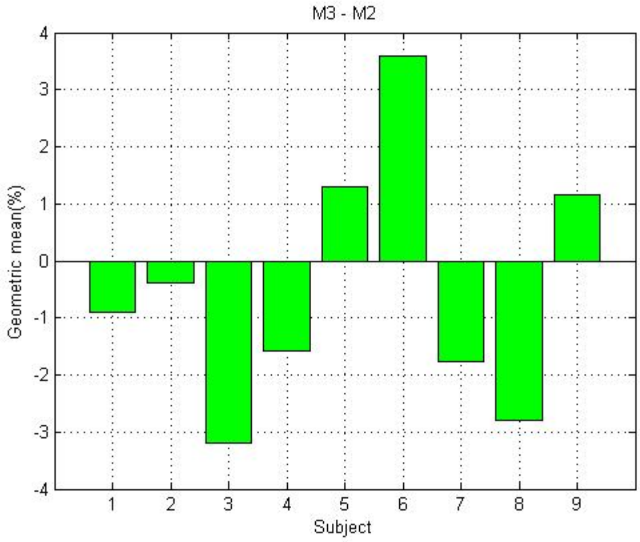


Figure 4.15. Difference in geometric-mean between model M3 (dataset 1) and model M2 (dataset 1).

In Figure 4.16, a comparison between model M4 and model M2 is shown. For seven people, model M4 is better and the remaining two people prefer model M2. Based on the average over the people, model M4 outperforms model M2 by 1.64%.

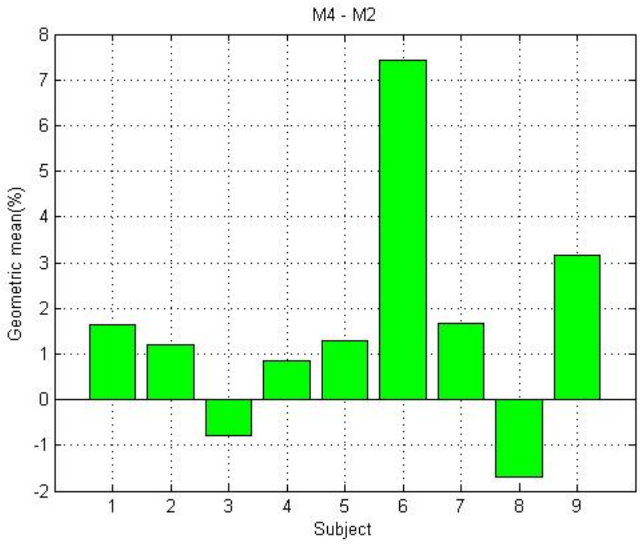


Figure 4.16. Difference in geometric-mean between model M4 (dataset 1) and model M2 (dataset 1).

In Figure 4.17, a comparison between model M3 and SC is shown. For five people, model M3 is better and the remaining four people prefer SC. For person 2, the difference between the two models is small. Based on the average over the people, model M3 outperforms SC by 0.33%.

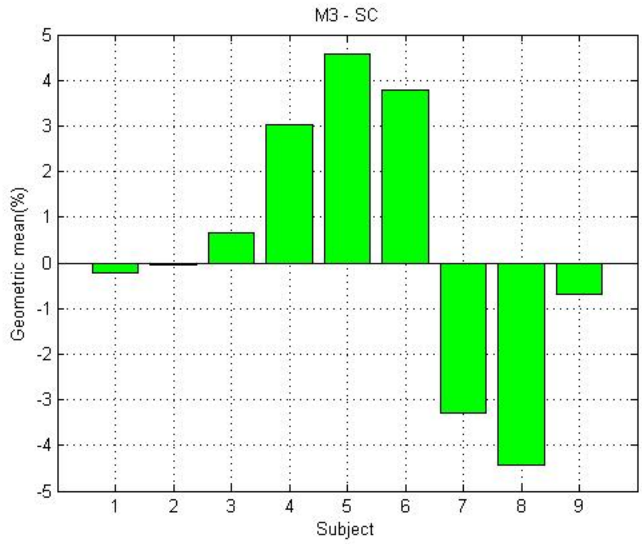


Figure 4.17. Difference in geometric-mean between model M3 (dataset 1) and model SC (dataset 4).

In Figure 4.18, a comparison between model M4 and model M3 is shown. In this case, model M4 is preferred for all the people. For person 5, the difference between the two models is small. Based on the average, the improvement of using model M4 over model M3 is 2.14%.

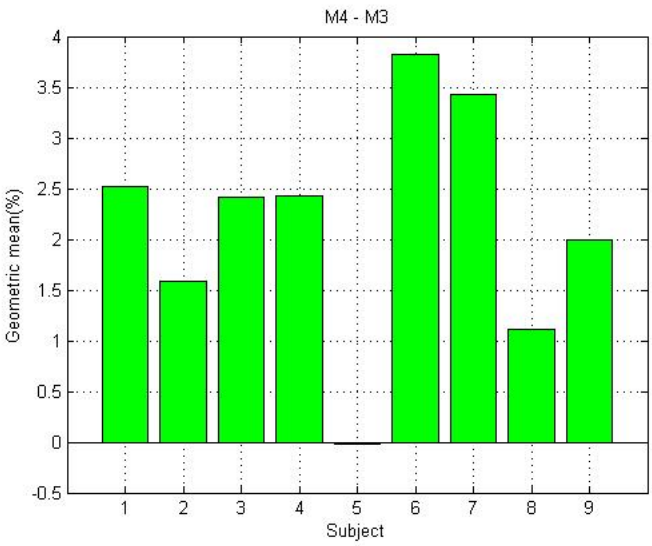


Figure 4.18. Difference in geometric-mean between model M4 (dataset 1) and model M3 (dataset 1).

In Figure 4.19, a comparison between model M4 and SC is shown. For eight people, model M4 is better and the remaining person prefers SC. For person 7, the difference between the two models is small. Based on the average over the people, model M4 outperforms SC by 2.47%.

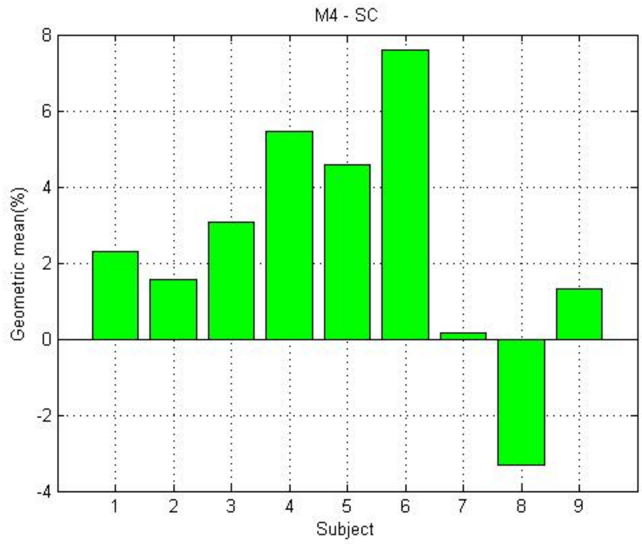


Figure 4.19. Difference in geometric-mean between M4 (dataset 1) and SC (dataset 4).

To summarize the comparison between the pairs of models, Table 4.10 shows the number of subjects for whom one model outperformed the other. The comparison is based on the following metrics: SE, SP, F_1-measure for ADL, and F_1-measure for fall activities.

4.3 EXPERIMENTAL

From the results, the best model is M4 using ABOD classifier and dataset 1 which was implemented on the Android device, using the algorithm in Figure 3.12 in order to save processing time. The Android smartphone was carried by the four subjects for three days and placed in the right trouser pocket of the subject. The reason for this was to ensure that the fall detection system can be personalized to the subject's movements and record new activities. The subjects were requested to incorporate the ADL moveset from Table 3.9, into their everyday lives. When the Android application starts up, the following screen in Figure 4.20(a) is shown. The toggle button was on when the subjects were carrying the smartphone in the pocket for three days. In Figure 4.20(a), the fall and false alarm button are disabled. It is enabled when the fall detection system detects a fall shown in Figure 4.20(b). In Figure 4.20(b), the subject can select the fall button when the fall has occurred or the false alarm button when no fall has occurred and that activity data is then sent to the classifier for retraining.

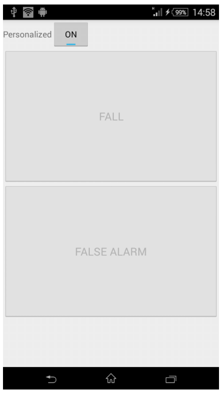
Table 4.10. Summary of the comparison with respect to SE, SP, geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities. For each pair of comparison, a cell contains the number of subjects for whom the member of the pair is better for SE, SP, geometric-mean or their F_1-measure.

Model	SE	SP	Geometric mean	F_1-measure (ADL)	F_1-measure (Fall)
M1 vs SC	7 vs 2	2 vs 7	4 vs 5	4 vs 5	5 vs 4
M1 vs M2	4 vs 5	1 vs 8	3 vs 6	3 vs 6	3 vs 6
M1 vs M3	4 vs 5	5 vs 4	3 vs 6	4 vs 5	2 vs 7
M1 vs M4	0 vs 9	3 vs 6	2 vs 7	2 vs 7	2 vs 7
M2 vs SC	5 vs 4	6 vs 3	6 vs 3	6 vs 3	7 vs 2
M2 vs M3	6 vs 3	3 vs 6	6 vs 3	7 vs 2	6 vs 3
M2 vs M4	1 vs 8	2 vs 7	2 vs 7	2 vs 7	2 vs 7
M3 vs SC	3 vs 6	7 vs 2	4 vs 5	5 vs 4	5 vs 4
M3 vs M4	0 vs 9	1 vs 7	1 vs 8	0 vs 9	1 vs 8
M4 vs SC	8 vs 1	8 vs 1	8 vs 1	7 vs 2	8 vs 1

When performing the experiments in Table 3.9, the toggle button is turned off to evaluate the performance of the fall detection system. When evaluating, the system should not be able to customize or learn new activities. The results of the experiment are shown in Table 4.11, where the values of SE, SP, their geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities is recorded. About 75% of cells are higher than 90%. The mean values are above 90% (last row, bold face).

4.4 CHAPTER SUMMARY

In this chapter, the results of the personalized fall detection system were presented. The different datasets and machine learning algorithms were applied to each personalized model and SC. The dataset was analysed based on the geometric-mean, and the difference in geometric-mean for each dataset between model M1 and model M4 for all unsupervised classifier was presented. For each model and SC, the best classifier and dataset was selected. All the models and SC were compared against one another and the best one was implemented on the Android device. Finally, the results of the fall detection system on the Android system were presented and evaluated. In Chapter 5, in-depth analysis and discussion of the results from Chapter 4 are provided.



(a) Start-up screen when the application opens.

(b) When a fall has occurred.

Figure 4.20. The screenshots of the Android fall detection application. The start-up screenshot (a), and the screenshot (b) is when a fall has occurred.

Table 4.11. Values of SE, SP, geometric-mean, F_1-measure for ADL, and F_1-measure for fall activities for each subject. The last row is the average value over subjects.

Person	**SP (%)**	**SE (%)**	**Geometric mean (%)**	**F_1-measure (%) (ADL)**	**F_1-measure (%) (Fall)**
1	92.31	85.71	88.95	89.36	88.64
2	89.74	90.48	90.11	90.07	90.15
3	94.87	90.48	92.66	92.83	92.51
4	92.31	95.24	93.76	93.68	93.87
Mean	**92.31**	**90.48**	**91.37**	**91.49**	**91.29**

CHAPTER 5 DISCUSSION

5.1 CHAPTER OVERVIEW

In this chapter, the results obtained from the four datasets, machine learning algorithms, models and the laboratory experiment using the Android application from Chapter 4 are reviewed and discussed. In Section 5.2, the different datasets are discussed, analysed, and compared against one another. In Section 5.3, the different models are discussed, analysed, and compared against one another and to determine which one is better. In Section 5.4, the different the machine learning algorithms used in the study are analysed to determine their strengths and weaknesses. In Section 5.5, the factors that influence the accuracy of the fall detection system are discussed. In Section 5.6, the results from the Android application experiment are analysed and discussed.

5.2 DATASET

Data used by the classifiers is very important as it influences the performance of the system. Different datasets (Table 3.6) were created with different features to determine which features will create a high accuracy classifier. For dataset 1, raw acceleration values from the accelerometer were used. Using raw acceleration values, the orientation of the smartphone and the three-dimensional information is provided [90]. The advantage of this approach is that no features extraction is required, which saves processing time. The disadvantage of this approach is that the feature vector may have features which may contain redundant information. In the second dataset, statistical features are extracted from the following signals in both the time and frequency domain: the x, y, z-axis from the accelerometer, the slope SMV and the SMV. Statistical features that are extracted from both time and frequency domain can improve the accuracy of the classification system [97]. The advantage of dataset 2 is that it clearly describes the data which can be used to differentiate between ADL and fall activities.

The disadvantage of dataset 2 is it increases the feature space hence increasing the dimensionality of the dataset and some of these features may make a minimal contribution to the overall success of the classifiers [97]. The third dataset makes use of PCA which creates new features from dataset 2. The advantage of PCA is that it creates new features based on the contribution of features in the dataset thus, removing meaningless information and reducing the dimensionality of the dataset. The disadvantage of this approach is that the simplest invariance cannot be captured unless the training data provides this information [84]. The fourth dataset makes use of a feature selection method called variance threshold to select features with have high variance from the second dataset. Features used in this dataset are shown in Table 3.4 and 3.5. The disadvantage is that it does not take into consideration the relationship between the feature variables and the target variable.

5.2.1 Datasets used by the supervised machine learning algorithms

Using dataset 1, k-NN achieved the highest geometric mean against SVM, with a small difference of 0.65%. For dataset 2-4, SVM dominates k-NN where the differences are 6.37%, 9.02% and 7.18% respectively. From all the datasets, the best one was dataset 4 using SVM classifier, where features were selected using variance threshold feature selection method which achieved a geometric mean of 93.70%. The difference between the best dataset (dataset 4 using SVM) and the worst dataset (dataset 1 using k-NN) is 1.83%. The SVM with dataset 4 was selected for SC.

5.2.2 Datasets used by the unsupervised machine learning algorithms

From Figures 4.1, 4.3, 4.5 and 4.7, it can be observed that all the unsupervised classifiers for every model and in every dataset, except the ISF classifier using dataset 3, achieved a minimum geometric mean of 80%. The following Tables 4.1-4.4, show the best unsupervised classifier for each dataset for the different models. All of them have ABOD classifier, except model M2 with dataset 1, and the NN classifier with the highest. Overall for model M1, the results are very close for dataset 1, 2 and 4, with dataset 4 being the best. The difference between the best dataset (dataset 4) and the worst dataset (dataset 3) is about 1.28%. With regard to model M2, the difference between the best dataset (dataset 1) and the worst dataset (dataset 3) is about 2.41% and the difference between the best dataset (dataset 1) and the second-best dataset (dataset 4) is about 1.39%. This indicates that adapting the classifier with ADL is better to use raw acceleration values (dataset 1). For model M3, the difference between the best dataset (dataset 1) and worst dataset (dataset 3) is about 1.9% but the difference is insignificant between dataset 1 and 4 with 0.05%. Both datasets 1 and 4 could be implemented, with dataset 1 not requiring any features to be extracted, and dataset 4 reducing the computational complexity when using distance calculations in both the NN and ABOD classifiers. For model M4, the difference between the datasets is huge and the difference between the best dataset (dataset 1) and the worst dataset (dataset 3) is about 3.89% and the difference between the best dataset (dataset 1) and the second-best dataset (dataset 4) is about 1.44%. Based on this, it is better to use the raw acceleration values (dataset 1) for model M4. From all the models, statistical features were slightly significant than raw acceleration values in model M1, otherwise raw acceleration values dominate the other models.

5.2.3 Datasets conclusion

When examining datasets 2 to 4, dataset 4, which is the feature selection method, performs the best and the worst one is dataset 3 which uses PCA for feature reduction. The low accuracy of PCA may be because fall data were not included in the dataset which made it difficult to interpret the activities since the training dataset did not have enough information. The dataset has a nonlinear relationship between ADL and fall activities, which make it difficult to interpret. Another reason as to why PCA and feature selection did not achieve the best results compared to raw acceleration values is that the features or principal components selected are from the entire dataset, and not based on the current dataset the classifier is using, which may reduce accuracy. This is because whenever a new record is added to the system, the features selection methods will need to select new features. This requires a new threshold and parameters to be selected which can cause computational complexity and low performance since the threshold and parameters are chosen differently for each feature selected. The computational complexity will increase for extracting features, selecting features and to determine the new parameters and threshold. To resolve the problems, both PCA and variance threshold is applied to the entire dataset to extract the features before the threshold and parameters are selected and the same features selected will be used whenever a new record is added.

In an adaptive or personalized model, feature selection and feature reduction would not work since each person does not perform the same activity and when selecting features, threshold, and parameters for the classifier, fall data would be required. The fall data would be impossible to obtain from the person unless simulated fall activities are used from other people from the training dataset which will reduce the quality of the classification system.

Studies that use statistical analysis achieve high accuracy because they use a limited number of ADLs and have a lot of training data for each ADL. The other reason for low accuracy when using statistical features is that features extracted did not represent the best way of describing data. Different feature extraction methods need to be investigated such as extracting more statistical features or extracting features from a wavelet transform of the accelerometer data. From the datasets, the conclusion is that raw acceleration values are best for discriminating between ADL and fall activities and provide numerous advantages such as reduced complexity since no features extraction and feature selection need to be implemented.

5.3 MODELS

Different models were analysed to determine the performance and how it affects each person from the dataset. Model M1 is the basic classifier which is implemented when the classifier is trained offline. Model M1 is generally applied to the supervised model where adapting is difficult since it results in an imbalance of data between the classes. Model M2 adapts by retraining the classifier whenever the input data is classified as an ADL. The classifier adapts the user movement better and it may also add new activities to the dataset. Model M3 adapts by retraining the false positive and this allows inclusion of new activities which the system had previously not learned. Finally, model M4 is the combination of model M2 and model M3. By applying adaption or personalization to the classifier, the results show that the performance of the classifier improves. The adaption model (M2, M3, and M4) achieved better results compared to model M1 and SC.

For each dataset, the difference in geometric mean between model M4 and model M1 is plotted for every unsupervised classifier which is shown in Figures 4.2, 4.4, 4.6, and 4.8. For dataset 1, the improvement is high for all the classifiers with an improvement of greater 3% excluding the one-class SVM classifier which has an improvement of less than 1%. The reason for the low improvement for one-class SVM classifier is that features used are not suitable for density estimation in one-class SVM [90]. For dataset 2, the improvement is high for all the classifiers except for the NN classifier which shows an improvement below of < 1% and the rest show an improvement of > 1.5%. The reason for the low improvement for the NN classifier is that all the features that were extracted were used which may contain redundant information thus affecting the performance of the classifier since it uses distances formula to classify between ADL and fall activity. For dataset 3, the improvement is high for one-class SVM and ISF classifier which shows an improvement of > 2%; whereas NN and ABOD classifier shows an improvement of < 1%. For dataset 4, the improvement is high for all the classifiers except the ISF classifier which shows an improvement below of < 1% and the rest show an improvement of > 1%. The other reason for the lack of improvement is that the classifiers are very good in terms of AUC which makes the improvement difference very low [90]. The following classifiers with improvement below of 1%, should not be implemented due to high processing, computational time and increased storage required for implementation of model M4. The classifiers achieve improvement > 3%, showing that the performance of the classifier can improve overtime by retraining ADLs and false-positive data.

To determine how each model (1 to 4) affects each person in the dataset; first the best classifier and the best dataset is selected for each model shown in Tables 4.1, 4.2, 4.3, and 4.4. Table 4.1, shows the unsupervised classifiers based on model M1 using different datasets where the geometric mean for all the classifiers is above > 91%, with ABOD classifier using dataset 4 being the best and the difference with dataset 2, 3 which uses ABOD classifier is not significant. Table 4.2, shows the model M2 classifiers where the geometric mean for all the classifiers is above > 92%, with the NN classifier using dataset 1 being the best. Table 4.3, shows model M3 classifiers where the geometric mean for all the classifiers is above > 92%, with the ABOD classifier using dataset 1 being the best. Table 4.4, shows model M4 classifiers where the geometric mean for all the classifiers are above > 92%, with the ABOD classifier using dataset 1 being the best. The following Tables 4.2, 4.3, and 4.4, one can notice the geometric-mean for dataset 4 (variance-threshold feature selection method) is always the highest compared to dataset 2 (all features extracted) and dataset 3 (PCA features). Figure 4.9, shows the comparison between SVM and k-NN using the four datasets, where SVM achieved high geometric mean using dataset 4.

Tables 4.5 to 4.9 is created, where the following metrics are recorded SP, SE, geometric mean, F_1-measure for ADL and F_1-measure for fall activities for every subject in the public dataset using the best dataset and machine learning algorithm for every model. The SC (SVM with dataset 4), in terms of the mean value all the metrics are above 92%, which is suitable for implementation. The SP is greater than SE by 2.32% likewise with F_1-measure for ADL activities is greater F_1-measure fall activities by 0.16%, which indicates the classifier detection for both ADL and fall activities is closely matched. For M1 (ABOD with dataset 4), in terms of the mean average, all the metrics are above > 91% indicating that an unsupervised model can be suitable for implementation. The SE is greater than SP by 3.4%; but by considering the difference between F_1-measure for fall activities and F_1-measure for ADL is 0.47%; which shows the detection of both fall and ADL are equal in performance. For M2 (NN with dataset 1), in terms of mean value, all the metrics are above > 94% which is better than model M1 and SC, with both the ADL and fall indicators closely matching with a small difference between them. Model M3 (ABOD with dataset 1), in terms of mean average all the metrics are above > 93%; and are better than model M1 and SC but lower than M2 by taking into account only F_1-measure for both ADL and fall activities. Finally, model M4 (ABOD with dataset 1), in terms of the mean average of the metrics is above > 96%; which is greater than all the models and the difference between F1-measure ADL and F_1-measure fall activities for M4 are equally matched.

Based on the mean averages, from Tables 4.5 to 4.9, all the models can be implemented on a smartphone and the mean averages show that the models can equally detect both ADL and fall activities. To determine how each model fares against each other for each subject, the difference in geometric mean is plotted in Figures 4.10 - 4.19, Table 4.10 shows a comparison between the pairs of models.

5.3.1 M1 vs SC

In Figure 4.10, with model M1 vs SC, where SC is the preferred choice for five people, false alarms are reduced by a factor of $\frac{(1-0.9125)}{(1-0.9487)} = 1.706$; but model M1 reduces the miss fall activities by a factor of $\frac{(1-0.9255)}{(1-0.9465)} = 1.39$. For model M1, the best-case improvement is person 5 with 3.47% and the worst-case improvement is person 8 with 0.70%. For SC, the best-case improvement is person 6 with 7.12% and the worst-case improvement is person 2 with 0.55%. The average improvement using model M1 for four people is 2.39% compared to SC five people with 3.35%. Considering Table 4.10 the results are very similar; with model M1 detecting falls better and SC detecting ADL better. More subjects are required to come up with a conclusion since the results are very similar.

5.3.2 M2 vs M1

In Figure 4.11, with M2 vs M1, model M2 is the preferred choice for six people. False alarms are reduced by a factor of $\frac{(1-0.9125)}{(1-0.9487)} = 1.706$; but model M1 reduces the miss fall activities by a factor of $\frac{(1-0.9430)}{(1-0.9465)} = 1.065$. For model M2, the best-case improvement is person 6 with 7.3% and the worst-case improvement is person 2 with 1.08%. For model M1, the best-case improvement is person 8, with 2.35% and the worst-case improvement is person 5 with 0.18%. The average improvement using model M2 for six people is 3.26% compared to model M1 three people with 1.48%. Considering Table 4.10, model M2 has a greater SP compared to model M1; since model M2 adapts ADL but in terms of SE they are very equal. In terms of F_1-measure, model M2 is leading with 6 people to 3 people. From the results, model M2 is better compared to model M1, and by adapting the system with the user ADL the accuracy of the system increases.

5.3.3 M3 vs M1

In Figure 4.12, model M3 is the preferred choice for six people. False alarms are reduced by a factor of $\frac{(1-0.9125)}{(1-0.9374)} = 1.40$; but model M1 reduces the miss fall activities by a factor of $\frac{(1-0.9443)}{(1-0.9465)} = 1.04$. For model M3, the best case improvement is person 6 with 10.90% and the worst-case improvement is person 2 with 0.51%. For model M1, the best case improvement is person 8 with 5.14% and the worst-case improvement is person 3 with 2.16%. The average improvement using model M3 for six people is 3.45% which is equal to model M1 three people of 3.45%. From Table 4.10, the difference between the models is one person in terms of SE, SP and F_1-measure for ADL. The F_1-measure for fall activities is better in model M3, compared to model M1. From the results, in terms of geometric mean and F1-measure for fall activities, model M3 is better, compared to model M1 as it detects more falls compared to model M1; but more data from more subjects are required to properly conclude.

5.3.4 M4 vs M1

In Figure 4.13, model M4 is the preferred choice for seven people. False alarms are reduced by a factor of $\frac{(1-0.9125)}{(1-0.9674)} = 2.684$ and model M4 reduces the miss fall activities by a factor of $\frac{(1-0.9465)}{(1-0.9570)} = 1.24$. For model M4, the best case improvement is person 6 with 14.72% and the worst-case improvement is person 3 with 0.25%. For model M1, the best case improvement is person 8 with 4.03% and the worst-case improvement is person 1 with 0.28%. The average improvement using model M4 for seven people is 4.89% compared to model M1 two people with 2.15%. From Table 4.10, model M4 outperforms model M1 in all the metrics; which makes model M4 the best choice to implement on a smartphone device.

5.3.5 M2 vs SC

In Figure 4.14, model M2 is the preferred choice for six people and it reduces the miss fall activities by a factor of $\frac{(1-0.9255)}{(1-0.9430)} = 1.31$. For model M2, the best case improvement is person 4 with 4.61% and the worst-case improvement is person 6 with 0.18%. For SC, the best case improvement is person 9 with 1.84% and the worst-case improvement is person 7, with 1.52%. The average improvement using model M2 for six people is 2.15% compared to SC three people with 1.67%. From Table 4.10, model

M2 outperforms SC in all the metrics and SE difference between the models is only by one person. From the results, model M2 is better compared to SC.

5.3.6 M3 vs M2

In Figure 4.15, model M2 is the preferred choice for six people. False alarms are reduced by a factor of $\frac{(1-0.9374)}{(1-0.9487)} = 1.22$, but model M3 reduces the miss fall activities by a factor of $\frac{(1-0.9443)}{(1-0.9430)} = 0.98$. For model M2, the best case improvement is person 3 with 3.20% and the worst-case improvement is person 2 with 0.38%. For model M3, the best case improvement is person 6 with 3.60% and the worst-case improvement is person 9 with 1.16%. The average improvement using model M2 for six people is 1.76% compared to model M3 for three people with 2.02%. From Table 4.10, model M3 is only better in SE compared to model M2. From the results, it is better to adapt the system with ADL instead of false-positives since ADL provides more information of the behaviour of the user.

5.3.7 M4 vs M2

In Figure 4.16, model M4 is the preferred choice for seven people. False alarms are reduced by a factor of $\frac{(1-0.9487)}{(1-0.9674)} = 1.57$, and M4 reduces the miss fall activities by a factor of $\frac{(1-0.9430)}{(1-0.9570)} = 1.33$. For model M4, the best case improvement is person 6 with 7.42% and the worst-case improvement is person 4 with 0.85%. For model M2, the best case improvement is person 8 with 1.68% and the worst-case improvement is person 3 with 0.79%. The average improvement using model M4 for seven people is 2.46% compared to model M2 three people with 1.23%. From Table 4.10, model M4 is better in every metric in the table, making it the best choice for implementation.

5.3.8 M3 vs SC

In Figure 4.17, model M3 is the preferred choice for five people which reduced the miss fall activities by a factor of $\frac{(1-0.9255)}{(1-0.9443)} = 1.34$, but SC reduces false alarms by a factor of $\frac{(1-0.9374)}{(1-0.9487)} = 1.22$. For model M3, the best case improvement is person 5 with 4.59% and the worst-case improvement is person 2 with 0.04%. For SC, the best case improvement is person 8 with 4.44% and the worst-case improvement is person 1 with 0.23%. The average improvement using model M3 for five people is 3.015% compared to SC four people with 1.73%. From Table 4.10, model M3 is better in SE and SC better in SP; in terms of the other metrics they are very close. From the results, it is difficult to come up with a conclusion since model M3 is better than SC by one person; hence more subjects are required to conclude.

5.3.9 M4 vs M3

In Figure 4.18, model M4 is the preferred choice for eight people. False alarms are reduced by a factor of $\frac{(1-0.9374)}{(1-0.9674)} = 1.92$, and model M4 reduces the miss fall activities by a factor of $\frac{(1-0.9443)}{(1-0.9570)} = 1.30$. For M4, the best case improvement is person 6 with 3.43% and the worst-case improvement is person 8 with 2%. For model M3, person 5 is the preferred one with 0.02%. The average improvement using model M4 for eight people is 2.41% compared to M3 one person with 0.02%. Model M4 is better in every metric in Table 4.10, making it the best choice for implementation.

5.3.10 M4 vs SC

In Figure 4.19, model M4 is the preferred choice for eight people. False alarms are reduced by a factor of $\frac{(1-0.9374)}{(1-0.9674)} = 1.92$, and model M4 reduces the miss fall activities by a factor of $\frac{(1-0.9570)}{(1-0.9443)} = 1.73$. For model M4, the best case improvement is person 6 with 7.60% and the worst-case improvement is person 7 with 0.15%. For SC, person 8 is the preferred one with 3.33%. The average improvement using model 4 for eight people is 3.25% compared to supervised model one person with 3.33%. Model M4 is better in every metric in Table 4.10, making it the best choice for implementation.

5.3.11 Models conclusion

Model M4 is the best as it outranks every model in all metrics which indicates that adaptation is key to creating a robust fall detection system. Even though model M4 performs better than the rest of the models, it does not work for all the subjects. Since adaption does not work for all people, the study should be expanded with the different characteristics of the user, to further examine.

5.4 CLASSIFIERS

5.4.1 Unsupervised machine learning algorithms

To determine the best novelty detectors, the following methods have been considered: the basic NN and three state-of-the-art algorithms, ABOD, ISF and one-class SVM. However, ABOD has proved to be the best one using dataset 1 and the difference with NN is not significant. Since ADL are not limited to a set of activities, methods such as ABOD and NN achieved high accuracy because the incoming data is compared to the classifier training data to determine if the pattern exists. In high dimensional space, the use of distance is useless; since the distance between an outlier sample and normal sample becomes small. This makes it difficult to differentiate if the input data is an outlier. This can be solved by using angles instead of distance; since angles are more stable. Since the input size is large, ABOD outperforms all the classifiers in every dataset except NN using model M2 and dataset 1.

The possible reason for ABOD high performance is that it is able to detect outliers which have a spectrum featuring high fluctuations. The disadvantage of ABOD is that the computational and processing time increases as the dataset grows. From the results, NN is the second best classifier compared to the rest novelty detectors and the difference with ABOD accuracy is very close. The advantages of NN are that it is simple to implement since it returns no parameters to tune and it is easy to compute [90]. A possible reason for NN not having high accuracy compared to ABOD is that the input vector is large, which makes it difficult to differentiate between similar or close patterns. If the fall activity is similar to the ADL, NN would not be able to differentiate since it uses only distance as a metric to classify. Another possible reason which may affect the accuracy of NN is if the data consists of regions of different densities [90].

The disadvantage of NN is that the processing and computational time grows as the dataset grows. One-class SVM is the most popular unsupervised method for fall detection. One-class SVM achieves decent results, with 2 - 4 datasets achieving better results compared to the first dataset. One-class SVM was derived from SVM, which was not initially designed to be an outlier detection. However, one-class SVM is sensitive to outliers that can change the hyper-plane position to a higher degree. The features used may not be suitable for density estimation in one-class SVM. ISF has lower computational time compared to other novelty methods and it was known to be the most efficient way to detect outliers compared to other methods that use distance and density measures to detect outliers. ISF performs the worst among the classifiers because ISF randomly selects a feature and also randomly chooses a splitting value between the maximum and minimum values of the selected feature when designing a tree. The feature may not have a high variance between ADL and fall activity, which can result in poor accuracy. Since the classifier uses random numbers, the accuracy varies every time the ISF classifier runs the same dataset; which may result in good classifier or a bad classifier. Not all activities share the same characteristics, which can result in a bad split. Performance of isolation forest is influenced by the large sampling size which affects its ability to isolate anomalies as normal instances and can interfere with isolation process, which decreases its performance of isolating anomalies [97].

5.4.2 Supervised machine learning algorithms

To determine the best-supervised detector, the following popular methods in fall detection have been considered: SVM and k-NN. Overall, all classifiers perform the best; but SVM outperforms k-NN in all of the datasets except for dataset 1. Since k-NN is a lazy learner, it does not train, which makes it difficult to optimize and k-NN is sensitive to features which do not contribute to the classification. The reason for the high accuracy for SVM is that it has decision boundary to differentiate between ADL and fall activities. SVM achieves high accuracy in the experimental setup, but not in reality since it does not have the true fall event data. The reason for SVM not achieving higher accuracy compared to the personalized models is that the performance of the SVM classifier is highly dependent on the selected features. In terms of base-classifier or M1, SVM achieved the best performance and the difference with ABOD is 0.85% in terms of geometric-mean.

5.4.3 Classifier conclusion

The unsupervised classifier is the best option for detecting falls and it can reach similar accuracy compared to SC. Both NN and ABOD concepts are simpler compared to SVM and one-class SVM; which allows personalization using model M4 to be done on the fly using an Android smartphone device. The advantage of unsupervised methods and personalization models is that it only requires ADL data, unlike supervised methods which use laboratory fall data, which may not represent real fall event. The effectiveness of the fall detection system can only be determined using falls in reality; but based on the available data, the personalization system proved that it can be used to effectively detect falls. Since ABOD does not use any training algorithm, it removes the complexity of retraining the classifier where it only needs the table of the ADL data. The table of the ADL data is updated every time the user carries the smartphone in the trouser pocket. Classifiers with parameters would require the data to be sent to a server, where the server would retrain the classier and send back the parameters or classifier model to the smartphone however, this would require an internet connection. The best classifier from the experiment is the ABOD classifier combined with model M4 and dataset 1.

5.5 ACCURACY

The placement of the sensor may affect the accuracy of the system if not placed in the correct position or if the sensor is worn wrong [87]. The placement of the sensor around the waist produces low acceleration forces, hence low frequency which can result in low power consumption. There are not a lot of frequency movements around the waist region as it is the centre of gravity [87], [92], [97]. The problem with the placement of a sensor around the waist is that its uncomfortable and very invasive therefore, the pocket is chosen since it is the most common place.

The reason for the low sensitivity may be that when test subjects perform the fall they hesitate to fall properly, due to danger. ADL accuracy is influenced by the dataset created; if the dataset does not contain that particular activity, the system generates a false positive. The ADL was requested for the subjects to perform but they performed their normal ADL reality. The different characteristics of the person need to be studied in order to fully understand how it would affect the performance of the classifier. ADLs have similar attributes to falls such as sitting down, standing up and jumping which can result in false positives [94]. The falls used in both the dataset and the experiment were simulated falls which were conducted on a soft mattress to protect the user from injuries. Falls that occur on a hard material such as floors can result in serious injuries; whereas the mattress can reduce/absorb the impact which does not represent a true fall in reality [93], [94]. Since the fall is simulated, the results are affected because the subject may change their posture or response to reduce the impact of the fall [93]. Subjects also wear protective equipment in these experiments which can affect the fall signal. The true test of the system needs to be tested on an elderly person, but it is difficult to obtain real falls with only 7% of fall detection studies using fall data recorded from the elderly subject in reality; whereas 93.8% studies used simulated fall data [93].

The accelerometer sensors can affect the performance of fall detection because different sensors have different noise levels. The range and resolution of the sensors in the smartphone can affect the performance of the fall detection system [88]. Phones were used in the experiment in this paper and the dataset was low cost where the accelerometer range is about $\sim 2g$; using a larger accelerometer range can reduce false alarms [90]. The reason being $\sim 2g$ accelerometer cannot clearly differentiate acceleration peaks caused by the impact during a fall [88]. The sampling size can affect the performance if the signal is not properly captured. The effect of fall direction, the direction of the fall affect the information content of the signals provided by separate axes of the accelerometer worn by the subjects.

This implies that the axis that provides the most valuable information in detecting a fall may change according to the direction of the fall. The smartphone is not fixed in the user's pocket, which can result in noise when the smartphone moves around the pocket. Accuracy of the fall detection system increases when more than one sensor is attached to the subject; which makes it uncomfortable and invasive [92].

Increasing the threshold value tends to increase the number of false negatives and decreasing the threshold value tends to increase the number of false positives. From the ROC, the threshold was selected, which is the best balance of minimizing both false positives and false negatives. Since fall accuracy is more important than ADL accuracy, the threshold could also be adjusted based on the ROC curve to favour fall accuracy. This can lead to high false positives which can be compensated by adding more decisions such as angle and signal magnitude area. The problem with using signal magnitude area is that it assumes that the person fell and is in the lying state for a certain period. The accuracy of the system can increase by gathering additional information from context. The small impact signal of a fall can be in ADL and falls having similar impact signals [94]. The system demonstrated that it can work under real-world conditions. Since a fall that goes undetected could mean injury to an elderly resident and a false positive only means potential inconvenience to caretakers, a threshold value should be chosen with a preference towards minimizing false negatives. Future studies can be conducted to determine the characteristics of fall and ADL acceleration patterns. Studies that evaluated fall detection systems usually used a limited number of ADL and fall activities, which made it possible for these studies to achieve high-performance [19]. The accuracy of these studies is reduced when new activities are included in the system [19].

5.6 ANDROID APPLICATION

The Android application uses raw acceleration values with M4 and the ABOD classifier. The purpose was to test how personalization would fare in everyday activities. From the results, person 1 SE was below 89% due to hesitation when performing the fall activities; which resulted in the fall pattern being similar to an ADL pattern. Overall, the mean values are > 90% which is very good considering the phone was placed in the pocket, and more ADLs were added to the dataset compared to previous studies. The F_1-measure for ADL and fall activities are almost equal which shows that the fall detection system equally detects ADL and fall activities.

5.7 CHAPTER SUMMARY

In this chapter, the results obtained from datasets, models, machine learning and Android application in Chapter 4 were reviewed and discussed. Overall, the different datasets performed well, with dataset 1 using the raw accelerometer performing the best. Different models were investigated and adaption model M4 achieved the highest accuracy which adapts to ADL and false-positive data and which allows users to perform various activities. This was not limited to a certain set of ADL that the user can perform, like other studies. Different unsupervised machine learning algorithms were analysed with ABOD achieving the best results by combining angle-based and distance-based methods for classifying data. The advantage of unsupervised machine learning algorithm is that it allows personalization since these algorithms only require one class data, in this case, ADL data. The best SC was SVM and SVM achieved better results compared to ABOD when using M1, but ABOD with M4 achieved higher and greater performance than SVM. The problem with SC is that it cannot be used for personalization; since it may result in an imbalance of data which is difficult to compensate or fix when using SC. The experimental setup shows that model M4 can work in reality by achieving high accuracy. In Section 6, the conclusion of the study, the benefits of the study, and the recommendation for future work.

CHAPTER 6 CONCLUSION

6.1 CONCLUSION

The goal of the study is to detect different types of falls while keeping the false-positive rate as low as possible by applying different personalization models. The personalized model has an ability to learn new activities and recognize user movements. The study covers an in-depth analysis of the different types of models, machine learning algorithms and input features which can be applied to the system. The research showed an increase in performance by personalizing or adapting the fall detection model and unsupervised classifiers that make use of ADL for training can achieve better performance compared to supervised classifiers which require both ADL and fall data. Retraining of false positive input data and data that are detected as ADL increases accuracy. Different features, feature selection, and machine learning algorithms were investigated to obtain the optimal model for fall detection. The results of different models showed that accuracy increased. The objective was to develop a system that will be able to learn user movements and new ADL, which will improve the accuracy of the system. Thus, producing a novel algorithm. The algorithm is capable of overcoming the inherent limitations of fall detection and improve the accuracy of the system.

The results of this research can be summarized as follows:

- The use of raw acceleration values as input data provides higher accuracy compared to features extracted from the accelerometer, or features selected from variance-threshold feature selection and new features created using PCA. Raw acceleration values provide more context information.
- Model M4, which makes use of both ADL and false-positive data performed best compared to other models, showing that the future fall detection system should adapt, which will result in increased performance.

- Even though supervised classifiers are best for detecting falls from offline data, they cannot be used as an adaptive system since it will create an imbalance of data. This can be solved by using unsupervised classifier. The study shows that SVM is the best SC and it can achieve higher accuracy compared to unsupervised classifier using the M1 model. Among all the unsupervised machine learning algorithms, ABOD performed the best and achieved higher accuracy when using model M4 with raw acceleration values as input data compared to SVM using model M1 and features selected from the variance threshold method as input data.

Even though the system is not 100% accurate, it provides other benefits compared to other fall detection systems; such as an unlimited set of ADL's can be performed and it does not need fall data. The system can work in both outdoor and indoor environment. Although there are areas of this research that can be improved, this research provides some novel contributions; firstly the use of the novelty detectors ABOD and ISF; and secondly, the use of variance feature selection methods. The study provides a comparison between raw data values vs features extraction. The research also successfully demonstrated the need for an adaptation model and how accuracy can be improved, using a public dataset.

6.2 BENEFITS OF THE STUDY

The fall detection model defined is capable of overcoming the problem of the current systems which are limited to certain ADL and have a low accuracy of detecting falls in reality. By retraining the input data when the classifier recognizes it as ADL it helps to learn or adapt to the user movement. By retraining the input data, when the classifier wrongly detects an ADL as a fall, it helps to learn a new ADL. This reduces false positives and the base classifier does not need to have data representing different characteristics of the person since the classifier will adapt to the person’s traits. The algorithm becomes less intrusive. The study investigates the different unsupervised algorithms, including ABOD and ISF which were not previously implemented in any fall detection study. The study presents a different approach to fall detection, compared to traditional approaches where the classifier is trained with a certain dataset, and the system is never updated to include new data. The advantage of the study is that it was tested on real data obtained from subjects, instead of experimental data used in current fall detection studies, where a user is performing a particular task. The approach and methods demonstrated in this study will be applicable in reality where there are multiple activities and continuous data streams.

The advantage of this system is that it provides independence, reduces health care cost, improves the quality of life and does not restrict users to a limited set of activities.

6.3 RECOMMENDATION FOR FUTURE WORK

The algorithm can be improved further to reduce computational time and processing time and increase the overall accuracy of the algorithm. The biggest problem with the algorithm is the storage space, as the algorithm learns new data; it is added to the application. As the data increase, the computational and processing time increases. More research needs to be done to solve the problem. The biggest problem with the current system is that it only works when placed at a particular location. To make it work at different locations, an investigation into on-body sensor localization needs to be conducted. Further recommendations include; investigating other outlier detection methods to improve accuracy creating a hybrid version of these classifiers and improving battery life through reduction of processing time and sampling rates as well as sleep mode. Cloud storage implementation can be considered as another research implementation. This can allow researchers to upload any recorded ADL or fall activities to verify future fall detection systems.

REFERENCES

[1] Y. S. Delahoz and M. A. Labrador, "Survey on fall detection and fall prevention using wearable and external sensors," *Sensors*, vol. 14, no. 10, pp. 19 806–19 842, 2014.

[2] J. Wannenburg and R. Malekian, "Physical activity recognition from smartphone accelerometer data for user context awareness sensing," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2016.

[3] R. M. Gibson, A. Amira, N. Ramzan, P. Casaseca-de-la Higuera, and Z. Pervez, "Multiple comparator classifier framework for accelerometer-based fall detection and diagnostic," *Applied Soft Computing*, vol. 39, pp. 94–103, 2016.

[4] N. Pannurat, S. Thiemjarus, and E. Nantajeewarawat, "A hybrid temporal reasoning framework for fall monitoring," *IEEE Sens.*, vol. 17, pp. 1749–1759, 2017.

[5] S. S. Khan and J. Hoey, "Review of fall detection techniques: A data availability perspective," *Medical Engineering & Physics*, vol. 39, pp. 12–22, 2017.

[6] F. De Backere, F. Ongenae, F. Van den Abeele, J. Nelis, P. Bonte, E. Clement, M. Philpott, J. Hoebeke, S. Verstichel, A. Ackaert *et al.*, "Towards a social and context-aware multi-sensor fall detection and risk assessment platform," *Computers in Biology and Medicine*, vol. 64, pp. 307–320, 2015.

[7] P. van de Ven, H. O'Brien, J. Nelson, and A. Clifford, "Unobtrusive monitoring and identification of fall accidents," *Medical Engineering & Physics*, vol. 37, no. 5, pp. 499–504, 2015.

[8] B. Kwolek and M. Kepski, “Human fall detection on embedded platform using depth maps and wireless accelerometer,” *Computer Methods and Programs in Biomedicine*, vol. 117, no. 3, pp. 489–501, 2014.

[9] M. S. Khan, M. Yu, P. Feng, L. Wang, and J. Chambers, “An unsupervised acoustic fall detection system using source separation for sound interference suppression,” *Signal Processing*, vol. 110, pp. 199–210, 2015.

[10] M. Bosch-Jorge, A.-J. Sánchez-Salmerón, Á. Valera, and C. Ricolfe-Viala, “Fall detection based on the gravity vector using a wide-angle camera,” *Expert Systems with Applications*, vol. 41, no. 17, pp. 7980–7986, 2014.

[11] K. Ozcan, S. Velipasalar, and P. K. Varshney, “Autonomous fall detection with wearable cameras by using relative entropy distance measure,” *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 1, pp. 31–39, 2017.

[12] C. Garripoli, M. Mercuri, P. Karsmakers, P. J. Soh, G. Crupi, G. A. Vandenbosch, C. Pace, P. Leroux, and D. Schreurs, “Embedded dsp-based telehealth radar system for remote in-door fall detection,” *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 1, pp. 92–101, 2015.

[13] M. Á. Á. de la Concepción, L. M. S. Morillo, J. A. Á. García, and L. González-Abril, “Mobile activity recognition and fall detection system for elderly people using ameva algorithm,” *Pervasive and Mobile Computing*, vol. 34, pp. 3–13, 2017.

[14] R. Luque, E. Casilari, M.-J. Morón, and G. Redondo, “Comparison and characterization of android-based fall detection systems,” *Sensors*, vol. 14, no. 10, pp. 18 543–18 574, 2014.

[15] B. Andò, S. Baglio, C. O. Lombardo, and V. Marletta, “An event polarized paradigm for adl detection in aal context,” *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 7, pp. 1814–1825, 2015.

[16] V. R. Shen, H.-Y. Lai, and A.-F. Lai, “The implementation of a smartphone-based fall detection system using a high-level fuzzy petri net,” *Applied Soft Computing*, vol. 26, pp. 390–400, 2015.

[17] C. Zhang, C.-F. Lai, Y.-H. Lai, Z.-W. Wu, and H.-C. Chao, "An inferential real-time falling posture reconstruction for internet of healthcare things," *Journal of Network and Computer Applications*, 2017.

[18] R. Igual, C. Medrano, and I. Plaza, "A comparison of public datasets for acceleration-based fall detection," *Medical Engineering & Physics*, vol. 37, no. 9, pp. 870–878, 2015.

[19] A. T. Özdemir and B. Barshan, "Detecting falls with wearable sensors using machine learning techniques," *Sensors*, vol. 14, no. 6, pp. 10 691–10 708, 2014.

[20] L. Yang, Y. Ren, H. Hu, and B. Tian, "New fast fall detection method based on spatio-temporal context tracking of head by using depth images," *Sensors*, vol. 15, no. 9, pp. 23 004–23 019, 2015.

[21] L. Yang, Y. Ren, and W. Zhang, "3d depth image analysis for indoor fall detection of elderly people," *Digital Communications and Networks*, vol. 2, no. 1, pp. 24–34, 2016.

[22] H. Jian and H. Chen, "A portable fall detection and alerting system based on k-nn algorithm and remote medicine," *China Communications*, vol. 12, no. 4, pp. 23–31, 2015.

[23] M. Daher, A. Diab, M. E. B. El Najjar, M. Khalil, and F. Charpillet, "Elder tracking and fall detection system using smart tiles," *Sensors*, vol. 15800, p. 1, 2016.

[24] M. Aslan, A. Sengur, Y. Xiao, H. Wang, M. C. Ince, and X. Ma, "Shape feature encoding via fisher vector for efficient fall detection in depth-videos," *Applied Soft Computing*, vol. 37, pp. 1023–1028, 2015.

[25] Y. Yun and I. Y.-H. Gu, "Human fall detection in videos by fusing statistical features of shape and motion dynamics on riemannian manifolds," *Neurocomputing*, vol. 207, pp. 726–734, 2016.

[26] B. Andò, S. Baglio, C. O. Lombardo, and V. Marletta, "A multisensor data-fusion approach for adl and fall classification," *IEEE Transactions on Instrumentation and Measurement*, vol. 65, no. 9, pp. 1960–1967, 2016.

[27] F. Bagalà, C. Becker, A. Cappello, L. Chiari, K. Aminian, J. M. Hausdorff, W. Zijlstra, and J. Klenk, "Evaluation of accelerometer-based fall detection algorithms on real-world falls," *PloS One*, vol. 7, no. 5, p. e37062, 2012.

[28] A. M. Sabatini, G. Ligorio, A. Mannini, V. Genovese, and L. Pinna, "Prior-to-and post-impact fall detection using inertial and barometric altimeter measurements," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 7, pp. 774–783, 2016.

[29] E. Principi, D. Droghini, S. Squartini, P. Olivetti, and F. Piazza, "Acoustic cues from the floor: A new approach for fall classification," *Expert Systems with Applications*, vol. 60, pp. 51–61, 2016.

[30] Y. Zigel, D. Litvak, and I. Gannot, "A method for automatic fall detection of elderly people using floor vibrations and soundâĂŤproof of concept on human mimicking doll falls," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 12, pp. 2858–2867, 2009.

[31] B. Kwolek and M. Kepski, "Improving fall detection by the use of depth sensor and accelerometer," *Neurocomputing*, vol. 168, pp. 637–645, 2015.

[32] K. Ozcan and S. Velipasalar, "Wearable camera-and accelerometer-based fall detection on portable devices," *IEEE Embedded Systems Letters*, vol. 8, no. 1, pp. 6–9, 2016.

[33] D. Naranjo-Hernandez, L. M. Roa, J. Reina-Tosina, and M. A. Estudillo-Valderrama, "Personalization and adaptation to the medium and context in a fall detection system," *IEEE Transactions on Information Technology in Biomedicine*, vol. 16, no. 2, pp. 264–271, 2012.

[34] X. Ma, H. Wang, B. Xue, M. Zhou, B. Ji, and Y. Li, "Depth-based human fall detection via shape features and improved extreme learning machine," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 6, pp. 1915–1922, 2014.

[35] X. Hu and X. Qu, "An individual-specific fall detection model based on the statistical process control chart," *Safety Science*, vol. 64, pp. 13–21, 2014.

[36] G. Shi, C. S. Chan, W. J. Li, K.-S. Leung, Y. Zou, and Y. Jin, "Mobile human airbag system for fall protection using mems sensors and embedded svm classifier," *IEEE Sensors Journal*, vol. 9, no. 5, pp. 495–503, 2009.

[37] X. Zhuang, J. Huang, G. Potamianos, and M. Hasegawa-Johnson, "Acoustic fall detection using gaussian mixture models and gmm supervectors," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE, 2009, pp. 69–72.

[38] L. Liu, M. Popescu, M. Skubic, and M. Rantz, "An automatic fall detection framework using data fusion of doppler radar and motion sensor network," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*. IEEE, 2014, pp. 5940–5943.

[39] Y. Li, K. Ho, and M. Popescu, "A microphone array system for automatic fall detection," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 5, pp. 1291–1301, 2012.

[40] M. Popescu, B. Hotrabhavananda, M. Moore, and M. Skubic, "Vampir-an automatic fall detection system using a vertical pir sensor array," in *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2012 6th International Conference on*. IEEE, 2012, pp. 163–166.

[41] T. Zhang, J. Wang, L. Xu, and P. Liu, "Fall detection by wearable sensor and one-class svm algorithm," *Intelligent Computing in Signal Processing and Pattern Recognition*, pp. 858–863, 2006.

[42] M. Yu, Y. Yu, A. Rhuma, S. M. R. Naqvi, L. Wang, and J. A. Chambers, "An online one class support vector machine-based person-specific fall detection system for monitoring an elderly individual in a room environment," *IEEE Journal of Biomedical and Health Informatics*, vol. 17, no. 6, pp. 1002–1014, 2013.

[43] M. Popescu and A. Mahnot, "Acoustic fall detection using one-class classifiers," in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*. IEEE, 2009, pp. 3505–3508.

[44] G. Debard, P. Karsmakers, M. Deschodt, E. Vlaeyen, E. Dejaeger, K. Milisen, T. Goedemé, B. Vanrumste, and T. Tuytelaars, "Camera-based fall detection on real world data," *Outdoor and Large-Scale Real-World Scene Analysis*, pp. 356–375, 2012.

[45] C. Medrano, R. Igual, I. Plaza, M. Castro, and H. M. Fardoun, "Personalizable smartphone application for detecting falls," in *Biomedical and Health Informatics (BHI), 2014 IEEE-EMBS International Conference on*. IEEE, 2014, pp. 169–172.

[46] B. Kwolek and M. Kepski, "Fuzzy inference-based fall detection using kinect and body-worn accelerometer," *Applied Soft Computing*, vol. 40, pp. 305–318, 2016.

[47] X. Hu and X. Qu, "Detecting falls using a fall indicator defined by a linear combination of kinematic measures," *Safety Science*, vol. 72, pp. 315–318, 2015.

[48] C. Rougier, E. Auvinet, J. Rousseau, M. Mignotte, and J. Meunier, "Fall detection from depth map video sequences," in *International Conference on Smart Homes and Health Telematics*. Springer, 2011, pp. 121–128.

[49] A. Hakim, M. S. Huq, S. Shanta, and B. Ibrahim, "Smartphone based data mining for fall detection: analysis and design," *Procedia Computer Science*, vol. 105, pp. 46–51, 2017.

[50] T. Tamura, T. Yoshimura, M. Sekine, M. Uchida, and O. Tanaka, "A wearable airbag to prevent fall injuries," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 6, pp. 910–914, 2009.

[51] K. Yang, C. R. Ahn, M. C. Vuran, and S. S. Aria, "Semi-supervised near-miss fall detection for ironworkers with a wearable inertial measurement unit," *Automation in Construction*, vol. 68, pp. 194–202, 2016.

[52] S. Huang, Y. Yang, and W. Liu, "An enhanced fall detection approach based on cost sensitivity analysis," in *Software and Network Engineering (SSNE), 2011 First ACIS International Symposium on*. IEEE, 2011, pp. 81–85.

[53] J. T. Perry, S. Kellog, S. M. Vaidya, J.-H. Youn, H. Ali, and H. Sharif, "Survey and evaluation of real-time fall detection approaches," in *High-Capacity Optical Networks and Enabling Technologies (HONET), 2009 6th International Symposium on*. IEEE, 2009, pp. 158–164.

[54] J. Wang, Z. Zhang, B. Li, S. Lee, and R. Sherratt, "An enhanced fall detection system for elderly person monitoring using consumer home networks," *IEEE Transactions on Consumer Electronics*, vol. 60, no. 1, pp. 23–29, 2014.

[55] D. M. Karantonis, M. R. Narayanan, M. Mathie, N. H. Lovell, and B. G. 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.

[56] M. Á. Estudillo-Valderrama, L. M. Roa, J. Reina-Tosina, and D. Naranjo-Hernández, "Design and implementation of a distributed fall detection system-personal server," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 6, pp. 874–881, 2009.

[57] M. Kangas, A. Konttila, P. Lindgren, I. Winblad, and T. Jämsä, "Comparison of low-complexity fall detection algorithms for body attached accelerometers," *Gait & Posture*, vol. 28, no. 2, pp. 285–291, 2008.

[58] S. Steidl, C. Schneider, and M. Hufnagl, *Fall detection by recognizing patterns in direction changes of constraining forces*. na, 2012.

[59] W. Wu, S. Dasgupta, E. E. Ramirez, C. Peterson, and G. J. Norman, "Classification accuracies of physical activities using smartphone motion sensors," *Journal of Medical Internet Research*, vol. 14, no. 5, p. e130, 2012.

[60] L. N. V. Colon, Y. DeLaHoz, and M. Labrador, "Human fall detection with smartphones," in *Communications (LATINCOM), 2014 IEEE Latin-America Conference on*. IEEE, 2014, pp. 1–7.

[61] L.-J. Kau and C.-S. Chen, "A smart phone-based pocket fall accident detection, positioning, and

rescue system," *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 1, pp. 44–56, 2015.

[62] H. Ghasemzadeh, R. Jafari, and B. Prabhakaran, "A body sensor network with electromyogram and inertial sensors: multimodal interpretation of muscular activities," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 2, pp. 198–206, 2010.

[63] A. Leone, G. Rescio, A. Caroppo, and P. Siciliano, "A wearable emg-based system pre-fall detector," *Procedia Engineering*, vol. 120, pp. 455–458, 2015.

[64] T.-T. Nguyen, M.-C. Cho, and T.-S. Lee, "Automatic fall detection using wearable biomedical signal measurement terminal," in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*. IEEE, 2009, pp. 5203–5206.

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

[66] A. K. Bourke and G. M. Lyons, "A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor," *Medical Engineering & Physics*, vol. 30, no. 1, pp. 84–90, 2008.

[67] S.-W. Yang and S.-K. Lin, "Fall detection for multiple pedestrians using depth image processing technique," *Computer Methods and Programs in Biomedicine*, vol. 114, no. 2, pp. 172–182, 2014.

[68] E. E. Stone and M. Skubic, "Fall detection in homes of older adults using the microsoft kinect," *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 1, pp. 290–301, 2015.

[69] D. Litvak, Y. Zigel, and I. Gannot, "Fall detection of elderly through floor vibrations and sound," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*. IEEE, 2008, pp. 4632–4635.

[70] F. Sposaro and G. Tyson, "ifall: an android application for fall monitoring and response," in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*. IEEE, 2009, pp. 6119–6122.

[71] 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, 2006. ICTTA'06. 2nd*, vol. 1. IEEE, 2006, pp. 1003–1007.

[72] 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 *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2011 5th International Conference on*. IEEE, 2011, pp. 354–361.

[73] K. Chaccour, R. Darazi, A. H. el Hassans, and E. Andres, "Smart carpet using differential piezoresistive pressure sensors for elderly fall detection," in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2015 IEEE 11th International Conference on*. IEEE, 2015, pp. 225–229.

[74] A. Yazar, F. Keskin, B. U. Töreyin, and A. E. Çetin, "Fall detection using single-tree complex wavelet transform," *Pattern Recognition Letters*, vol. 34, no. 15, pp. 1945–1952, 2013.

[75] S. Tomii and T. Ohtsuki, "Falling detection using multiple doppler sensors," in *e-Health Networking, Applications and Services (Healthcom), 2012 IEEE 14th International Conference on*. IEEE, 2012, pp. 196–201.

[76] Y. Nizam, M. N. H. Mohd, and M. M. A. Jamil, "Human fall detection from depth images using position and velocity of subject," *Procedia Computer Science*, vol. 105, pp. 131–137, 2017.

[77] S.-G. Miaou, P.-H. Sung, and C.-Y. Huang, "A customized human fall detection system using omni-camera images and personal information," in *Distributed Diagnosis and Home Healthcare, 2006. D2H2. 1st Transdisciplinary Conference on*. IEEE, 2006, pp. 39–42.

[78] M. Kreković, P. Čerić, T. Dominko, M. Ilijaš, K. Ivančić, V. Skolan, and J. Šarlija, "A method

for real-time detection of human fall from video," in *MIPRO, 2012 Proceedings of the 35th International Convention*. IEEE, 2012, pp. 1709–1712.

[79] N. Thome and S. Miguet, "A hhmm-based approach for robust fall detection," in *Control, Automation, Robotics and Vision, 2006. ICARCV'06. 9th International Conference on*. IEEE, 2006, pp. 1–8.

[80] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Fall detection from human shape and motion history using video surveillance," in *Advanced Information Networking and Applications Workshops, 2007, AINAW'07. 21st International Conference on*, vol. 2. IEEE, 2007, pp. 875–880.

[81] H. Foroughi, A. Rezvanian, and A. Paziraee, "Robust fall detection using human shape and multi-class support vector machine," in *Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on*. IEEE, 2008, pp. 413–420.

[82] L. Hazelhoff, J. Han *et al.*, "Video-based fall detection in the home using principal component analysis," in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2008, pp. 298–309.

[83] L. Schwickert, C. Becker, U. Lindemann, C. Maréchal, A. Bourke, L. Chiari, J. Helbostad, W. Zijlstra, K. Aminian, C. Todd *et al.*, "Fall detection with body-worn sensors," *Zeitschrift für Gerontologie und Geriatrie*, vol. 46, no. 8, pp. 706–719, 2013.

[84] C. Medrano, I. Plaza, R. Igual, Á. Sánchez, and M. Castro, "The effect of personalization on smartphone-based fall detectors," *Sensors*, vol. 16, no. 1, p. 117, 2016.

[85] Y. Cao, Y. Yang, and W. Liu, "E-falld: A fall detection system using android-based smartphone," in *Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on*. IEEE, 2012, pp. 1509–1513.

[86] O. T.-C. Chen and C.-J. Kuo, "Self-adaptive fall-detection apparatus embedded in glasses," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International*

Conference of the IEEE. IEEE, 2014, pp. 4623–4626.

[87] M. Gjoreski, H. Gjoreski, M. Luštrek, and M. Gams, "How accurately can your wrist device recognize daily activities and detect falls?" *Sensors*, vol. 16, no. 6, p. 800, 2016.

[88] E. Casilari, J.-A. Santoyo-Ramón, and J.-M. Cano-García, "Analysis of public datasets for wearable fall detection systems," *Sensors*, vol. 17, no. 7, p. 1513, 2017.

[89] "tfall: Eduqtech dataset." [Online]. Available: http://eduqtech.unizar.es/fall-adl-data

[90] C. Medrano, R. Igual, I. Plaza, and M. Castro, "Detecting falls as novelties in acceleration patterns acquired with smartphones," *PloS one*, vol. 9, no. 4, p. e94811, 2014.

[91] J. He, S. Bai, and X. Wang, "An unobtrusive fall detection and alerting system based on kalman filter and bayes network classifier," *Sensors*, vol. 17, no. 6, p. 1393, 2017.

[92] A. T. Özdemir, "An analysis on sensor locations of the human body for wearable fall detection devices: Principles and practice," *Sensors*, vol. 16, no. 8, p. 1161, 2016.

[93] "Implementing a principal component analysis (pca)." [Online]. Available: http://sebastianraschka.com/Articles/2014_pca_step_by_step.html

[94] "Principal component analysis in 3 simple steps." [Online]. Available: http://sebastianraschka.com/Articles/2015_pca_in_3_steps.html

[95] H.-P. Kriegel, P. Kröger, and A. Zimek, "Outlier detection techniques," *Tutorial at KDD*, vol. 10, 2010.

[96] H.-P. Kriegel, A. Zimek *et al.*, "Angle-based outlier detection in high-dimensional data," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2008, pp. 444–452.

[97] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation-based anomaly detection,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, p. 3, 2012.

[98] K. A. Heller, K. M. Svore, A. D. Keromytis, and S. J. Stolfo, “One class support vector machines for detecting anomalous windows registry accesses,” in *Proc. of the Workshop on Data Mining for Computer Security*, vol. 9, 2003.

[99] L. M. Manevitz and M. Yousef, “One-class svms for document classification,” *Journal of Machine Learning Research*, vol. 2, no. Dec, pp. 139–154, 2001.